An Exploration of Semidefinite Programming Applied to Rigid Motion Factorization

Kevin Novins[1] and James Arvo[2]

[1]University of Auckland, novins@cs.auckland.ac.nz, +64 9 373-7599

[2]University of California, Irvine, USA, arvo@uci.edu, +1 (626) 577-9648

Abstract

We re-examine the problem of recovering the shape of a rigid body from a sequence of images. A classic solution proposed by Tomasi and Kanade in 1992 employs a factorization of the noisy data into a product of two rank-three matrices, one encoding camera coordinates and the other encoding scene coordinates. In a second phase an optimization process is employed to enforce orthogonality constraints on the camera coordinate system. In this paper we focus on this second phase and observe that the powerful constrained optimization technique known as *semidefinite programming* (SDP) can be used to impose the necessary constraints. We describe our initial experiments in applying this technique and suggest future directions, including the use of different norms and the potential for subsuming the first phase of rank-three factorization into the SDP optimization as well.

Keywords: Semidefinite programming, rigid body motion, shape recovery

1 Introduction

One of the perennial problems of computer vision is the recovery of three-dimensional coordinates from two-dimensional images. This problem has been attacked using a variety of information, including shading, focus, and structured light. A wide assortment of solutions have been proposed. In this paper we address a restricted form of this problem in which the coordinates of a collection of n labeled fixed points in space are to be recovered from a stream of 2D projections, or images, each containing all n points. Tomasi and Kanade introduced a factorization method for solving this problem in 1992 [1].

Meanwhile, in the field of mathematical programming, a powerful new tool appeared in the 1990's called *Semidefinite Programming* (SDP) [2]. Semidefinite programming is a generalization of linear and quadratic programming that solves convex optimization problems. While SDP has been applied to a wide variety of problems, including several related to pattern recognition, it is still largely unknown in the computer vision community. In this paper, we explore the use of SDP to help solve Tomasi-Kanade reconstruction problems.

It is often a challenge to express an optimization problem in the language of SDP. The constraints must be put in terms of the non-negativity of the eigenvalues of a matrix; that is, the matrix is constrained to be positive semidefinite. Moreover, the approach is only applicable to *convex problems*; those for which any convex combination of feasible solutions is also a feasible solution. Often it is necessary to recast a non-convex problem into one that is "near" the original in some sense in order employ SDP. For instance, we might change the norm, or select a different but similar objective function that is convex. In our current approach, we have chosen a norm that is readily expressed in the context of SDP, yet different from norms previously reported.

Our approach does not yet exhibit any consistent advantage over existing techniques. While our results are very similar to those produced by previous techniques in most cases, it is less reliable for very ill-posed problem instances; that is, instances with extreme amounts of noise, or an extremely limited range of views. There are potential advantages, however: SDP is very rich as it is 1) capable of encompassing many types of constraints simultaneously within a single optimization process, and 2) capable of optimizing with respect to several different norms, two of which we explore in this paper. These properties allow for subtle tradeoffs to be made between various types of approximation error, and for error to be bounded in whichever way is deemed appropriate for a given application.

In the following sections we review the problem of recovering scene coordinates from a sequence of images, we then introduce the idea of semidefinite programming and recast part of the problem as a semidefinite program, and then describe our experiments. We also suggest promising future directions.

2 Previous Work

In a classic paper, Tomasi and Kanade described a factorization method for extracting rigid shape from a set of orthographic projections of 3D points [1].

Without loss of generality, we consider a constrained problem in which the orthographic projections are always centered on a fixed point in space. Generalization to arbitrary weak perspective cameras is well understood and does not radically change the underlying math [3].

Projection data from the n points over k frames can be organized as a single $2k\times n$ matrix, which we denote by $P_{2k\times n}$. Each column of P contains the acquired 2D image coordinates for a single point tracked over all k frames, and each row contains one image coordinate, either x or y, for all 3D points in a given frame. The rows of P may be grouped into two $k \times n$ blocks corresponding to x and y coordinates, or into k smaller $2 \times n$ blocks, each containing both the x and y coordinates of all points in a single frame. We adopt the former notation in this paper, although both partitionings have appeared in the literature. Tomasi and Kanade's key contribution was to exploit the intrinsic structure of P that arises from the process of projecting fixed scene coordinates onto multiple image planes. In particular, in the absence of measurement error, the rank of P will be exactly three, which implies that it may be expressed as the product

$$P \;=\; R_{2k\times 3} S_{3\times n}, \tag{1}$$

where S contains the original 3D point set and R contains the rotation-projection parameters.

Adopting the organization of P noted earlier, both P and R may be expressed in terms of two smaller blocks that separate the two orthogonal components of the image coordinates. That is,

$$\begin{bmatrix} X \\ Y \end{bmatrix} = \begin{bmatrix} U \\ V \end{bmatrix} \begin{bmatrix} S \end{bmatrix} \tag{2}$$

where X and Y are both $k \times n$ matrices, U and V are both $k \times 3$ matrices, and S is a $3 \times n$ matrix. If we now define a sequence of 2×3 matrices

$$\Pi_i \;=\; \begin{bmatrix} u_i \\ v_i \end{bmatrix}, \tag{3}$$

for $i = 1, 2, \ldots, k$, where u_i and v_i are the i^{th} rows of U and V, respectively, then we may express a necessary property of the R matrix as

$$\Pi_i \, \Pi_i^{\mathrm{T}} \;=\; I_{2\times 2}, \tag{4}$$

for $i = 1, 2, \ldots, k$. That is, the corresponding rows of U and V are orthonormal. This is an immediate consequence of the fact that the screen coordinates of the points in each frame are expressed in terms of an orthogonal camera coordinate system.

We therefore may attempt to uniquely recover the original 3D point set, S, up to an orthonormal transformation, by factoring P into R times S in such a way that the constraints in Equation (4) are met. Tomasi and Kanade show that such a factorization can be obtained through a sequence of optimization problems. First, a rank-3 thin SVD is applied to P to generate

$$\hat{P} \;=\; \hat{R}_{2k\times 3}\, \hat{S}_{3\times n} \;\approx\; P, \tag{5}$$

where $\hat{P}$ is the best approximation to P, with respect to the 2-norm, that meets the rank constraints imposed by the process of orthogonal projection. In general $P \neq \hat{P}$ due to measurement error such as lens distortion and imprecise calibration, and also due to simplifying assumptions, such as orthographic projection. More importantly, the factorization in equation (5) is not unique, since

$$\hat{P} \;=\; (\hat{R}Q)(Q^{-1}S), \tag{6}$$

is also a factorization of the desired form for any non-singular 3×3 matrix Q. Thus, the decomposition in Equation (5) will generally fail to satisfy the constraints of Equation (4), which ensure that the matrix $\hat{R}$ can be interpreted as a collection of camera coordinate systems. Consequently, $\hat{S}$ will not correspond to the desired coordinates.

We must therefore invoke a second minimization process by which we find a Q that imposes the necessary structure on $\hat{R}$. Since the structural constraints are expressed in terms of $\Pi_i \, \Pi_i^{\mathrm{T}}$ in Equation (4), the optimum Q is not unique: Any orthonormal matrix applied to an optimal Q will yield another optimal Q. Consequently, the coordinates in S are specified only up to rotations and/or reflections.

Tomasi and Kanade state that solving for Q is a simple non-linear optimization that can be solved "efficiently and reliably," but do not suggest a particular solution method [1]. Hajder later suggested bundling the constraints into an objective function that can be minimized using Levenberg-Marquadt [4].

Taking a different approach, Brand [3] suggested converting the non-linear problem in to a linear one by solving for the entries of the matrix $H = QQ^T$ rather than those of Q in Equation (6). In particular, Brand's approach consists in solving for the matrix H that imposes the orthonormality constraints on the $\hat{\Pi}_i$ matrices, via

$$\hat{\Pi}_i \, H \, \hat{\Pi}_i^{\mathrm{T}} \; = \; I, \tag{7}$$

for $i = 1, 2, \ldots, k$, and then setting

$$Q \; = \; AD^{\frac{1}{2}}, \tag{8}$$

where ADA^{T} is the eigen-decomposition of H. Unfortunately, if any of the eigenvalues of H are negative, the Q matrix would necessarily have complex entries, thereby violating a fundamental physical constraint on the factorization process. In this case, Brand suggests employing a tertiary optimization to glean a real-valued Q from H using a fixpoint iteration [3].

Brand's method is appealing because it replaces a complicated optimization with one, or possibly two, simpler optimizations. Unfortunately the two optimizations are not coupled. We propose an alternative solution technique to the second optimization phase of the Tomasi-Kanade algorithm that uses semidefinite programming to ensure that positive definiteness of H is maintained at the same time that the entries of H are optimized.

3 Semidefinite Programming

Semidefinite programming (SDP) is a constrained optimization technique for convex problems that subsumes both linear programming and quadratic programming. Practical and efficient solvers are available. A good overview of the technique is provided by Vandenberghe and Boyd [2], including how linear and quadratic programming can be rephrased in terms of semidefinite programming. This optimization technique has been applied in pattern recognition by Keuchel [5] and Weinberger [6], and to principal component analysis (PCA) and singular value decomposition (SVD) by Aspremont. [7].

There are several common forms for SDP problems. The form we apply can be expressed as

$$\begin{array}{ll} \text{minimize} & c^T x \\ \text{subject to} & F(x) \succeq 0, \end{array} \tag{9}$$

where $c, x \in \mathrm{I\!R}^n$, and

$$F(x) \; = \; F_0 + x_1 F_1 + \cdots + x_n F_n, \tag{10}$$

where each $F_0, \ldots, F_n$ is a fixed square matrix of the same size. The relation "$\succeq$" indicates a bound on the *eigenvalues* of a matrix; thus, $M \succeq 0$ means that the matrix M is positive semidefinite. It is precisely the form of this latter constraint that characterizes semidefinite programming.

Adopting the notation that $\hat{u}_i$ and $\hat{v}_i$ are the i^{th} rows of $\hat{U}$ and $\hat{V}$, respectively, which are blocks of $\hat{R}$, we wish to create an objective function that has the effect of minimizing

$$\left|\left| \, \hat{\Pi}_i \, H \, \hat{\Pi}_i^{\mathrm{T}} - I \, \right|\right| \tag{11}$$

for $i = 1, \ldots, k$. In order to fit the problem into the SDP framework, we chose to create an objective function that has the effect of minimizing the maximum absolute difference between these matrices; that is, we wish to solve

$$\min_{H} \max_{i=1}^{k} \left\{ \begin{array}{c} |\hat{u}_i H \hat{u}_i^{\mathrm{T}} - 1| \\ |\hat{v}_i H \hat{v}_i^{\mathrm{T}} - 1| \\ |\hat{u}_i H \hat{v}_i^{\mathrm{T}}| \end{array} \right\}, \tag{12}$$

subject to the constraints that H be symmetric and positive semidefinite. Observe that in Equation (12) the first two terms in brackets impose the constraint that all rows have unit length, while the last term imposes the constraint that corresponding rows of $\hat{U}$ and $\hat{V}$ are orthogonal. To put this objective function into a standard form for SDP, it is necessary to move the complexity into the constraints by introducing one or more auxillary variables [2], also commonly called a *slack variables.* We introduce one such variable, t, which allows us to express the desired optimization problem as

$$\begin{array}{lll} \multicolumn{3}{l}{\text{Minimize } t} \\ \multicolumn{3}{l}{\text{Subject to}} \\ \quad |\, \hat{u}_i H \hat{u}_i^{\mathrm{T}} - 1 \,| & < & t \\ \quad |\, \hat{v}_i H \hat{v}_i^{\mathrm{T}} - 1 \,| & < & t \\ \quad |\, \hat{u}_i H \hat{v}_i^{\mathrm{T}} \,| & < & t \\ \quad H & = & H^{\mathrm{T}} \\ \quad H & \succeq & 0. \end{array} \tag{13}$$

It is now straightforward to recast this problem in the language of semidefinite programming. We shall have seven variables, t, and $x_1, \ldots, x_6$, where the latter six encode the distinct elements of the symmetric matrix H; that is,

$$H \; = \; \begin{bmatrix} x_1 & x_4 & x_6 \\ x_4 & x_2 & x_5 \\ x_6 & x_5 & x_3 \end{bmatrix}. \tag{14}$$

Letting

$$x = \begin{bmatrix} t & x_1 & x_2 & x_3 & x_4 & x_5 & x_6 \end{bmatrix}^{\mathrm{T}}, \tag{15}$$

we construct a collection of $k \times k$ diagonal matrices, A^-, A^+, etc., each of which depends on the vector of unknowns, x, so that

$$\begin{array}{rcl} A^- \succeq 0 & \Longleftrightarrow & \hat{u}_i H \hat{u}_i^{\mathrm{T}} - 1 + t \geq 0 \;\; \forall i \\ A^+ \succeq 0 & \Longleftrightarrow & -\hat{u}_i H \hat{u}_i^{\mathrm{T}} + 1 + t \geq 0 \;\; \forall i \\ B^- \succeq 0 & \Longleftrightarrow & \hat{v}_i H \hat{v}_i^{\mathrm{T}} - 1 + t \geq 0 \;\; \forall i \end{array}$$

$$
\begin{aligned}
B^+ \succeq 0 &\iff -\hat{v}_i H \hat{v}_i^{\mathrm{T}} + 1 + t \geq 0 \;\; \forall i \\
C^- \succeq 0 &\iff \hat{u}_i H \hat{v}_i^{\mathrm{T}} + t \geq 0 \;\; \forall i \\
C^+ \succeq 0 &\iff -\hat{u}_i H \hat{v}_i^{\mathrm{T}} + t \geq 0 \;\; \forall i
\end{aligned}
$$

All of the above matrices are diagonal, with the constraints on the right as the entries. Observe that A^+ and A^- together establish the constraint on the first absolute value appearing in Equation (12); similarly, B^+ and B^- establish the second term, and C^+ and C^- establish the third. Given the six $k \times k$ matrices above, we can express the optimization problem of Equation (12) in the form shown in Equation (9) by defining

$$c = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}^{\mathrm{T}}, \tag{16}$$

and

$$F(x) = \begin{bmatrix} A^- & & & & & & \\ & A^+ & & & & & \\ & & B^- & & & & \\ & & & B^+ & & & \\ & & & & C^- & & \\ & & & & & C^+ & \\ & & & & & & H \end{bmatrix} \tag{17}$$

All of these blocks are diagonal except for H. Diagonal blocks amount to ordinary positivity constraints in the system; that is, if D is diagonal, then $D \succeq 0$ is equivalent to $D \geq 0$, where the latter is interpreted in the usual sense of element-by-element non-negativity. Semidefinite packages typically allow simplified constraints such as these to be specified separately, thus improving efficiency of the solution process in both space and time.

Each of the diagonal elements consists of a linear expression in the unknowns. For example, if we let $u_i = [\alpha_i, \beta_i, \gamma_i]$, then the diagonal entries of the matrix A^- are given by

$$
\begin{aligned}
a_{ii}^- \;=\; & \alpha_i^2 x_1 + \beta_i^2 x_2 + \gamma_i^2 x_3 + \\
& 2\alpha_i\beta_i x_4 + 2\beta_i\gamma_i x_5 + 2\alpha_i\gamma_i x_6 - \\
& 1 + t.
\end{aligned} \tag{18}
$$

The expression for a_{ii}^+ is obtained by negating all terms on the right of Equation (18) except for t. The expressions for b_{ii}^- and b_{ii}^+ are obtained by using v_i instead of u_i. Similar expressions hold for c_{ii}^- and c_{ii}^+.

The final step in formulating the optimization problem is to construct the sequence of constant matrices $F_0, F_1, \ldots, F_7$ that are linearly combined to form the matrix $F(x)$, according to Equation (10). The matrix F_0 consists of all the constants, the matrix F_1 consists of all the coefficients of the variable x_1, and so on for all F_i and x_i.

4 Results

We have implemented the semidefinite program from the previous section using Matlab[1] and the publicly available SeDuMi toolbox[2] [8], and we have compared it to our own implementation of Brand's algorithm [3] using Matlab.

We used a small synthetic test scene for our experiments, consisting of points on the eight corners of a unit cube. The cube was rotated and projected according to random constrained rotation matrices [9], and the projected points were perturbed by random noise.

As mentioned in Section 2, factorization methods only specify the original 3D point set, S, up to an orthonormal transformation, making it challenging to compare reconstructions. One way to address this is to rotate and reflect all reconstructions to a common coordinate frame [1]. However this approach requires an additional minimization step that may distort the relative error measurements. Instead, we take advantage of the known point labeling to define a distance matrix, D, such that D_{ij} is the Euclidean distance between point i and point j in the model. As distance matrices are invariant under rotation and reflection, we may then measure reconstruction error as

$$\sum_{ij} (D_{ij} - D_{ij}^0)^2, \tag{19}$$

where D_{ij}^0 is the distance matrix of the ground truth model.

We found that for well-conditioned problems the semidefinite program produces results comparable to Brand's method. In any particular trial, either method could be more accurate than the other. However, on average, Brand's method was more accurate by about 10%, with a lower standard deviation. When the amount of noise was increased to extremely high levels (on the order of the distance between model vertices), or the angular coverage of the rotation matrices was restricted to an extremely small range (on the order of one degree), the problem becomes ill-conditioned. Brand's algorithm performs significantly better in such cases.

This difference in reconstruction error for small to moderate amounts of noise is illustrated by the scatter plot Figure 1a. The major difference between the two methods is in the choice of norm during the optimization. We minimize the maximum absolute deviation, which corresponds to the ∞-norm, whereas Brand, by employing linear least squares, minimizes the 2-norm. In Figure 1b, we

[1] Version 7.0, http://www.mathworks.com
[2] Version 1.05, http://sedumi.mcmaster.ca/

observe that SDP performs as expected in that it always finds a lower maximum absolute deviation from the norm/orthogonality constraints than Brand's algorithm. The opposite is true when considering the 2-norm, as shown in Figure 1c.

5 Discussion

We have observed that SeDuMi tends to find solutions where several terms of the objective function share the same maximum absolute deviation. In order to force more terms to be as small as possible, we also tried rephrasing the semidefinite program to minimize the sum of absolute deviations across all terms of the objective function, which is equivalent to the 1-norm. This can be done by introducing more slack variables, $t_1, t_2, \ldots, t_{3F}$, constraining each term individually. Unfortunately, this does not generally improve the results, as the system then then becomes more lax in controlling the maximum absolute deviation. We conclude that the 2-norm is more appropriate for this problem, and are in the process of forumlating this norm in the context of SDP using Shur complements [2], as shown in Appendix A.

Semidefinite programming is an extremely general and powerful tool, and it is thus more complex and more computationally expensive than the methods used by Brand [3] and Hajder [4] to enforce the orthogonality constraints. It may never be the method of choice for finding the few entries in the H matrix. However a problem affecting all algorithms that descend from Tomasi and Kanade's work is that each of them applies the rank constraints first, leaving only six degrees of freedom (in the H matrix) for orthonormalization. Ideally, both the rank and orthogonality constraints would be applied in concert so that appropriate tradeoffs can be made between the rank-reduction phase and the orthogonalization phase. Thus, we are currently exploring the use of semidefinite programming to solve the entire optimization problem, not just for finding the best H matrix.

6 Conclusion

We have demonstrated that semidefinite programming can be applied to one phase of the problem of recovering scene and camera coordinates from a sequence of images. Our formulation attempts to impose normality, orthogonality, and positive semidefiniteness constraints on a low-rank factorization obtained via singular value decomposition. Our results are comparable to those obtained by Brand in typical situations. While our current formulation is not practical, we believe that this approach has promise in that it is an extremely flexible framework. It can potentially subsume the entire problem, including the role played by SVD, into a single optimization process. We expect to see wider application of SDP to fundamental computer vision problems in the future.

References

[1] C. Tomasi and T. Kanade, "Shape and motion from image streams under orthography: a factorization method," *International Journal of Computer Vision*, vol. 9, no. 2, pp. 137–154, 1992.

[2] L. Vandenberghe and S. Boyd, "Semidefinite programming," *SIAM Review*, vol. 38, no. 1, pp. 49–95, 1996.

[3] M. Brand, "Morphable models from video," in *Proceedings of Computer Vision and Pattern Recognition (CVPR)*, vol. 2, (Kauai, Hawaii), pp. 456–463, Dec. 2001.

[4] L. Hajder, "Shape and motion from video," in *Proceedings of Grafika*, (Budapest), pp. 98–102, 2003.

[5] J. Keuchel, C. Schnörr, C. Schellewald, and D. Cremers, "Binary partitioning, perceptual grouping, and restoration with semidefinite programming," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 25, pp. 1364–1379, Nov. 2003.

[6] K. Q. Weinberger and L. K. Saul, "Unsupervised learning of image manifolds by semidefinite programming," in *Proceedings of Computer Vision and Pattern Recognition (CVPR)*, vol. 2, (Washington, DC), pp. 998–995, July 2004.

[7] A. d'Aspremont, L. El Ghaoui, M. I. Jordan, and G. R. G. Lanckriet, "A direct formulation for sparse pca using semidefinite programming," in *Advances in Neural Information Processing Systems (NIPS)*, Dec. 2004.

[8] J. Sturm, "Using SeDuMi 1.02, a MATLAB toolbox for optimization over symmetric cones," *Optimization Methods and Software*, pp. 625–653, 1999.

[9] J. Arvo, "Fast random rotation matrices," in *Graphics Gems III* (D. Kirk, ed.), pp. 117–120, Academic Press, 1992.

A Quadratic constraints

The optimization discussed in Section 3 can also be expressed in terms of quadratic constraints, including the 2-norm, by means of the following well-known relationship involving the blocks of a square 2×2 block matrix. Specifically,

$$\begin{bmatrix} A & B \\ B^{\mathrm{T}} & D \end{bmatrix} \succeq 0, \tag{20}$$

if and only if

$$A \succeq 0 \quad \wedge \quad (D - B^{\mathrm{T}} A^{-1} B) \succeq 0, \tag{21}$$

where the matrix on the right is known as the Schur complement of D [2]. We can use this fact to solve the minimization problem

$$\min_{H} \max_{i=1}^{k} \left\{ \begin{array}{c} (\hat{u}_i H \hat{u}_i^{\mathrm{T}} - 1)^2 \\ (\hat{v}_i H \hat{v}_i^{\mathrm{T}} - 1)^2 \\ (\hat{u}_i H \hat{v}_i^{\mathrm{T}})^2 \end{array} \right\} \tag{22}$$

using SDP; here we shall show how to handle the top constraint only, as the others are handled similarly. First, define the matrix

$$\begin{bmatrix} I & B \\ B^{\mathrm{T}} & tI \end{bmatrix}, \tag{23}$$

where $t > 0$ and B is diagonal with its ith diagonal element being

$$\hat{u}_i H \hat{u}_i^{\mathrm{T}} - 1. \tag{24}$$

Since $tI \succeq 0$, it follows from Equation (21) that if the matrix in Equation (23) is positive semidefinite then the Schur complement is positive semidefinite as well. That is,

$$tI - B^{\mathrm{T}} B \succeq 0. \tag{25}$$

Since the matrix above is diagonal, the semidefinite constraint is equivalent to a simple positivity constraint, which results in

$$B^{\mathrm{T}} B \leq tI. \tag{26}$$

This constraint ensures that the square of each diagonal element is bounded above by t. The other two expressions in Equation (22) lead to two additional block matrices; all three of these matrices become blocks on the diagonal of a larger matrix that corresponds to Equation (17). Thus, the resulting matrix has far more off-diagonal elements than in the case of the 1-norm.

To obtain the 2-norm, we replace the single slack variable in Equation (23) with k distinct slack variables along the diagonal, then minimize the sum of all such variables. This is similar to the approach outlined in Section 5.

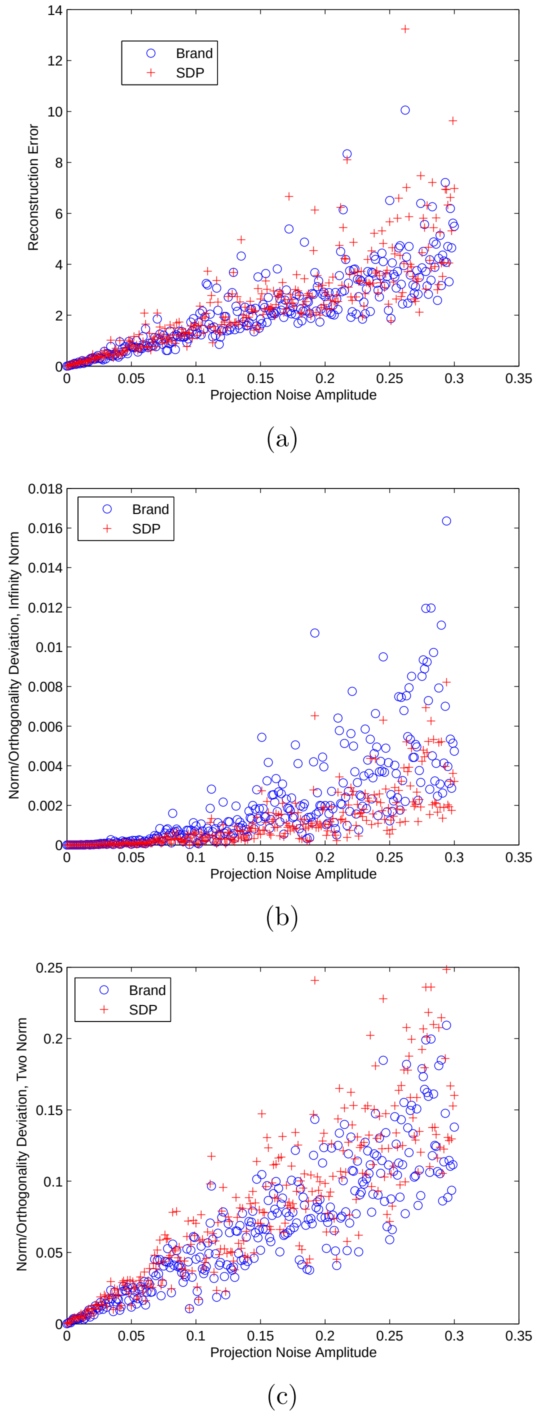

Figure 1: Comparing Brand's reconstruction algorithm [3] to the SDP algorithm outlined in Section 3. (a) Reconstruction error, according to the two norm of the difference of the distance matrices. (b) Deviation from the norm/orthogonality constraints according to the ∞-norm. (c) Deviation from the norm/orthogonality constraints according to the 2-norm.