Efficient Optimization for L_∞-problems using Pseudoconvexity

Carl Olsson
calle@maths.lth.se

Anders P. Eriksson
anderspe@maths.lth.se

Fredrik Kahl
fredrik@maths.lth.se

Centre for Mathematical Sciences
Lund University, Sweden

Abstract

In this paper we consider the problem of solving geometric reconstruction problems with the L_∞-norm. Previous work has shown that globally optimal solutions can be computed reliably for a series of such problems. The methods for computing the solutions have relied on the property of quasiconvexity. For quasiconvex problems, checking if there exists a solution below a certain objective value can be posed as a convex feasibility problem. To solve the L_∞-problem one typically employs a bisection algorithm, generating a sequence of convex problems. In this paper we present more efficient ways of computing the solutions.

We derive necessary and sufficient conditions for a global optimum. A key property is that of pseudoconvexity, which is a stronger condition than quasiconvexity. The results open up the possibility of using local optimization methods for more efficient computations. We present two such algorithms. The first one is an interior point method that uses the KKT conditions and the second one is similar to the bisection method in the sense it solves a sequence of SOCP problems. Results are presented and compared to the standard bisection algorithm on real data for various problems and scenarios with improved performance.

1. Introduction

Many geometrical computer vision problems may be formulated as optimization problems. For example, solving a multiview triangulation problem can be done by minimizing the L_2-reprojection error. In general this is a hard non-convex problem if the number of views is more than two [5]. Since closed form solutions are only avaliable in special cases, optimization problems are often solved using iterative methods such as bundle adjustment. The success of this type of local methods rely on good initialization methods. However, the initialization techniques frequently used optimize some algebraic cost function which, on one hand, simplifies the problem, but, on the other hand, has no geometrical or statistical meaning. When significant measurement noise is present such estimates may be far from the global optimum.

Geometric L_∞-problem	References
– Multiview triangulation	[4, 6, 7, 3]
– Camera resectioning	[6, 7]
– Homography estimation	[6, 7]
– Structure and motion recovery with known camera orientation	[4, 6, 7]
– Reconstruction by using a reference plane	[6]
– Camera motion recovery	[10]
– Outlier detection	[11, 9]
– Reconstruction with covariance-based uncertainty	[10, 8]

Table 1. *List of different geometric reconstruction problems that can be solved globally with the L_∞-norm.*

To remedy this problem, the use of L_∞-optimization was introduced in [4]. It was shown that many geometric vision problems have a single local optimum if the L_2-norm is replaced by the L_∞-norm. This work has been extended in several directions and there is now a large class of problems that can be solved globally using L_∞-optimization, see Table 1. They have all been shown to have a single local optimum when using the L_∞-norm.

In [6, 7] it was shown that these problems are examples of quasiconvex optimization problems. A bisection-algorithm based on second order cone programs (SOCP) for solving this type of problems was also introduced. Let $f_i(x)$ be quasiconvex functions. The algorithm works by checking if there is an x satisfying $f_i(x) \leq \mu$ for all i for a fixed μ. A bisection is then performed on the parameter μ. Thus to solve the original problem we are led to solve a sequence of SOCPs. A particular problem when running the bisection algorithm is that it is not possible to specify a starting point for the SOCP. Even though a good solution might be available from a previously solved SOCP, this solution will in general not lie on the so called central-path.

This is however required for for good convergence of the interior-point-method used to solve the SOCP. Hence much would be gained if it was possible to let μ vary, and not have to restart the optimization each time μ is changed.

In [4] the μ was allowed to vary during the optimization, hence the problem was solved using a single program. Although this program is not convex, it was observed that this worked well for moderate scale problems. Still convergence to the global minimum was not proven. In [3], an alternative optimization technique based on interval analysis was proposed for solving the multiview triangulation problem. However, the optimization technique does not exploit the convexity properties of the problem and it may be inefficient. For the multiview triangulation problems reported in [3], the execution times are in the order of several seconds which is considerably slower than the bisection algorithm [6, 7].

In this paper we show that we are not limited to keeping μ fixed. We show that the functions involved in L_∞-problems are not just quasiconvex but actually pseudoconvex which is a stronger condition. This allows us to derive necessary and sufficient conditions for a global optimum, which opens up the possibility of using local optimization algorithms as the ones used in [4]. We show that these algorithms are more efficient than the bisection algorithm in terms of execution times. For large scale algorithms we propose an algorithm that is similar to the bisection algorithm in that it solves a sequence of SOCPs. However rather than fixing μ we will approximate the original program using a SOCP.

2. Theoretical background

In this section we formulate the L_∞-problem and briefly review some concepts from optimization which will be needed in Sections 3 and 4.

2.1. Problem formulation

The problems in geometric computer vision that we will be considering in this paper may be written in the following minimax form:

$$\min_{x} \quad \max_{i} \quad \frac{||\,[\,a_{i1}^T x + b_1, a_{i2}^T x + b_2\,]\,||_2}{a_{i3}^T x + b_3} \tag{1}$$

$$\text{s.t.} \quad a_{i3}^T x + b_3 > 0, \qquad i = 1, \ldots, m \tag{2}$$

where $a_{ij} \in \mathbb{R}^n$ and $b_j \in \mathbb{R}$ for $j = 1, 2, 3$. The dimension of the problem depends on the particular application, starting from $n = 3$ for the (basic) multiview triangulation problem.

If we consider the individual (residual) functions $\frac{||\,[\,a_{i1}^T x + b_1, a_{i2}^T x + b_2\,]\,||_2}{a_{i3}^T x + b_3}$, $i = 1, \ldots, m$, as the components of an m-vector, then the problem may be thought of as minimizing the L_∞-norm of this (residual) vector.

2.2. Various types of convexity

Next we recapitulate on some of the different types of convexity and their properties. This is well-known in the optimization literature, cf. [1]. When we discuss properties of different function classes we need to distinguish between the following three types of points.

Definition 2.1. $\bar{x}$ is a stationary point if $\nabla f(\bar{x}) = 0$.

Definition 2.2. $\bar{x}$ is a local minimum if there exists $\epsilon > 0$ such that $f(x) \geq f(\bar{x})$ for all x with $||x - \bar{x}|| \leq \epsilon$.

Definition 2.3. $\bar{x}$ is a strict local minimum if there exists $\epsilon > 0$ such that $f(x) > f(\bar{x})$ for all x with $||x - \bar{x}|| \leq \epsilon$.

For a differentiable function we always have that *strict local minimum* $\Rightarrow$ *local minimum* $\Rightarrow \nabla f(\bar{x}) = 0$. The reversed implications are, however, not true in general.

Definition 2.4. A function f is called quasiconvex if its sublevel sets $S_\mu(f) = \{\, x; f(x) \leq \mu \,\}$ are convex.

In [6, 7] it was shown that the objective functions in a variety of structure and motion problems are quasiconvex. Quasiconvex functions have the property that any strict local minimum is also a global minimum. They may however have several local minima and stationary points. Thus a local decent algorithm may not converge to the desired global optimum. Instead it is natural to use the property of convex sublevel sets as a basis for a bisection algorithm. For a fixed μ, finding a solution $\bar{x}$ such that $f(\bar{x}) \leq \mu$ can be turned into a convex feasibility problem. See [6, 7] for further details.

In [11] the notion of strict quasiconvexity was introduced.

Definition 2.5. A function f is called strictly quasiconvex if f is continuous quasiconvex and its sublevel sets $S_\mu(f) = \{\, x; f(x) \leq \mu \,\}$ fulfills the additional property

$$\bigcup_{\bar{\mu} < \mu} S_{\bar{\mu}}(f) = \operatorname{int} S_\mu(f). \tag{3}$$

Strictly quasiconvex functions have the property that any local minimum is a global minimum. They may however still have additional stationary points. As an example consider the function $f(x) = x^3$ on the set $-1 \leq x \leq 1$ (see Figure 1). The gradient vanishes at $x = 0$ but the global optimum is clearly in $x = -1$.

Note that there is also a class of functions that are referred to as strongly quasiconvex in the literature. To further complicate things the notions of strongly and strictly quasiconvex are sometimes interchanged.

One of the goals is to show that for all of the problems considered in Table 1, we are able to use local search algorithms rather than the bisection algorithm. In these types of

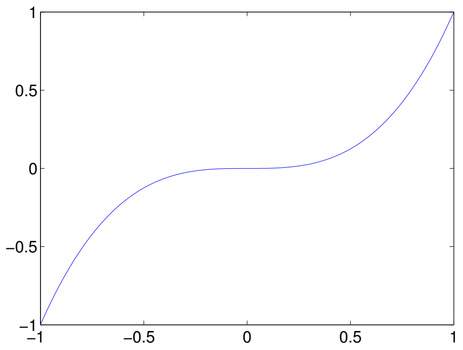

Figure 1. The function $f(x) = x^3, -1 \leq x \leq 1$, is strictly quasiconvex but still has a stationary point in $x = 0$.

algorithms, improving or descent directions are often determined using gradients and possibly Hessians. Therefore it is crucial that the gradients do not vanish anywhere except in the global optimum, that is, there can be no stationary points except for the global minimum. As we have seen this is not necessary true for quasiconvex (or even strictly quasiconvex) functions. Therefore we are led to study the following class of functions.

Definition 2.6. f is called pseudoconvex if f is differentiable and whenever $\nabla f(\bar{x})(x - \bar{x}) \geq 0$ we also have that $f(x) \geq f(\bar{x})$.

A pseudoconvex function is always quasiconvex (see [1]), but not necessarily the other way around. Pseudoconvex functions also have the following nice property:

Lemma 2.7. *Suppose f is pseudoconvex, then $\nabla f(\bar{x}) = 0$ if and only if $f(x) \geq f(\bar{x})$ for all x.*

Proof. If $\nabla f(\bar{x}) = 0$ then $\nabla f(\bar{x})(x - \bar{x}) = 0$ for all x and by definition $f(x) \geq f(\bar{x})$.
If $f(x) \geq f(\bar{x})$ for all x then x is a global (and hence local) minimum and therefore $\nabla f(x) = 0$. □

Thus for a pseudoconvex function any stationary point is a global minimum. This is a useful property since it ensures that the gradient does not vanish anywhere except in the optimum, making it possible to solve using, for instance, a steepest decent algorithm. For further details on various convexity issues, see e.g. [1, 2].

2.3. Constrained optimization

When minimizing an unconstrained differentiable function f, one is interested in solving the equations $\nabla f(x) = 0$. In the constrained case the corresponding equations are the KKT conditions (see [2, 1]). The KKT conditions play an important role in optimization. Many algorithms, such as interior point methods or sequential quadratic programming, are conceived as methods for solving the KKT conditions. Consider the constrained optimization problem

$$\begin{aligned} \min \quad & f(x) && (4) \\ \text{s.t.} \quad & f_i(x) \leq 0, \quad i = 1, ..., m. && (5) \end{aligned}$$

The KKT conditions for this problem are

$$\begin{aligned} \nabla f(x) + \sum_{j=1}^{m} \lambda_j \nabla f_j(x) &= 0 && (6) \\ \lambda_i f_i(x) &= 0 && (7) \\ f_i(x) &\leq 0 && (8) \\ \lambda_i &\geq 0, \quad i = 1, \ldots, m. && (9) \end{aligned}$$

In the general nonconvex case there may be many solutions to these equations. However in Section 3 we will show that only the global optimum solves them for the class of problems in Table 1.

3. Theoretical results

In this section we will derive our main results. We will first show that the objective function considered in (1) is in fact pseudoconvex. Then we proceed to derive necessary and sufficient conditions for global optima of a function that is a maximum of pseudoconvex functions.

Note that for a minimax problem of the form (1), one may equivalently consider the squared residual functions. Therefore, let

$$f(x) = \frac{(a_1^T x + b_1)^2 + (a_2^T x + b_2)^2}{(a_3^T x + b_3)^2}, \qquad (10)$$

where $a_j \in \mathbb{R}^n$ and $b_j \in \mathbb{R}$. Here $f(x)$ can be written as a quotient $\frac{w(x)}{v(x)}$ between the convex function

$$w(x) = \frac{(a_1^T x + b_1)^2 + (a_2^T x + b_2)^2}{a_3^T x + b_3} \qquad (11)$$

and the linear (and hence concave) function

$$v(x) = a_3^T x + b_3. \qquad (12)$$

on the set $\{ x \,|\, v(x) > 0 \}$. In the next lemma we show that any such function is pseudoconvex on this domain.

Lemma 3.8. *If $w : S \mapsto \mathbb{R}$ is convex and $v : S \mapsto \mathbb{R}$ is concave then $f(x) = \frac{w(x)}{v(x)}$ is pseudoconvex on $S = \{ x \,|\, v(x) > 0\}$.*

Proof. Since w is convex and w concave we have

$$\begin{aligned} w(x) - w(\bar{x}) &\geq \nabla w(\bar{x})(x - \bar{x}) && (13) \\ v(x) - v(\bar{x}) &\leq \nabla v(\bar{x})(x - \bar{x}). && (14) \end{aligned}$$

The gradient of f is given by

$$\nabla f(x) = \frac{1}{v(x)}\left(\nabla w(x) - \frac{w(x)}{v(x)}\nabla v(x)\right). \tag{15}$$

Setting

$$\nabla f(\bar{x})(x-\bar{x}) \geq 0 \tag{16}$$

and since $v(x) > 0$ we have

$$\left(\nabla w(\bar{x}) - \frac{w(\bar{x})}{v(\bar{x})}\nabla v(\bar{x})\right)(x-\bar{x}) \geq 0. \tag{17}$$

Inserting (13) and (14) yields

$$0 \leq w(x) - w(\bar{x}) - \frac{w(\bar{x})}{v(\bar{x})}(v(x) - v(\bar{x})) \tag{18}$$

$$\Leftrightarrow \frac{w(\bar{x})}{v(\bar{x})} \leq \frac{w(x)}{v(x)} \Leftrightarrow f(\bar{x}) \leq f(x). \tag{19}$$

□

Thus, from Definition 2.6, $f(x)$ is pseudoconvex on the set $\{\, x \,|\, a_3^T x + b_3 > 0 \,\}$. Now recall that we wish to minimize $f(x) = \max_i f_i(x)$, where each f_i is pseudoconvex. It does not make sense to say that pseudoconvexity is preserved under the max-operation, since the resulting function is in general not differentiable everywhere. However we are able to use pseudoconvexity to derive optimality conditions for the max-function.

Theorem 3.9. *x^* solves $\mu^* = \inf_{x\in S} f(x)$, where $S = \{\, x; v_i(x) > 0\ \forall i \,\}$, if and only if there exists λ_i^* such that*

$$\sum_{j=1}^{m} \lambda_j^* \nabla f_j(x^*) = 0 \tag{20}$$

where $\lambda_i^* \geq 0$ if $f_i(x^*) = \mu^*$ and $\lambda_i^* = 0$ if $f_i(x^*) < \mu^*$ for $i = 1,\dots,m$ and $\sum_j \lambda_j^* = 1$.

Proof. If $f_i(x^*) < \mu^*$ then there is a neighborhood such that $f_i(x) < \mu^*$ since f_i is continuous. Hence we may disregard the functions where $f_i(x^*) < \mu^*$ and assume that all $f_i(x^*) = \mu^*$.

First we show that if x^* is a local minimizer then (20) is fulfilled. If x^* is a local minimizer, then for all directions d there is an i such that $\nabla f_i(x^*)^T d \geq 0$, or equivalently the system $\nabla f_i(x^*)^T d < 0$ for all i has no solution. Let A be the matrix with rows $\nabla f_i(x^*)^T$. Then the system $\nabla f_i(x^*)^T d < 0$ for all i can be written

$$Ad < 0 \tag{21}$$

and the system (20) can be written

$$A^T\lambda^* = 0,\;\; \lambda^* \geq 0,\; \sum_j \lambda_j^* = 1. \tag{22}$$

By Gordan's theorem (which is a Farkas type theorem, see [1]) precisely one of these systems has a solution, and therefore (20) has a solution.

Next assume that there exists an $\bar{x}$ such that $f(x^*) > f(\bar{x})$. We will show that the system

$$\begin{aligned} \sum_j \lambda_j^* \nabla f_j(x^*) &= 0 \\ \sum_j \lambda_j^* &= 1 \\ \lambda_i^* &\geq 0 \quad \text{for } i = 1,\dots,m \end{aligned} \tag{23}$$

has no solution. Since f is quasiconvex (f_i are pseudoconvex and thereby quasiconvex) the direction $d = \bar{x} - x^*$ is a decent direction. Therefore $\nabla f_i(x^*)^T d \leq 0$ for all i. Now assume that $\nabla f_i(x^*)^T d = 0$ for some i. Then we have

$$f(\bar{x}) \geq f_i(\bar{x}) \geq f_i(x^*) = \mu^*, \tag{24}$$

since f_i is pseudoconvex, which contradicts $f(x^*) > f(\bar{x})$. Therefore we must have that $\nabla f_i(x^*)^T d < 0$ for all i. Now since all of the λ_i^* are nonnegative and sum to one, we have

$$d^T \sum_i \lambda_i \nabla f_i(x^*) < 0 \tag{25}$$

and therefore the system (23) has no solution. □

Note that pseudoconvexity is used in the second part of the theorem. In fact, for general functions these conditions are necessary but not sufficient for a global minimum, for the sufficiency part we require pseudoconvexity.

The interpretation of the optimality conditions is that if none of the gradients vanish, then in each direction d there is an i such that $\nabla f_i(x)^T d \geq 0$, that is in each direction at least one of the functions f_i does not decrease. This theorem shows that a steepest descent algorithm that follows the gradient (or subgradient where the function is not differentiable) would find the global minimum, since the gradients does not vanish anywhere except for the optimum. Such a method only uses first order derivatives, we would however like to employ higher order methods like interior point methods since these are in general more effective. Therefore we rewrite the problem as follows:

$$\begin{aligned} P_1 \;:\;\; & \min \;\; \mu \\ & \text{s.t.} \;\; f_i(x) - \mu^2 \leq 0 \\ & \qquad\; x \in S,\;\; \mu \geq 0, \end{aligned} \tag{26}$$

This gives us a constrained problem where all functions are twice differentiable. The KKT conditions for this problem are

$$\begin{pmatrix} 0 \\ 1 \end{pmatrix} + \sum_j \lambda_j \begin{pmatrix} \nabla f_j(x^*) \\ -2\mu^* \end{pmatrix} = 0 \quad (27)$$
$$f_i(x^*) - (\mu^*)^2 \leq 0 \quad (28)$$
$$\lambda_i \geq 0 \quad (29)$$
$$\lambda_i(f_i(x^*) - (\mu^*)^2) = 0. \quad (30)$$

Corollary 3.10. *The KKT conditions* (27)-(30) *are both necessary and sufficient for a global optimum in problem* (26).

Proof. The KKT conditions are always necessary and sufficient for stationary points. By condition (30) we know that λ_i is zero if $f_i(x^*) < \mu^*$. By Theorem 3.9 the system (23) has a solution if and only if x^* is a global optimum of $\min_{x \in S} f(x)$. We see that if $\mu^* > 0$ then (27) and (29) have a solution if and only if (23) has a solution. Since $\min_{x \in S} f(x)$ is equivalent to problem (26), it follows that (x^*, μ^*) is a global optimum for this problem. If $\mu = 0$ then the result is trivial since then $\nabla f_i(x) = 0$ for all i. □

To avoid working with functions containing quotients we rewrite our problem again. Let

$$g_i(x) = (a_{i1}^T x + b_{i1})^2 + (a_{i2}^T x + b_{i1})^2 \quad (31)$$
$$h_i(x) = (a_{i3}^T x + b_{i3})^2 \quad (32)$$
$$h_i(x, \mu) = \mu^2 h_i(x). \quad (33)$$

An equivalent problem is

$$\begin{aligned} P_2 \; : \quad & \min \quad \mu \\ & \text{s.t.} \quad g_i(x) - h_i(x,\mu) \leq 0 \\ & \qquad\; x \in S, \mu \geq 0. \end{aligned} \quad (34)$$

Note that for a fixed μ this is the SOCP used in the bisection algorithm where we have squared the cone conditions in order to be able to take derivatives. The KKT conditions for this problem are

$$\begin{pmatrix} 0 \\ 1 \end{pmatrix} + \sum_j \gamma_j \begin{pmatrix} \nabla g_j(x^*) - \nabla_x h_j(x^*, \mu^*) \\ -2\mu^* h_j(x^*) \end{pmatrix} = 0 \quad (35)$$
$$g_i(x^*) - h_i(x^*, \mu^*) \leq 0 \quad (36)$$
$$\gamma_i \geq 0 \quad (37)$$
$$\gamma_i(g_i(x^*) - h_i(x^*, \mu^*)) = 0. \quad (38)$$

Corollary 3.11. *The KKT conditions* (35)-(38) *are both necessary and sufficient for a global optimum in problem* (34).

Proof. We have

$$\nabla f_i(x) = \frac{1}{h_i(x)} \left(\nabla g_i(x) - \frac{g_i(x)}{h_i(x)} \nabla h_i(x) \right). \quad (39)$$

Let $\gamma_i = \frac{\lambda_i}{h_i(x)}$. We see that since $h_i(x)$ is positive for $x \in S$, it follows that (36) - (38) is equivalent to (28) - (30). For equation (35) we see that if γ_i is nonzero then $\mu^2 = \frac{g_i(x)}{h_i(x)}$ and therefore the two KKT systems are equivalent. □

4. Algorithms

We present two algorithms for solving our L_∞-problems. They are both based on interior-point methods which solve the KKT conditions of the system. The first one is a standard solver for nonconvex problems. Using this solver we may formulate the program such that μ is allowed two vary and thus solves the problem with one program. This is in contrast with the bisection method which solves a sequence of programs for fixed μ. A particular problem with the bisection algorithm is it is not possible to specify a starting point when using the existing interior-point-methods. This is because it may not lie on the so called central-path, which is resulting in slow convergence. Hence much could be gained by, letting μ vary, and not have toe restart the optimization procedure each time mu is changed.

In the second algorithm we use SeDuMi to solve a sequence of SOCP that approximates the original problem.

4.1. A Primal dual interior point algorithm for the pseudoconvex system.

In this section we briefly review the LOQO-algorithm [13] which is a state-of-the-art optimization algorithm that we will use for solving moderate scale geometric reconstruction problems. In fact, in [4] this algorithm was also tested. It was observed to work well, however convergence was not proved. Since quasiconvexity is not enough to prove convergence, to our knowledge this has not been pursued further. LOQO is an interior point algorithm for general nonconvex problems. As most interior point methods it searches for a solution to the KKT-conditions. For a general nonconvex problem the solution is not necessary the global minima since there may be more than one KKT-point, however in our case we know that there is only one. Recall that our problem is

$$\begin{aligned} P_2 \; : \quad & \min \quad \mu \\ & \text{s.t.} \quad g_i(x) - h_i(x,\mu) \leq 0 \\ & \qquad\; x \in S, \;\; \mu \geq 0. \end{aligned} \quad (40)$$

LOQO starts by adding slack variables $w_i \geq 0$ such that the inequality constraints are replaced by the equality constraints $g_i(x) - h_i(x, \mu) - w = 0$. The constraints $w \geq 0$ are eliminated by adding a logarithmic penalty term to the objective function.

$$\begin{aligned} P_{LOQO} \; : \min \quad & \mu + \nu \textstyle\sum_{j=1}^m \log(w_j) \\ \text{s.t.} \quad & g_i(x) - h_i(x,\mu) - w_i = 0. \end{aligned} \quad (41)$$

The first order conditions for this problem are

$$\begin{pmatrix} 0 \\ 1 \end{pmatrix} + \sum_j \gamma_j \begin{pmatrix} \nabla g_j(x^*) - \nabla_x h_j(x^*, \mu^*) \\ -2\mu^* h_j(x^*) \end{pmatrix} = 0 \quad (42)$$

$$\nu + w_i \gamma_i = 0 \quad (43)$$

$$g_i(x) - h_i(x, \mu) - w_i = 0. \quad (44)$$

LOQO uses Newton's method to solve these equations. It can be shown that as ν tends to zero this gives a solution to the KKT conditions for our problem. And by the theory in section 3 we know that this gives us the global optimum.

4.2. Solving the KKT condition via a sequence of SOCP

Although the algorithm presented in the previous section is much more efficient than the bisection algorithm of [6] for moderate scale problems (see Section 5) we have found that for large problems it converges very slowly. Therefore we also present an algorithm that solves the KKT conditions via a sequence of SOCPs.

It resembles the bisection algorithm, in that it solves a sequence of SOCPs. The difference is that instead of fixing μ to get a cone program we will make a simple approximation of the condition $g_i(x) - h_i(x, \mu) \leq 0$ with a cone condition such that the KKT-conditions of the resulting program approximates the KKT-conditions of the original program. Recall that in the bisection algorithm we solve feasibility problems of the type

$$\begin{aligned} P_\mu : \quad & \text{find} \quad x \\ & \text{s.t.} \quad g_i(x) - h_i(x, \mu) \leq 0 \\ & \qquad\; x \in S, \mu \geq 0 \end{aligned} \quad (45)$$

for a sequence of fixed $\mu = \{\mu_l\}$. Here μ is fixed since we want $h_i(x, \mu)$ to be an affine function squared. Recall that $h_i(x, \mu) = (\mu(a_{i3}^T x + b_{i3}))^2$. However instead of fixing μ we may choose to approximate $\mu(a_{i3}^T x + b_{i3})$ with its 1st order Taylor expansion around a point (x_l, μ_l). The Taylor expansion can be written

$$\mu(a_{i3}^T x + b_{i3}) \approx \mu_l(a_{i3}^T x + b_{i3}) + \Delta\mu(a_{i3}^T x_l + b_{i3}), \quad (46)$$

where $\Delta\mu = \mu - \mu_l$. Note that if the second term is disregarded we get P_μ. Let

$$h_{il}(x, \mu) = (\mu_l(a_{i3}^T x + b_{i3}) + \Delta\mu(a_{i3}^T x_l + b_{i3}))^2 \quad (47)$$

and consider the program

$$\begin{aligned} \tilde{P}_\mu : \quad & \min \quad \mu \\ & \text{s.t.} \quad g_i(x) - h_{il}(x, \mu) \leq 0 \\ & \qquad\; x \in S, \mu \geq 0. \end{aligned} \quad (48)$$

The first order conditions of this program are

$$\begin{pmatrix} 0 \\ 1 \end{pmatrix} + \sum_j \lambda_j \begin{pmatrix} \nabla g_j(x^*) - \nabla_x h_{jl}(x^*, \mu^*) \\ -\partial_\mu h_{jl}(x^*, \mu^*) \end{pmatrix} = 0 \quad (49)$$

$$g_i(x^*) - h_{il}(x^*, \mu^*) \leq 0 \quad (50)$$

$$\lambda_i \geq 0 \quad (51)$$

$$\lambda_i(g_i(x^*) - h_{il}(x^*, \mu^*)) = 0. \quad (52)$$

It is reasonable to assume that this program approximates problem P_2 well in a neighborhood around (x_l, μ_l). Therefore we define the sequence $\{x_l, \mu_l\}$ as follows. For a given (x_l, μ_l) we let x_{l+1} be the solution of the program (48). To ensure that (x_{l+1}, μ_{l+1}) is feasible in P_2 we put $\mu_{l+1} = \max_i f_i(x_{l+1})$. Note that $\{\mu_l\}$ is a descending sequence with $\mu_l \geq 0$ for all l.

We will see that if $\mu_{l+1} = \mu_l$ then x_{l+1}, μ_l also solves problem (34). We have that

$$h_{il}(x, \mu_l) = h_i(x, \mu_l) \quad (53)$$

$$\nabla_x h_{il}(x, \mu_l) = \nabla_x h_i(x, \mu_l). \quad (54)$$

Since both x_l and x_{l+1} are feasible we have $\partial_\mu h_{il}(x_{l+1}, \mu_l) > 0$. By rescaling the dual variables it is now easy to see that since the system (49)-(52) is solvable with (x_{l+1}, μ_l) then so is (35)-(38).

5. Experimental results

In this section we compare the proposed algorithms with the state of the art, which is the bisection algorithm. For the moderate scale problems we tested the algorithms on randomly generated instances of the triangulation, resection and homography problems of different sizes. The reported execution times are the total time spent in the optimization routines, that is, we do not include the time spent setting up the problem.

All the experiments have been carried out on a standard PC P4 3.0 GHz machine. For solving SOCP problems, we use the publicly available SeDuMi [12]. Both SeDuMi and LOQO are considered to be state-of-the-art optimization software and they are both optimized for efficiency. Still, we are aware of that the reported results dependent on the actual implementations, but it gives an indication of which approach is most efficient. Another measure of time complexity is the number of Newton iterations each method has to solve. Even though the Newton iterations are not equivalent for the different approaches, it gives, again, an indication of which scheme is preferred. This measure is mostly relevant for large-scale problems.

To achieve approximately the same accuracy for the different algorithms we chose the following termination criteria. For the bisection algorithm we used the difference between the upper and lower bounds. When the difference is less than 10^{-4} the algorithm terminates. LOQO uses a threshold on the duality gap as termination criteria. The threshold was set to 10^{-4}. For the SOCP algorithm we used $\Delta\mu \leq 10^{-4}$ as termination criteria. For each size we measured the average execution time for solving 100 instances of the problem. The results are shown in Table 2.

	bisection	*SOCP-approx.*	*LOQO*
Triangulation:			
5 cameras	1.23	.195	.00281
10 cameras	1.38	.207	.00358
20 cameras	1.29	.223	.00645
30 cameras	1.36	.234	.00969
Homography:			
10 points	1.05	.363	.00816
20 points	1.17	.373	.0128
30 points	1.22	.377	.0193
Resectioning:			
10 points	.823	.327	.0128
20 points	.994	.345	.0287
30 points	1.04	.349	.0418

Table 2. Average execution times (s) for 100 random instances of each problem.

For the large scale test we used the known rotation problem. Here we assume that the orientations of the cameras are known. The objective is to determine the 3D structure (in terms of 3D points) and the positions of the cameras. We have tested the SOCP-algorithm on two sequences. The first one is a sequence of 15 images of a flower and a chair (see Figure 2), and the second one is the well known dinosaur sequence (see Figure 3). For large scale problems, we have noticed that the LOQO algorithm sometimes stops prematurely, without converging to the global optimum. We believe that this is due to bad numerical conditioning. Therefore no experiments are reported with LOQO for these test scenarios.

The obtained reconstructions are shown in Figures 2and 3. Figure 4 shows the convergence of the SOCP-algorithm. For compairison we have also plotted the upper (dashed line) and lower bound (dotted line) of the bisection algorithm at each iteration. For both sequences the SOCP algorithm converges after 4 iterations. In contrast the bisection algorithm takes 15 iterations to achieve an accuracy of 10^{-4}. However, comparing the number of SOCPs solved is not completely fair since each SOCP solved by the bisection algorithm is slighty simpler. Therefore we also calculated the total number of Newton steps taken during the optimization. Table 3 shows the measured execution times and the total number of Newton steps.

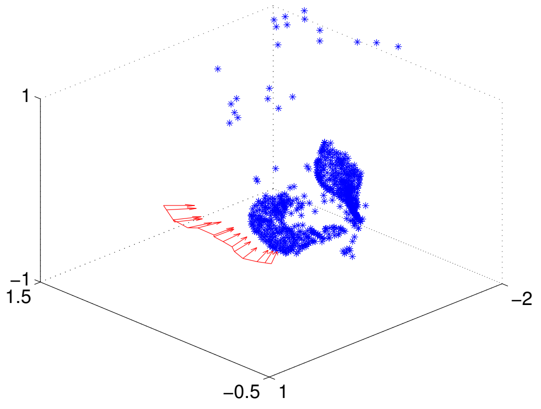

Figure 2. Three images from the flower sequence, and the reconstructed structure and motion.

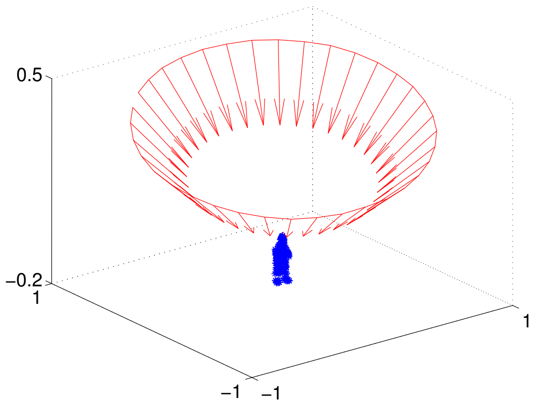

Figure 3. Three images form the dino sequence, and the reconstructed structure and motion.

6. Conclusions

We have analyzed a class of L_∞-problems for which reliable global solutions can be computed. We have shown several important convexity properties for these problems

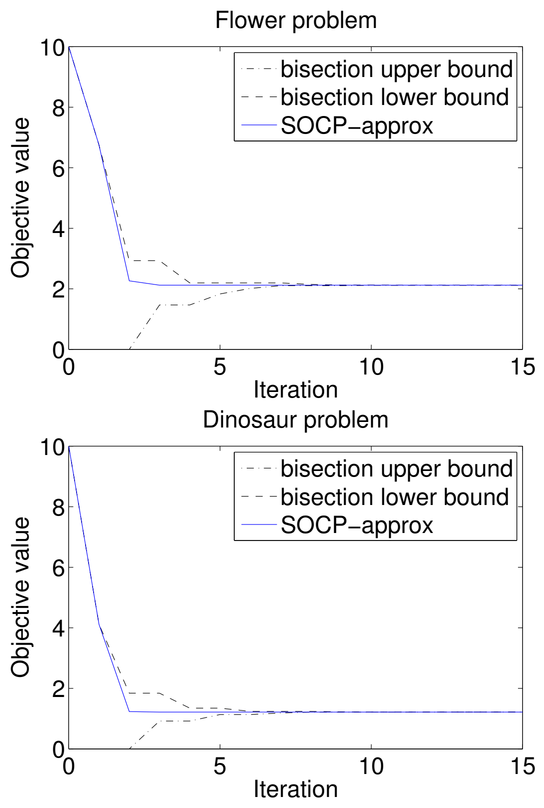

Figure 4. Convergence of the bisection and the SOCP algorithms for the flower sequence (top) and the dinosaur sequence (bottom).

	bisection	*SOCP-approx.*
Flower sequence:		
Execution times (s)	47.4	16.6
Newton iterations	261	87
Dinosaur sequence:		
Execution times (s)	34.0	15.9
Newton iterations	215	70

Table 3. Measured execution times (s) and total number of Newton iterations for computing the structure and motion of the flower and dinosaur sequences.

- including pseudoconvexity and necessary/sufficient conditions for a global minimum. Based on these results, it should be possible to design more efficient algorithms for multiview geometry problems. We have presented two algorithms for efficient optimization for this class of problems. Comparison to the state-of-the-art bisection algorithm shows considerable improvements both in terms of execution times and the total number of Newton iterations required.

Acknowledgments

This work has been supported by the European Commission's Sixth Framework Programme under grant no. 011838 as part of the Integrated Project SMErobot^{TM}, Swedish Foundation for Strategic Research (SSF) through the programme Vision in Cognitive Systems II (VISCOS II), Swedish Research Council through grants no. 2004-4579 'Image-Based Localisation and Recognition of Scenes' and no. 2005-3230 'Geometry of multi-camera systems'.

References

[1] Bazaraa, Sherali, and Shetty. *Nonlinear Programming, Theory and Algorithms.* Wiley, 1993.

[2] S. Boyd and L. Vandenberghe. *Convex Optimization*. Cambridge University Press, 2004.

[3] M. Farenzena, A. Fusiello, and A. Dovier. Reconstruction with interval constraints propagation. In *Proc. Conf. Computer Vision and Pattern Recognition*, pages 1185–1190. New York City, USA, 2006.

[4] R. Hartley and F. Schaffalitzky. L_∞ minimization in geometric reconstruction problems. In *Proc. Conf. Computer Vision and Pattern Recognition*, pages 504–509. 2004.

[5] R. I. Hartley and A. Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, 2004. Second Edition.

[6] F. Kahl. Multiple view geometry and the L_∞-norm. In *International Conference on Computer Vision*, pages 1002–1009. Beijing, China, 2005.

[7] Q. Ke and T. Kanade. Quasiconvex optimization for robust geometric reconstruction. In *International Conference on Computer Vision*, pages 986 – 993. Beijing, China, 2005.

[8] Q. Ke and T. Kanade. Uncertainty models in quasiconvex optimization for geometric reconstruction. In *Proc. Conf. Computer Vision and Pattern Recognition*, pages 1199 – 1205. New York City, USA, 2006.

[9] H. Li. A practical algorithm for l_∞ triangulation with outliers. In *Proc. Conf. Computer Vision and Pattern Recognition*. Minneapolis, USA, 2007.

[10] K. Sim and R. Hartley. Recovering camera motion using the L_∞-norm. In *Proc. Conf. Computer Vision and Pattern Recognition*, pages 1230–1237. New York City, USA, 2006.

[11] K. Sim and R. Hartley. Removing outliers using the L_∞-norm. In *Proc. Conf. Computer Vision and Pattern Recognition*, pages 485–492. New York City, USA, 2006.

[12] J. F. Sturm. Using sedumi 1.02, a matlab toolbox for optimization over symmetric cones. 1998.

[13] R. J. Vanderbei and D. F. Shanno. An interior point algorithm for nonconvex nonlinear programming. *Computational Optimization and Applications*, 13:231–252, 1999.