THIELE CENTRE
FOR APPLIED MATHEMATICS IN NATURAL SCIENCE

A new algorithm for 3D reconstruction from support functions

Richard J. Gardner and Markus Kiderlen

Research Report

No. 15 I September 2007

A new algorithm for 3D reconstruction from support functions

Richard J. Gardner and Markus Kiderlen

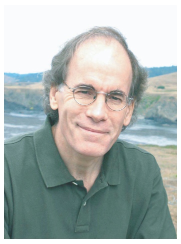
Abstract

We introduce a new algorithm for reconstructing an unknown shape from a finite number of noisy measurements of its support function. The algorithm, based on a least squares procedure, is very easy to program in standard software such as Matlab and allows, for the first time, good 3D reconstructions to be performed on an ordinary PC. Under mild conditions, theory guarantees that outputs of the algorithm will converge to the input shape as the number of measurements increases. Reconstructions may be obtained without any pre- or post-processing steps and with no restriction on the sets of measurement directions except their number, a limitation dictated only by computing time.

In addition we offer a linear program version of the new algorithm that is much faster and better, or at least comparable, in performance at low levels of noise and reasonably small numbers of measurements. Another modification of the algorithm, suitable for use in a "focus of attention" scheme, is also described.

Index Terms

Support function, convex body, geometric tomography, algorithm.



Richard J. Gardner holds B.Sc. and Ph.D. degrees in mathematics from University College London and was awarded a D.Sc. degree from the University of London in 1988 for contributions to measure theory and convex geometry. He has held positions at universities and research institutions in several countries and has been Professor of Mathematics at Western Washington University since 1991. He founded Geometric Tomography, an area of geometric inverse problems involving data concerning sections by and projections on lines or planes, and published a book on the subject in 1995.

Markus Kiderlen received an Erasmus certificate in mathematics at the University Joseph Fourier, France, in 1990, and a M.Sc. degree at the University of Karlsruhe, Germany, in 1993. At the same university, he got a Ph.D. degree in mathematics in 1999. He is currently working as assistant professor at the University of Aarhus, Denmark, and is one of the three directors of the Histoinformatics project. This project, funded by a major grant from the Danish Council for Strategic Research, aims to include cutting-edge mathematical tools into image analysis software for medical and biological applications. His research interests include convex geometry, geometric tomography, and stereology, the latter in particular in a digital setting.

Supported in part by U.S. National Science Foundation grant DMS-0603307, by the Carlsberg Foundation, and by the Danish Council for Strategic Research.

I. Introduction

THIS paper addresses the problem of reconstructing an unknown shape from a finite number of noisy measurements of its support function. Given a measurement direction, i.e., a unit vector, the corresponding support function value provides the signed distance from some fixed reference point, usually taken to be the origin, to the supporting line (for 2D shapes) or plane (for 3D shapes) to the shape orthogonal to the direction. This is illustrated for 2D shapes in Fig. 1, where the support function and support line of the shape K in the direction u are $h_K(u)$ and $H(u)$, respectively.

Support functions have found application in many different settings. Prince and Willsky [1] used such data in computerized tomography as a prior to improve performance, particularly when only limited data is available. Lele, Kulkarni, and Willsky [2] applied support function measurements to target reconstruction from range-resolved and Doppler-resolved laser radar data. Gregor and co-workers proposed using support function data in a "focus of attention" pre-processing scheme for reducing the computational cost associated with positron emission tomography [3], projection magnetic resonance imaging [4], and cone-beam based computerized tomography [5]. A different sort of application is made by Ikehata and Ohe [6], who obtain support function information from electrical impedance tomography data, with a view to finding cavities.

The support function was introduced by the mathematical genius Minkowski, and is of fundamental importance in convex geometry and geometric tomography; see [7] and [8]. The role of the support function in mathematical morphology is described by Serra [9]. Karl, Kulkarni, Verghese, and Willsky [10] note that support function measurements can be obtained from repeated grasps by a parallel-jaw gripper, as in [11] and [12]. They also list references to articles detailing the application of support functions to geometric probing, robot vision, and chemical component analysis. Ghosh and Kumar [13] provide a detailed survey of the support function in geometric computing, where they compare its importance in the representation, manipulation, and analysis of convex bodies to that of the Fourier transform in signal processing.

The first algorithms for reconstructing shapes from noisy support function measurements were proposed by Prince and Willsky [1] and Lele, Kulkarni, and Willsky [2]. The method in each case is to fit a polygon to the data by a least squares procedure. In contrast, Fisher, Hall, Turlach, and Watson [14] use spline interpolation and the so-called von Mises kernel to fit a smooth curve to the data. This procedure was taken up in [15] and [16], the former dealing with convex bodies with corners and the latter giving an example to show that the algorithm in [14] may fail for a given data set. All these papers consider only the 2D case. A theoretical study of the support function reconstruction problem in higher dimensions was carried out by Karl, Kulkarni, Verghese, and Willsky [10].

The nature of the data arising from support function measurements makes it natural to focus on convex bodies. Given a finite set of measurements, one can attempt to approximate the unknown shape by a convex polyhedron. If, as in some previous schemes, each face of the reconstructed polyhedron is to be orthogonal to a measurement direction, the obvious variables become the distances of the hyperplanes containing these faces from the fixed reference point. A least squares procedure then involves finding values of these variables that correspond to the "best" finite set of hyperplanes that contain the faces of an approximating convex polyhedron. The main theoretical difficulty lies in finding a constraint that guarantees that the hyperplanes really do contain the faces of a convex polyhedron, a property called *consistency* in the sequel. (See Section III and Fig. 2.) While this is quite easy to do in 2D—in [1] and [2] the constraint is formulated in terms of a matrix-vector inequality—it is much more difficult in higher dimensions. In [10] this problem was solved by constructing a set of "local" consistency tests that if satisfied guarantee "global" consistency. Perhaps due to the inherent computational complexity of their method, the authors of [10] did not describe any implementation, even in 3D. But later, Gregor and Rannou [4] accomplished a clever implementation in 3D (for certain sets of measurement directions; see Section IX), using several tricks to cut down the computational complexity to a manageable level. We are not aware, however, of any published algorithm for 3D support function reconstruction that is simple, effective, and easy to program.

In this paper we introduce a new algorithm for reconstructing shapes from support function measurements. Like many previous algorithms, the idea is to use k support function measurements and least squares optimization to find an n-dimensional convex polyhedron that approximates the n-dimensional unknown shape. The novel feature in our algorithm is a different choice of variables. While there are more real variables than before (kn instead of k), the new choice of variables allows the consistency constraint to be specified by only $O(k^2)$ linear inequalities. In particular, the local consistency tests of [10] are entirely circumvented. The result is an effective support function algorithm that is very easy to program in standard software such as Matlab, and that allows, for the first time, good 3D reconstructions to be performed on an ordinary PC. Moreover, provided the number of measurements is not too large, reconstructions may be obtained without restriction on the sets of measurement directions and without any pre- or post-processing steps.

The new algorithm is described in Section IV. In Section VI we offer a linear program version of the new algorithm, which is much faster and comparable in performance to the least squares version at low levels of noise. Another modification, suitable for use in a "focus of attention" scheme, is the subject of Section X.

II. Background

Suppose that K is a convex body in n-dimensional Euclidean space $\Re^n$ and u is a unit vector in the unit sphere S^{n-1}. The *support function* $h_K(u)$ of K is

$$h_K(u) = \sup_{x\in K} x^T u. \tag{1}$$

Then $h_K(u)$ is the signed distance from the origin 0 to the *support hyperplane*

$$H(u) = \{x \in \Re^n \,|\, x^T u = h_K(u)\} \tag{2}$$

to K orthogonal to u. See Fig. 1. A convex body is completely determined by its support function, which is a continuous function on the unit sphere; see [7, p. 38].

In 2D, it is often convenient to write the unit vector $u = [\cos\theta, \sin\theta]^T$. In this case the support function of K becomes a function $h_K(\theta)$ of the angle θ.

The most useful metrics for calculating the distance between convex bodies can be defined by means of the support function. The *Hausdorff distance* between two convex bodies K and L is

$$\delta_H(K, L) = \|h_K - h_L\|_\infty = \sup_{u\in S^{n-1}} |h_K(u) - h_L(u)|.$$

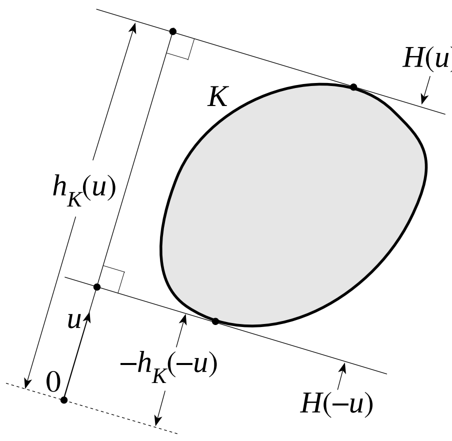


Fig. 1. The support function h_K of a convex body K

The corresponding L_2 *distance* is

$$\delta_2(K,L) = \|h_K - h_L\|_2 = \left(\int_{S^{n-1}} (h_K(u) - h_L(u))^2 \, du\right)^{1/2}, \tag{3}$$

where integration is over the unit sphere with respect to its natural measure. The two metrics are closely related. If K and L are contained in a fixed ball of radius R, then

$$c_1\,\delta_H(K,L)^{(n+1)/2} \le \delta_2(K,L) \le c_2\,\delta_H(K,L), \tag{4}$$

where c_1 and c_2 are constants depending only on n and R; see [17, Proposition 2.3.1].

III. The Prince-Willsky algorithm

Let $u_1, \dots, u_k$ be fixed vectors in S^{n-1} whose positive hull

$$\{x \in \Re^n \,|\, x = \sum_{i=1}^{k} \lambda_i u_i \text{ for } \lambda_i \ge 0\}$$

is $\Re^n$. If h_i is a support function measurement of an unknown shape in the direction u_i, we call the pair (u_i, h_i) a support function data point.

For the purposes of reconstruction, it is important to know for which real numbers $h_1, \dots, h_k$ there is a convex body L in $\Re^n$ that fits the support function data $((u_1,h_1), \dots, (u_k,h_k))$, in other words, is such that $h_L(u_i) = h_i$, $i = 1, \dots, k$. In this case the real numbers $h_1, \dots, h_k$ are called *consistent*. This is not always the case, as Fig. 2 illustrates.

If $h_1, \dots, h_k$ are consistent, there will be many convex bodies L that fit the support function data $((u_1,h_1), \dots, (u_k,h_k))$; let $P(h_1, \dots, h_k)$ denote the one that is the convex polyhedron defined by

$$P(h_1, \dots, h_k) = \bigcap_{i=1}^{k} \{x \in \Re^n \,|\, x^T u_i \le h_i\}. \tag{5}$$

See Fig. 3, from which it is clear that $P(h_1, \dots, h_k)$ is the maximal convex body that fits the support function data $((u_1,h_1), \dots, (u_k,h_k))$, because any other such convex

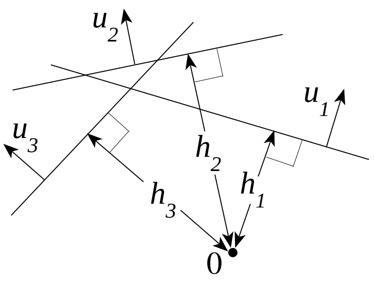


Fig. 2. Inconsistent support function data

body, such as K or the dotted polygon in Fig. 4, must be inscribed in $P(h_1, \ldots, h_k)$. (In these figures, only two of the six directions u_i and two of the six distances h_i are shown.)

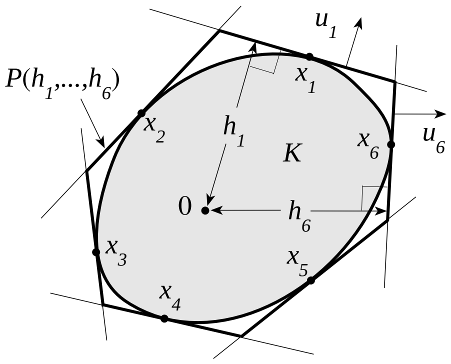


Fig. 3. The polygon $P(h_1, \ldots, h_k)$ (with $k = 6$)

For $n = 2$ and vectors $u_1, \ldots, u_k$ equally spaced in S^1, the following algorithm was proposed and implemented by Prince and Willsky [1].

Prince-Willsky Algorithm

Input: Natural numbers $n \geq 2$ and $k \geq n + 1$; vectors $u_i \in S^{n-1}$, $i = 1, \ldots, k$ whose positive hull is $\Re^n$; noisy support function measurements

$$y_i = h_K(u_i) + X_i, \tag{6}$$

$i = 1, \ldots, k$ of an unknown convex body K in $\Re^n$, where the X_i's are independent normal $N(0, \sigma^2)$ random variables.

Task: Construct a convex polyhedron $\hat{P}_k$ in $\Re^n$ that approximates K.

Action: Solve the following constrained linear least squares problem (LLS1):

$$\min_{h_1,\ldots,h_k} \quad \sum_{i=1}^{k} (y_i - h_i)^2, \tag{7}$$

$$\text{subject to} \quad h_1, \ldots, h_k \text{ are consistent.} \tag{8}$$

Let $\hat{h}_1, \ldots, \hat{h}_k$ be a solution of (LLS1) and let $\hat{P}_k = P(\hat{h}_1, \ldots, \hat{h}_k)$.

The model for noise in the Prince-Willsky algorithm is a natural choice. Support functions are typically measured using electronic sensor devices such as cameras, robot tools, grippers, etc., so the noise corrupting the measurements is generally the electrical noise coming from the sensors. This noise arises during data acquisition (readout noise) and can be approximated by a Gaussian distribution.

Of course, any implementation of the Prince-Willsky Algorithm involves making explicit the constraint (8). For $n = 2$, this was done by Prince and Willsky [1] for vectors $u_1, \dots, u_k$ equally spaced in S^1, and by Lele, Kulkarni, and Willsky [2] for arbitrary vectors $u_1, \dots, u_k$ in S^1, by means of an inequality constraint of the form $Ch \geq 0$, where $h = [h_1, \dots, h_k]^T$ and C is a certain matrix given explicitly in [2, p. 1696]. The matrix C depends only on the measurement directions $u_1, \dots, u_k$ (or, equivalently, on the corresponding angles $\theta_1, \dots, \theta_k$) and this allows a very efficient implementation of the algorithm in the 2D case.

The Prince-Willsky algorithm above actually corresponds to the *Closest Algorithm* of [1, Section IV] and Algorithm NUA of [2, Section 3]. These algorithms produce approximating polygons whose edges have unit outer normal vectors belonging to the set $\{u_1, \dots, u_k\}$ of measurement directions. Prince and Willsky [1] described in addition several variants of the algorithm, each involving prior knowledge of some kind, for example, information about the curvature or position of the unknown shape. Lele, Kulkarni, and Willsky [2] also listed related algorithms: BNGON, in which a number N of sides of the output polygon is specified in advance, and BNGONROT, in which both the number of the edges of the output polygon and their unit outer normal vectors are specified. (A nonlinear least squares procedure which also optimizes the orientations of the edges is described by Poonawala, Milanfar, and Gardner [18].)

Even in the 3D case, it is no longer a simple matter to deal with the consistency constraint (8). An exhaustive study of the n-dimensional case was carried out by Karl, Kulkarni, Verghese, and Willsky [10], who reduced (8) to a set of *local* consistency conditions. Roughly speaking, each support function data point (u_i, h_i) must be checked for consistency relative to each set of n such points for which u_i lies in the positive hull of the corresponding n directions. Perhaps due to the computational infeasibility of checking all such local consistency conditions, no implementation was described in [10]. However, such an implementation was undertaken by Gregor and Rannou [4], who made several clever reductions in the number of local consistency conditions by using special features of the set of directions employed (see Section IX for more details).

IV. The new algorithm

Our new algorithm is easily described. Its input and task are exactly as for the Prince-Willsky algorithm in the previous section. The essential difference is that the least squares variables are now vectors x_i in $\Re^n$ instead of real numbers h_i and the constraint (10) is explicit.

New Support Function Algorithm

Action: Solve the following constrained linear least squares problem (LLS2):

$$\min_{x_1\in\Re^n,\ldots,x_k\in\Re^n} \quad \sum_{i=1}^{k}(y_i - x_i^T u_i)^2, \tag{9}$$

$$\text{subject to} \quad x_j^T u_i \le x_i^T u_i \text{ for } 1 \le i \ne j \le k, \tag{10}$$

where the measurements y_i are given by (6).

Let $\hat{x}_1, \ldots, \hat{x}_k$ be a solution of (LLS2) and let $\hat{P}_k$ be the convex hull of $\{\hat{x}_1, \ldots, \hat{x}_k\}$.

To understand how the new algorithm works, suppose that K is a convex body in $\Re^n$ and $1 \le i \le k$ is fixed. Then there is at least one point x_i in K contained in the supporting hyperplane $H(u_i)$ to K; see Fig. 3. By the definitions (1) and (2) of the support function and supporting hyperplane, when $1 \le j \le k$, we have

$$x_j^T u_i \le h_K(u_i) = x_i^T u_i,$$

so the constraint (10) is satisfied. If the measurements are exact ($\sigma = 0$), then $\hat{P}_k$ is a convex polygon with the same support function values as K in the measurement directions $u_1, \ldots, u_k$.

The basic features of the new algorithm versus those of the Prince-Willsky Algorithm are easily summarized. The main disadvantage of the Prince-Willsky Algorithm is that the consistency constraint (8) is hard to make explicit except in 2D, but on the positive side, the number k of real variables in the objective function (7) is small. In the new algorithm, the consistency constraint is always completely explicit in (10), and indeed requires only $k(k-1)$ linear inequality constraints. On the other hand, the new algorithm requires, in n dimensions, kn real variables in the objective function (9).

Another difference between the two algorithms is inherited from the choice of variables. In the Prince-Willsky algorithm, the output polygon is unique and each of its edges is orthogonal to one of the measurement directions. By contrast, due to the larger number of variables, the output polygon of the new algorithm is not unique and its edges do not have any particular orientation. When the support function measurements are exact ($\sigma = 0$), the bold polygon in Fig. 3 corresponds to the output of the Prince-Willsky algorithm and the dotted polygon in Fig. 4 is a possible output of the new algorithm. As explained before, the convex hull of $\{x_1, \ldots, x_6\}$ in Fig. 3 is another possible output.

Nevertheless, there is a simple connection between the real variables h_i used in the Prince-Willsky algorithm and the variables $x_i \in \Re^n$ employed in the new algorithm. Since the output of the Prince-Willsky algorithm is unique, we have $h_i = x_i^T u_i$ for each i and the outputs of both algorithms have exactly the same support function measurements in the directions $u_1, \ldots, u_k$. Indeed, it is very easy to modify the new algorithm so that its output is the same as that of the Prince-Willsky algorithm, simply by defining

$$\hat{P}_k = P(\hat{x}_1^T u_1, \ldots, \hat{x}_k^T u_k) \tag{11}$$

instead. This choice of the output is appropriate whenever a superset of the unknown shape is required, for example in applications involving a "focus of attention" scheme; see Section X for more details. However, our choice of the output polygon is more natural in view of the new variables, and moreover has some slight advantages described in Section VII.

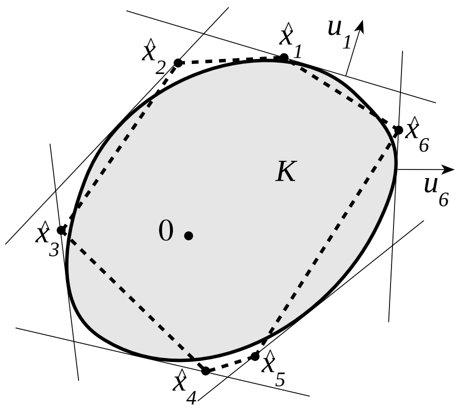


Fig. 4. Possible output (dotted) of the new algorithm when $\sigma = 0$

V. Convergence

Gardner, Kiderlen, and Milanfar [19] provided the first proof that the Prince-Willsky algorithm converges, under mild assumptions. In [19, Theorem 6.1], it is shown that if K is a convex body in $\Re^n$, (u_i) is an *evenly spread* infinite sequence of directions in S^{n-1}, and $\hat{P}_k$ is an output from the Prince-Willsky algorithm based on measurements in the first k directions from the sequence (u_i), then the Hausdorff distance $\delta_H(K, \hat{P}_k)$ and L_2 distance $\delta_2(K, \hat{P}_k)$ converge to zero as k approaches infinity. The exact meaning of “evenly spread” is given in [19, p. 1337]; suffice it to say that it is a weak restriction and satisfied by many natural sequences. Under a slightly stronger, but still easily satisfied, condition on (u_i) and other mild assumptions, [19, Theorem 6.2] even estimates the rate of convergence of $\delta_2(K, \hat{P}_k)$ to be $O(k^{-2/(n+3)})$. This holds provided the noise level σ is fixed, K is contained in a ball of a fixed known radius, and the dimension $n \leq 4$. The relation (4) allows a corresponding estimate for convergence in terms of the Hausdorff metric, but the L_2 metric is more natural in view of the least squares procedure involved. The convergence rates in [19] were obtained by an application of the powerful theory of empirical processes, which, incidentally, also suggests that the just-mentioned convergence rate is optimal. (Convergence rates for dimensions $n > 4$ are also given in [19, Theorem 6.2] but these need not be optimal.)

We observed in Section IV that the outputs of both the Prince-Willsky and the new algorithm have exactly the same support function measurements in the directions $u_1, \ldots, u_k$. This implies that the *mean square error*

$$\mathrm{MSE}\,(K, \hat{P}_k) = \left(\frac{1}{k} \sum_{i=1}^{k} \left(h_K(u_i) - h_{\hat{P}_k}(u_i) \right)^2 \right)^{1/2} \tag{12}$$

between K and the output polygon $\hat{P}_k$ is the same for both algorithms. With this in hand, the rest of the analysis in [19] then applies equally to the new algorithm, providing convergence, and the same rates of convergence, under the same assumptions.

VI. A linear program version

A linear program version of the new support function algorithm can be obtained by replacing the linear least squares problem (LLS2) of Section IV with the following linear program (LP):

$$\begin{aligned} \min_{x_1\in\Re^n,\ldots,x_k\in\Re^n} \quad & \sum_{i=1}^{k}(y_i - x_i^T u_i), && (13)\\ \text{subject to} \quad & x_j^T u_i \le x_i^T u_i \le y_i && (14)\\ & \text{for } 1 \le i \ne j \le k, \end{aligned}$$

where the measurements y_i are given as before by (6).

Suppose that (LP) has a solution $\hat{x}_1, \ldots, \hat{x}_k$ with objective function value in (13) equal to zero. Then it is easy to check that $\hat{x}_1, \ldots, \hat{x}_k$ is also a solution of (LLS2), and the convex hull of $\{\hat{x}_1, \ldots, \hat{x}_k\}$ is a possible output of the new support function algorithm. This situation will occur when the measurements are exact ($\sigma = 0$), so solving (LP) is then an alternative to solving (LLS2), and this could happen even when the measurements are noisy ($\sigma > 0$). If the minimal objective function value in (13) is positive, however, then a solution of (LP) will not in general be a solution of (LLS2) and the latter must be solved. Nevertheless, the solution $\hat{x}_1, \ldots, \hat{x}_k$ of (LP) might be a good choice as initial values in solving (LLS2), at least when the noise level is small. In fact, as we shall see in Sections VII and VIII, the linear program version of the new algorithm is a viable alternative to the least squares version at low levels of noise, and, since it is also much faster, may be preferable at a zero or very low level of noise.

Another possible use of the linear program (LP) is as a test for consistency. Suppose that $h_1, \ldots, h_k$ are real numbers and $u_1, \ldots, u_k$ are directions in S^{n-1}. If y_i is replaced by h_i in (LP) for $1 \le i \le k$ and the minimal objective function value is zero, then $\hat{x}_i^T u_i = h_i$ for $1 \le i \le k$. This means that for $1 \le i \le k$, h_i is a support function measurement in the direction u_i for the polyhedron $P(h_1, \ldots, h_k)$ defined by (5), so $h_1, \ldots, h_k$ are consistent. It is easily verified that, conversely, if the real numbers $h_1, \ldots, h_k$ are consistent, then the minimal objective function value in (LP) is zero. Thus the linear program (LP) might allow the consistency tests in [10] to be avoided when the dimension $n \ge 3$ and they become tedious to implement.

VII. Experimental results in 2D

The Prince-Willsky and new support function algorithms were implemented with Matlab programs. The experiments described in this and the next section were run on a standard PC with a 2.4 GHz Pentium 4 processor and 512 MB RAM.

In this section, we give in detail some results in 2D, where it is possible to compare the Prince-Willsky and new algorithms. We begin by discussing results concerning the least squares version of the new algorithm (see Section IV). In the figures, this is indicated by "new (LS)".

Fig. 5 illustrates sample 2D reconstructions of an irregular 7-sided polygon from 17 support function measurements taken in directions spaced equally around the unit circle and corrupted by noise with $\sigma = 0.1$.

The result is typical of our observations; both algorithms perform well and, when the number of measurements is not very small, they produce similar-looking output

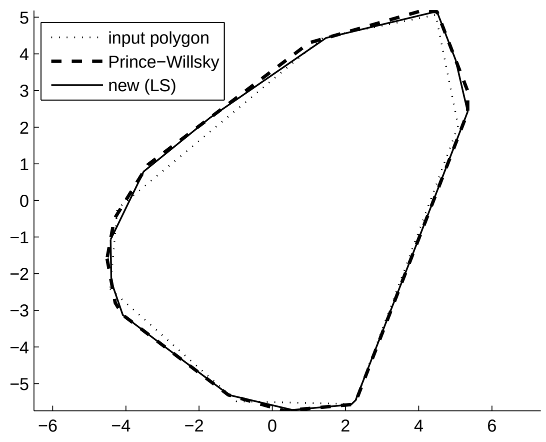


Fig. 5. Reconstructions of a polygon

polygons. This is corroborated by Figs. 6 to 8, each of which show the average, maximum, and minimum error between input and output polygons for a Monte Carlo run of 100 reconstructions for each number of measurements ranging from seven to 49 in steps of three. The input polygon (shown as the lightly dotted polygon in Fig. 14) was the image of a regular 11-gon under a certain linear transformation, somewhat arbitrary but corresponding to a dilation, rotation, and stretching factors designed to reduce the effect of any "matching" of polygon orientation and measurement directions. Measurements were always taken in directions equally spaced around the unit circle, and the noise level was fixed at $\sigma = 0.1$.

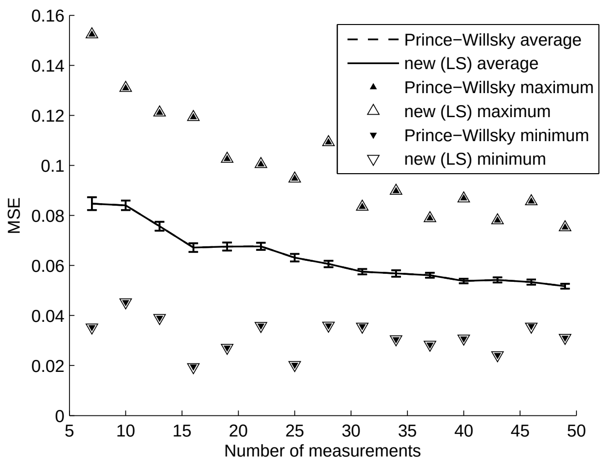


Fig. 6. MSE (least squares version)

Fig. 6 depicts the mean square error, given by (12), between input and output polygons. The average error over the 100 reconstructions is represented by the graph and the maximum and minimum errors are indicated by the triangular markers. As expected from our remark in Section V, the mean square error for the Prince-Willsky and new algorithms was so nearly identical that only one graph is visible and the two set of markers coincide.

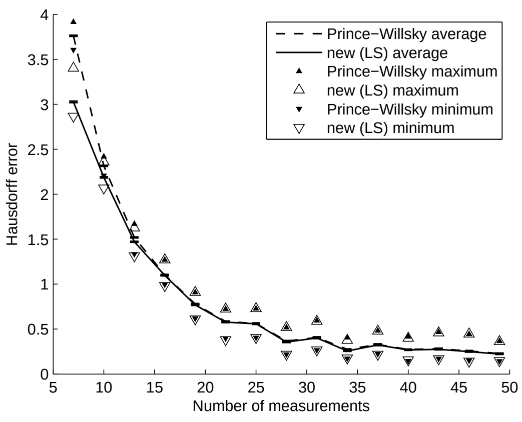


Fig. 7. Hausdorff error (least squares version)

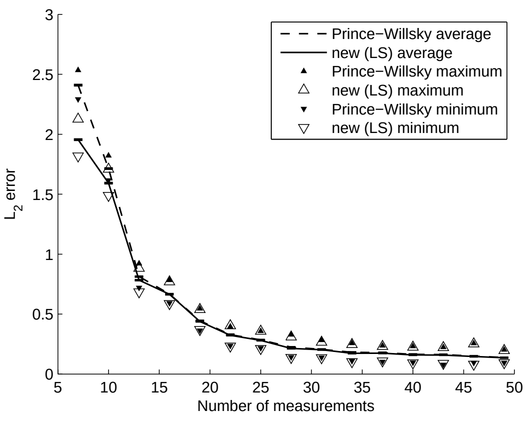


Fig. 8. L_2 error (least squares version)

In Figs. 7 and 8, however, which show the Hausdorff and L_2 distance, respectively, between input and output polygons, we see a noticeable difference between the two algorithms, at least for small numbers of measurements. In particular, the new algorithm tends to produce a smaller Hausdorff and L_2 error, an effect also visible in Fig. 5.

In rare cases the Prince-Willsky algorithm does yield smaller errors. These experimental results can be explained as follows. In the case of a reconstruction via the Prince-Willsky algorithm with no noise, the output polygon contains the input shape (cf. the bold polygon in Fig. 3). Thus, unless the output polygon is identical to the input shape, it *always* overestimates. Again, unless it is identical to the output polygon of the Prince-Willsky algorithm, the corresponding output polygon of the new algorithm will *always* be smaller (cf. the dotted polygon in Fig. 4), and so will have significantly less tendency to overestimate.

For 2D reconstructions, however, the new algorithm pays a high price for the double number of variables. Fig. 9 shows the average time taken for reconstructions in the Prince-Willsky and new algorithms. Whereas the Prince-Willsky algorithm is extremely fast for numbers of measurements up to 150 or so—in Fig. 9 the corresponding graph is almost invisible, since the reconstruction time is still tiny even for 49 measurements—the new algorithm slows significantly as these numbers increase, becoming rather tedious at around 60.

We now turn to the performance of the linear program version of the new algorithm (see Section VI). In the figures this is indicated by "new (LP)".

Figs. 10 and 11 show the Hausdorff and L_2 distance, respectively, between input and output polygons, for a Monte Carlo run of 100 reconstructions for the same 11-gon, same noise level ($\sigma = 0.1$), and the same numbers of measurements as for the least squares version described above.

At this level of noise, the linear program version of the new algorithm performs reasonably well. In fact, Figs. 7 and 10 indicate that in terms of the Hausdorff error, there is little difference between the least squares and linear program versions of the new algorithm. However, Figs. 8 and 11 are rather different, and show a

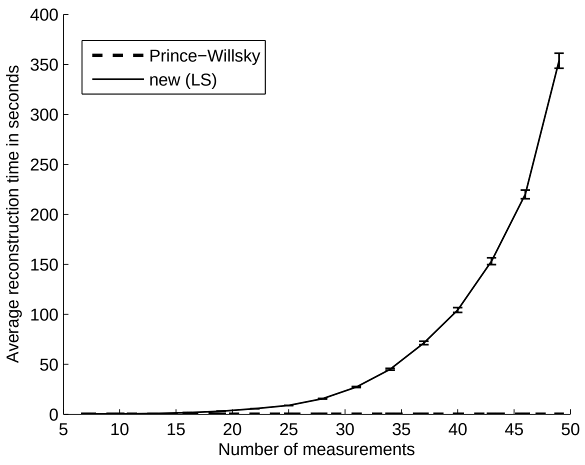


Fig. 9. Average reconstruction time (least squares version)

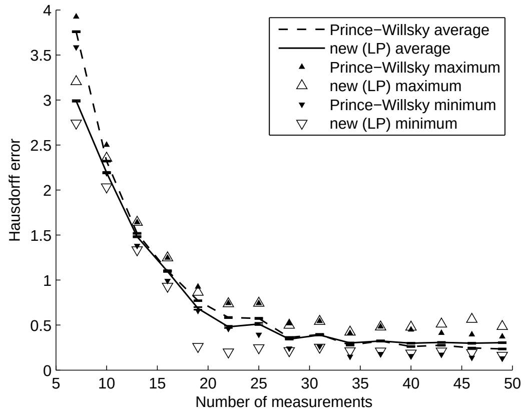


Fig. 10. Hausdorff error (linear program version)

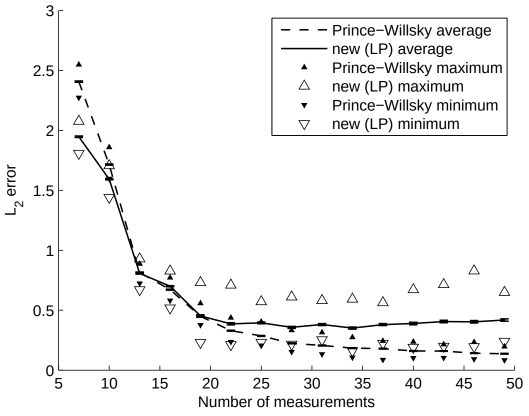


Fig. 11. L_2 error (linear program version)

loss in performance in the linear program version as the number of measurements increases beyond about 25 (the L_2 distance, defined by (3), is more appropriate when considering the global error between input and output). A steady divergence in performance between the linear program version and the Prince-Willsky algorithm as the number of measurements increases is also shown by the MSE error (we omit the graph).

The linear program version of the new algorithm is therefore quite satisfactory for low levels of noise, or medium levels of noise and small numbers of measurements. In such situations it has one distinct advantage over the least squares version: It is much quicker, as Figs. 9 and 12 demonstrate.

As the noise level increases, however, the linear program version becomes less effective. Fig. 13 depicts the L_2 error as the noise level increases from $\sigma = 0$ to $\sigma = 0.5$ in steps of 0.05, for a Monte Carlo run of 100 reconstructions. Here the same input 11-gon is used as before, and the number of measurements is fixed at 25. The least squares version of the new algorithm appears to be quite robust as the noise level increases, with reasonably low errors even when $\sigma = 0.5$. Similar results

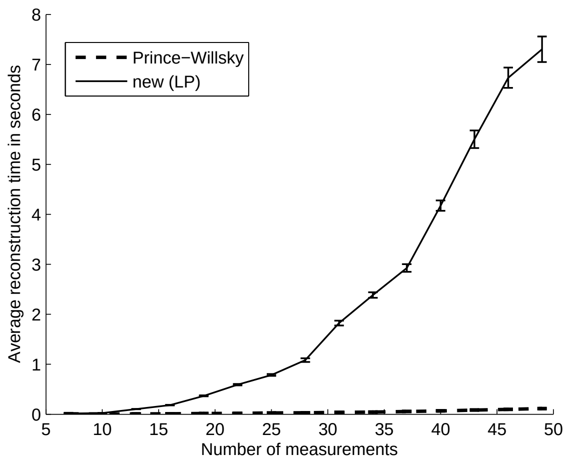


Fig. 12. Average reconstruction time (linear program version)

are obtained with the MSE and Hausdorff errors.

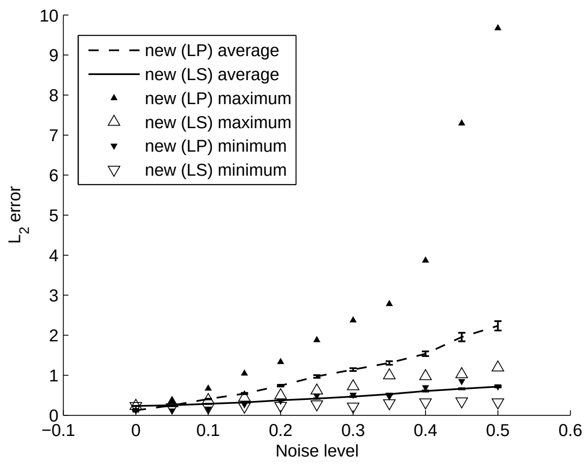


Fig. 13. L_2 error for least squares and linear program versions

On the other hand, the linear program version becomes much worse, and this deterioration of performance relative to the least squares version also occurs for any nonzero level of noise as the number of measurements increases. The reason for this phenomenon is the constraint (14), which causes an inappropriate reduction in size in the output shape whenever a measurement y_i is too small. An extreme case of this effect is clearly illustrated in Fig. 14. The same 11-gon was reconstructed using both the least squares and linear program versions of the new algorithm, with 35 measurements and a high level of noise ($\sigma = 1$). The least squares version still performs reasonably well, but the output polygon for the linear program version has suffered an enormous contraction.

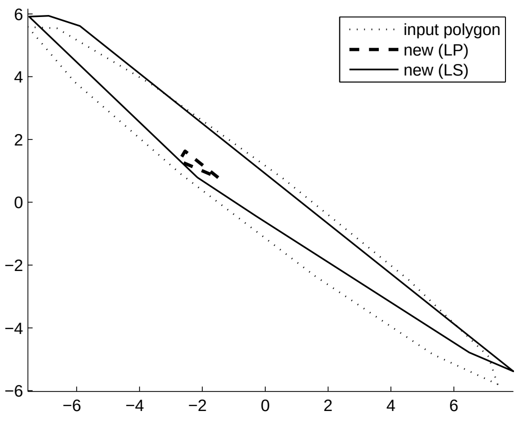


Fig. 14. Reconstructions for least squares and linear program versions with high noise level

VIII. Experimental results in 3D

In this section we describe experimental results in 3D. The discussion is limited to the two versions of our new algorithm, since it is not viable at present to make a comparison with the only other program for 3D support function reconstruction, that described by Gregor and Rannou [4] (see Section IX for further comments).

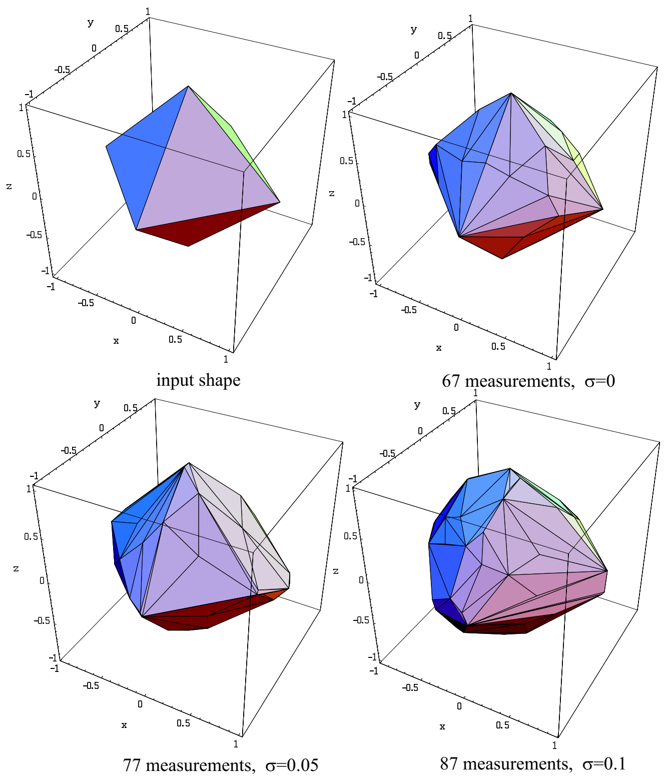


Fig. 15. Reconstruction of an octahedron (least squares version)

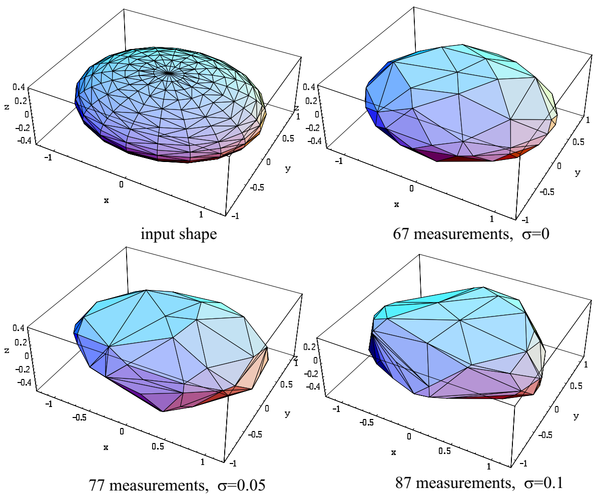


Fig. 16. Reconstruction of an ellipsoid (least squares version)

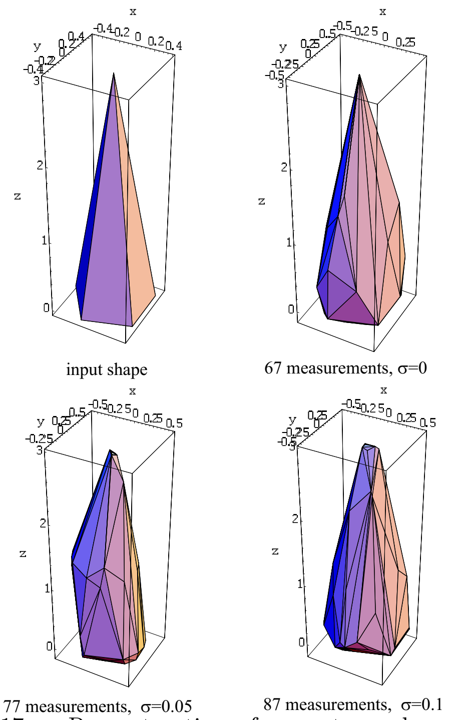


Fig. 17. Reconstruction of a pentagonal pyramid (least squares version)

In the absence of any a priori information, it is clearly desirable to use sets of measurement directions that are as "spread out" as possible. (Note that our algorithm does not *require* the measurement directions to be specially configured in any way.) We took our sets of measurement directions from a database of the best known sets of directions in this sense. (See www.research.att.com/~njas/grass/dim3/ on the web site of N. Sloane. Some of these sets are optimal, but in general the optimal spacing is not known.) However, since some of the sets of directions in this database have a special orientation that tends to align with the symmetries of our input shapes, we also rotated each measurement direction by a fixed amount, specifically, by $(\theta, \varphi) = (0.2, 0.7)$ radians in spherical polar coordinates, to reduce the chance of any such alignment.

Figs. 15, 16, and 17 show pictures produced with Mathematica (which, for 3D shapes, we find more helpful than Matlab) of reconstructions of a regular octahedron, an ellipsoid, and a pyramid with pentagonal base. In each of these figures the input shape is depicted at the top left, and reconstructions are shown with 67 measurements and $\sigma = 0$ (top right), 77 measurements and $\sigma = 0.05$ (bottom left), and 87 measurements and $\sigma = 0.1$ (bottom right). It is important to bear in mind that even with no noise, there are infinitely many different convex polyhedra whose support function measurements agree in the 67 directions (or any other finite set of directions) *exactly* with those of the input shape. The average relative errors in the support function measurements in the chosen directions between input and output shapes are in each case less than 0.0005, 0.035, and 0.05, for the reconstructions with $\sigma = 0$, 0.05, and 0.1, respectively.

The reconstructions shown in Figs. 15, 16, and 17 each took from one to a few hours on the PC described at the beginning of Section VII. It was noted in Sec-

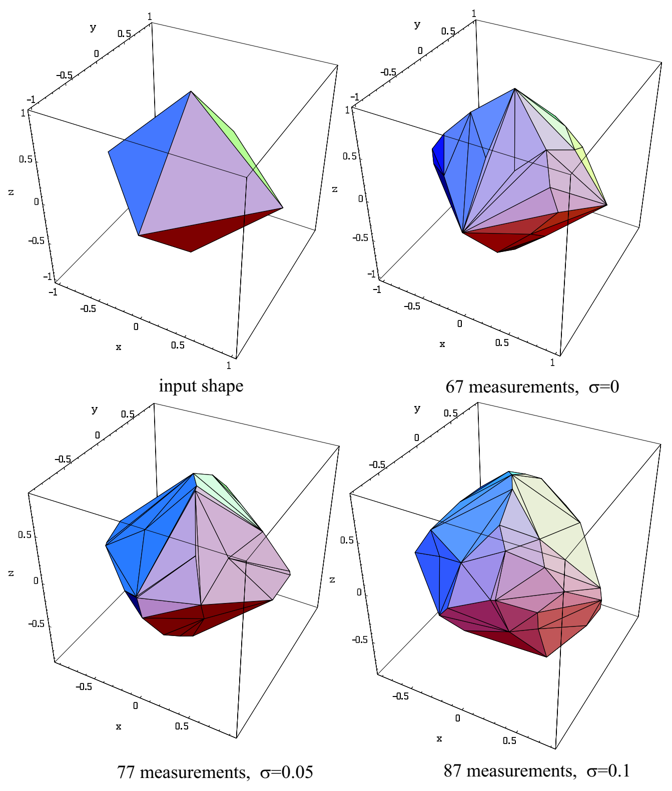


Fig. 18. Reconstruction of an octahedron (linear program version)

tion VII that the linear program version of the new algorithm is considerably faster, and the difference becomes more dramatic in 3D. Fig. 18 shows reconstructions of the octahedron using the linear program version, and each took only from two to five minutes. The average relative errors in the support function measurements in the chosen directions between input and output shapes are approximately (and coincidentally) 0, 0.05, and 0.1, for the reconstructions with $\sigma = 0$, 0.05, and 0.1, respectively. This is compatible with a visual comparison of Figs. 15 and 18, which indicates that the linear program version does as good a job with low levels of noise as the least squares version. Indeed, when $\sigma = 0$ the linear program version is exceedingly accurate and relative errors are essentially zero. However, at a noise level $\sigma = 0.1$ (and the relatively high number of 87 measurements) the least squares version is to be preferred if computing time is not an issue.

We stress that Figs. 15–18 are genuine in that they were not specially selected from a number of repeated reconstructions, which of course vary in quality due to the noise. It should also be mentioned that although no post-processing was used in these reconstructions, this would be possible if desired. In [18, Section 3.3], several methods, such as clustering, are described for reducing the number of edges of the output polygon in 2D reconstruction. It should be possible to extend these methods to 3D reconstruction in order to reduce the number of faces of the output polyhedron if it is known in advance that the input shape is a convex polyhedron with a small number of faces.

We end by describing briefly the results of some Monte Carlo simulations involving reconstructions of the ellipsoid depicted in Fig. 16. Fig. 19 shows the average, maximum, and minimum L_2 errors in a run of 100 reconstructions with numbers of measurements ranging from 7 to 49 in steps of 7, with a fixed noise level of $\sigma = 0.05$, for both the least squares and linear program versions of the new algorithm. (As before, the two versions are called “new (LS)” and “new (LP)” in the figures.) Results for the Hausdorff errors were similar. In Fig. 19 and perhaps more clearly in Fig. 20 the same deterioration in the performance of the linear program version as was reported for 2D reconstructions can be seen. The average time for reconstructions is illustrated in Fig. 21, which demonstrates again the dramatic difference in speed between the two versions of our algorithm. It should be noted that we have noticed considerable variation in reconstruction times.

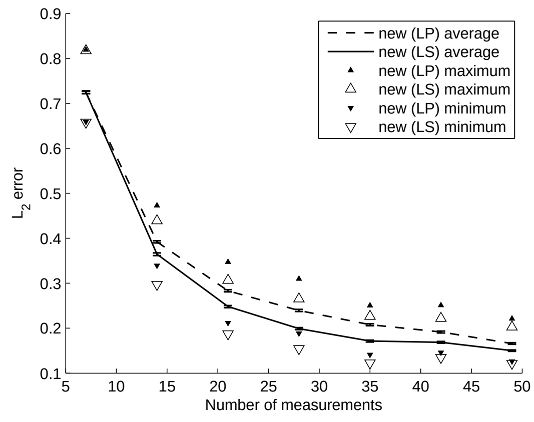


Fig. 19. L_2 error against number of measurements

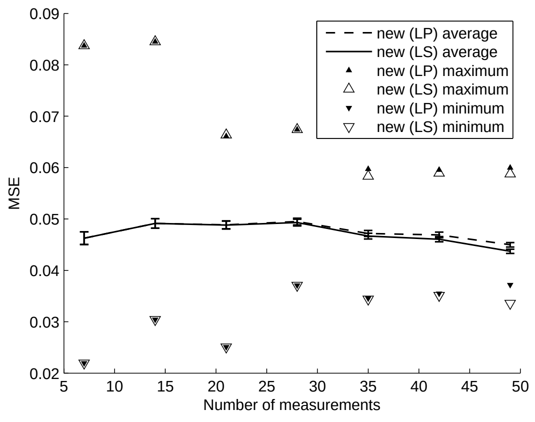


Fig. 20. MSE against number of measurements

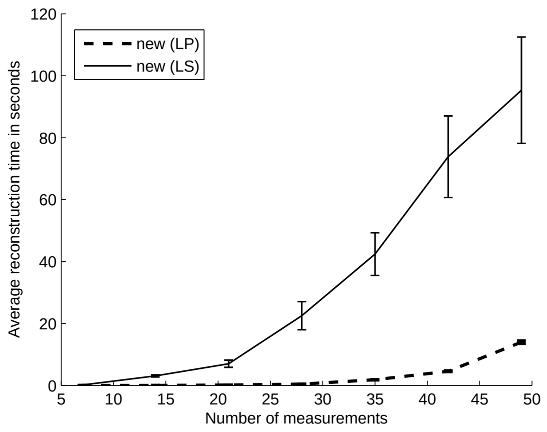


Fig. 21. Average reconstruction time against number of measurements

A further Monte Carlo run of 100 reconstructions, with the same ellipsoid, noise levels ranging from 0 to 0.5 in steps of 0.05, and a fixed number of 30 measurements, yielded Fig. 22 for the L_2 errors, and very similar graphs for the MSE and Hausdorff errors. At this relatively low number of measurements it can be seen that the linear program version competes quite well with the least squares version up to a noise level of perhaps $\sigma = 0.15$ or so.

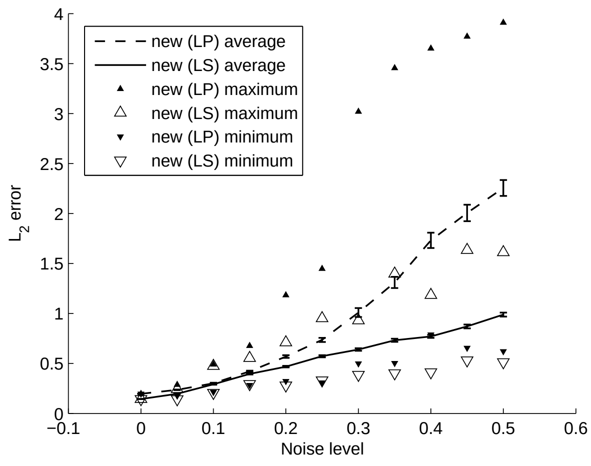


Fig. 22. L_2 error against noise level

IX. Previous work on 3D reconstruction

To the best of our knowledge, the only previous program for 3D reconstruction from support function measurements is that of Gregor and Rannou [4], who used it as a "focus of attention" pre-processing scheme for projection magnetic resonance imaging. One experiment involving 1,645 measurements is described in detail. Certain features of the specially chosen measurement directions were used to reduce the huge computational burden of applying the local consistency tests of [10] to a manageable level (see Section I). For example, the set of measurement directions enjoyed some symmetry that was used to advantage, and it was also required to be sufficiently uniformly distributed.

It seems that at present a direct comparison between our program and that of [4] is not viable. There are several reasons for this. Firstly, since the support function reconstruction in [4] is only a pre-processing scheme, there is no description there, either visual or in terms of computation of errors, of the quality of the support function reconstruction per se. Secondly, according to Gregor and Rannou (private communication), their support function reconstruction program is embedded inside the larger one for projection magnetic resonance imaging and it would take some work to extract it. Moreover, in its present form the program described in [4] works only with sets of measurement directions that are sufficiently uniformly distributed.

X. The new algorithm for "focus of attention"

In certain applications, such as those in which support function reconstruction is used in a "focus of attention" scheme as part of a larger algorithm, it is important that the output should contain the unknown shape. An example of such an application was described in the previous section.

Our new algorithm is easily modified for such use. In fact, all that is necessary is to choose the alternative output (11) defined in Section IV. As was pointed out there, this output is precisely the same as that of the Prince-Willsky algorithm. In 2D and when the support function measurements are exact ($\sigma = 0$), this corresponds to the bold polygon in Fig. 3.

For exact measurements in 3D, the output $\hat{P}_k$ given by (11) is a convex polyhedron that contains the unknown shape and has each of its facets orthogonal to one of the measurement directions. It can be constructed as follows. If $\hat{x}_1, \dots, \hat{x}_k$ is a solution of (LLS2) and F_i is a facet of $\hat{P}_k$ orthogonal to the measurement direction u_i, then $\hat{h}_i = \hat{x}_i^T u_i$ is the distance from the plane $H(u_i)$ containing F_i to the origin. Since $H(u_i)$ is also orthogonal to u_i, the equation of $H(u_i)$ can be found. In this way, we obtain a description of $\hat{P}_k$ in terms of its bounding planes $H(u_i)$, $1 \leq i \leq k$—its so-called $\mathcal{H}$-representation. If a picture of $\hat{P}_k$ is required, it is necessary to convert the $\mathcal{H}$-representation of $\hat{P}_k$ to a $\mathcal{V}$-representation, i.e., a list of its vertices, and then find the convex hull of these vertices. All this can be done very efficiently with readily available software.

XI. Conclusion

We have introduced a new algorithm for reconstructing an unknown shape from a finite number of noisy measurements of its support function. The algorithm, based on a least squares procedure, is very easy to program in standard software such as Matlab. Under mild conditions, theory guarantees that outputs of the algorithm will converge to the input shape as the number of measurements increases.

Experimental results in 2D have been described in detail in order to explain the main features of the new algorithm in relation to the primary known algorithm for 2D reconstruction, the Prince-Willsky algorithm. Reconstructions are of similar quality, but, as expected, the Prince-Willsky algorithm is generally preferable since it is considerably faster.

Our experimental results in 3D, however, show that the new algorithm yields good 3D reconstructions on an ordinary PC, without restriction on the sets of measurement directions (provided the number is not too large) and without any pre- or post-processing steps. As far as we know, no previous algorithm has accomplished this. In addition we have described a linear program version of the new algorithm that is much faster and better, or at least comparable, in performance at low levels of noise and reasonably small numbers of measurements.

The new algorithm should also readily find application, for example as a "focus-of-attention" pre-processing scheme in one or more of the scenarios mentioned in Section I.

Acknowledgment

The authors would like to thank a team of Western Washington University undergraduate students—Mark Lockwood, LeRoy Miller, Greg Richardson, Holly Rutledge, and Michael Taron—for writing the programs.

References

[1] J. L. Prince and A. S. Willsky, “Estimating convex sets from noisy support line measurements,” *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 12, no. 4, pp. 377–389, 1990.

[2] A. S. Lele, S. R. Kulkarni, and A. S. Willsky, “Convex polygon estimation from support line measurements and applications to target reconstruction from laser radar data,” *J. Optical Soc. Amer. A*, vol. 9, pp. 1693–1714, 1992.

[3] J. Gregor and D. Huff, “A focus of attention preprocessing scheme for EM-ML PET reconstruction,” *IEEE Trans. Medical Imaging*, vol. 16, pp. 218–233, 1997.

[4] J. Gregor and F. R. Rannou, “Three-dimensional support function estimation and application for projection magnetic resonance imaging,” *Int. J. Imaging Syst. Technology*, vol. 12, pp. 43–50, 2002.

[5] J. Gregor, S. Gleason, M. Paulus, and J. Cates, “Fast Feldkamp reconstruction based on focus of attention and distributed computing,” *Int. J. Imaging Syst. Technology*, vol. 12, pp. 229–234, 2002.

[6] M. Ikehata and T. Ohe, “A numerical method for finding the convex hull of polygonal cavities using the enclosure method,” *Inverse Problems*, vol. 18, pp. 111–124, 2002.

[7] R. Schneider, *Convex Bodies: The Brunn-Minkowski Theory.* Cambridge: Cambridge University Press, 1993.

[8] R. J. Gardner, *Geometric Tomography*, 2nd ed. New York: Cambridge University Press, 2006.

[9] J. Serra, *Image Analysis and Mathematical Morphology.* New York: Academic Press, 1982.

[10] W. C. Karl, S. R. Kulkarni, G. V. Verghese, and A. S. Willsky, “Local tests for consistency of support hyperplane data,” *J. Math. Imaging and Vision*, vol. 6, pp. 249–269, 1995.

[11] P. C. Gaston and R. Lozano-Perez, “Tactile recognition and localization using model objects: The case of polyhedra on the plane,” *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 6, no. 3, pp. 257–266, 1984.

[12] J. L. Schneiter and T. B. Sheridan, “An automated tactile sensing strategy for planar object recognition and localization,” *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 12, no. 8, pp. 775–786, 1990.

[13] P. K. Ghosh and K. V. Kumar, “Support function representation of convex bodies, its application in geometric computing, and some related representations,” *Comput. Vision and Image Understanding*, vol. 72, no. 3, pp. 379–403, 1998.

[14] N. I. Fisher, P. Hall, B. Turlach, and G. S. Watson, “On the estimation of a convex set from noisy data on its support function,” *J. Amer. Statist. Assoc.*, vol. 92, no. 937, pp. 84–91, 1997.

[15] P. Hall and B. Turlach, “On the estimation of a convex set with corners,” *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 21, no. 3, pp. 225–234, 1999.

[16] E. Mammen, J. S. Marron, B. A. Turlach, and M. P. Wand, “A general projection framework for constrained smoothing,” *Statist. Sci.*, vol. 16, no. 3, pp. 232–248, 2001.

[17] H. Groemer, *Geometric Applications of Fourier Series and Spherical Harmonics.* New York: Cambridge University Press, 1996.

[18] A. Poonawala, P. Milanfar, and R. J. Gardner, “Shape estimation from support and diameter functions,” *J. Math. Imaging Vision*, vol. 24, pp. 229–244, 2006.

[19] R. J. Gardner, M. Kiderlen, and P. Milanfar, “Convergence of algorithms for reconstructing convex bodies and directional measures,” *Ann. Statist.*, vol. 34, pp. 1331–1374, 2006.