3D Structure from a Single Calibrated View Using Distance Constraints

Rubin Gong and Gang Xu, Regular Member

SUMMARY We propose a new method to recover scene points from a single calibrated view using some known distances among the points. This paper first introduces the problem and its relationship with the perspective *n* point problem. Then the minimal number of distances required to uniquely recover scene points are derived. The theory has been developed into a practical vision algorithm to calculate the initial points' coordinates using distance constraints. Finally SQP (Sequential Quadratic Programming) is used to optimize the initial estimations. It can minimize a cost function defined as the sum of squared reprojection errors while keeping the specified distance constraints strictly satisfied. Both simulation data and real scene images have been used to test the proposed method, and very good results have been obtained.

Key words: **single view reconstruction, perspective *n* point problem, constrained minimization, Sequential Quadratic Programming**

1. INTRODUCTION

Traditional image-based 3D reconstruction methods use multiple views to extract 3D geometry. However, when reconstructing scene structures with only one photograph or retrieving dynamic scene properties shot by one video camera, only a single view exists. To recover 3D scene structure from a single view, 3D scene constraints need to be utilized [1]. A variety of 3D scene geometry constraints have been identified and used in computer vision algorithms. Traditional constraints have used vanishing points, planes and geometric invariants constraints [2,3,4]. Some work has used the constraint of multiple objects of the same type in the scene separated by a simple translation [5]. However, these geometry constraints are limited to certain model types, for example, scenes composed of planes or other simple primitives.

The 3D scene distance constraints are used to recover 3D models from a single view in this paper. Since 3D scene distance constraints exist in almost every kind of scenes, the proposed technique can be applied in a wide variety of scenes that contain points, lines, and planar or freeform surfaces. In multiple view cases, 3D scene distance constraint has been applied in motion capture applications to upgrade affine reconstruction to a metric one [6]. In single view reconstruction applications, the 3D scene distance constraints have not been effectively utilized so far.

The closest work to ours is the traditional perspective *n* point problem. For *P3P* and *P4P* problems, many algorithms [7,8,9,10,11,12] have used the distance constraints only to calculate the pose of the camera. For *P3P* problem, there are multiple solutions. For *P4P* problem, the work done by Hu [16] has shown that the effective solution upper bound of the *P4P* problem is 5. The solution is generally unique outside of certain critical configurations [1,13,14,15]. In real applications, it is difficult to measure all the distances among scene points. Usually only a small number of distances are known in real uncontrolled environment. And under these circumstances, the following questions arise: given a set of correspondences between 3D reference points and their projections on one photograph and given some distances among the points, can we uniquely calculate the 3D coordinates of the points in camera coordinate system? If yes, how many distances do we need?

In the following sections, these questions will be clarified. Here are the main contributions of our work:

1. Given some scene points and some distances among the points, the minimal number of distances required to uniquely recover the scene points in non-degenerate cases are found.
2. The theory has been developed into a practical algorithm to calculate the initial 3D coordinates of the points from a single calibrated view.
3. Given the initial solutions of the 3D coordinates of the points, SQP (Sequential Quadratic Programming) has been used to optimize the initial solutions while keeping the specified distance constraints strictly satisfied.

The paper is organized as follows. In Section 2, we introduce the basic notations that are used in the paper. In Section 3, the distance configuration required to solve the problem is explored. In Section 4, the initial solutions of each point is calculated and determined. In Section 5, SQP is used to optimize the initial solutions. In Section 6, the implementation and the experimental results are presented. The methods developed in this paper are verified both on simulation data and real images. Finally we summarize the contributions of the paper.

2. BASIC NOTATIONS

Given a calibrated camera at **O** and *n* correspondences between 3D reference points $\mathbf{p_i}$ and their image projections $\mathbf{u_i}$, each pair of correspondences $\mathbf{p_i}\leftrightarrow\mathbf{u_i}$ and $\mathbf{p_j}\leftrightarrow\mathbf{u_j}$ gives an equation on the unknown camera-point distances $l_i=|\mathbf{p_i}-\mathbf{O}|$ and $l_j=|\mathbf{p_j}-\mathbf{O}|$(cf. Figure 1)**:**

$D_{ij}^2 = l_i^2 + l_j^2 - 2l_i l_j cos\theta_{ij}$.

Here D_{ij}=|$\mathbf{p_i}$-$\mathbf{p_j}$| is the distance between points $\mathbf{p_i}$ and $\mathbf{p_j}$ and it is also written as $D_{(i,j)}$ below. θ_{ij} is the 3D viewing angle subtended at camera center **O** by points $\mathbf{p_i}$ and $\mathbf{p_j}$ and it is also written as $\theta_{(i,j)}$ below.

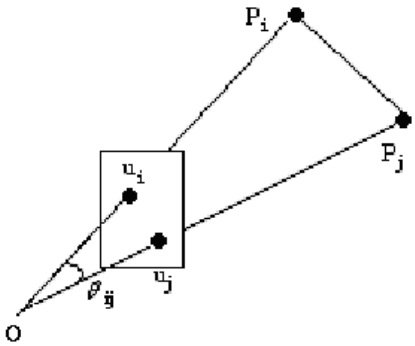

Figure 1: The basic geometry for each pair of the correspondences $\mathbf{p_i}\leftrightarrow\mathbf{u_i}$ and $\mathbf{p_j}\leftrightarrow\mathbf{u_j}$ between the 3D reference points and their images

The cosine value of the viewing angle is directly computed from the image projections and the calibration matrix **K** [9] as

$$cos\theta_{ij} = \frac{\mathbf{u_i}^T\mathbf{C}\mathbf{u_j}}{\sqrt{(\mathbf{u_i}^T\mathbf{C}\mathbf{u_i})(\mathbf{u_j}^T\mathbf{C}\mathbf{u_j})}}, \mathbf{C}=(\mathbf{K}\mathbf{K}^T)^{-1}.$$

Given a distance between two points $\mathbf{p_i}$ and $\mathbf{p_j}$, the quadratic equation can be rewritten as

$$f_{(i,j)}(l_i,l_j)=f_{ij}=l_i^2+l_j^2-2l_il_jcos\theta_{ij}-D_{ij}^2=0. \quad (1)$$

If there are m distances known in the scene, we can obtain a polynomial system consisting of m equations and n unknown variables l_{ik}=|$\mathbf{p_{ik}}$-$\mathbf{c}$|, where $1 \le k \le m$, ik and jk are index numbers of the unknown variables.

$$\begin{cases} f_{(i1,j1)}(l_{i1},l_{j1}) = 0, \\ f_{(i2,j2)}(l_{i2},l_{j2}) = 0, \\ \ldots\ldots \\ f_{(im,jm)}(l_{im},l_{jm}) = 0 \end{cases} \quad (2)$$

For some points in the scene, if the distance between two points is known, an edge can be drawn between them. A graph can be obtained from the scene points using the method. There is a one-to-one mapping relationship between the graph configuration and the polynomial system (2). The relationship between the solutions of the polynomial system and the graph configuration is explored in the next section.

3. GRAPH CONFIGURATION AND THE POLYNOMIAL SYSTEM SOLUTION

In this section, we consider the minimal number of distances required to solve the problem. Since all unknown variables in polynomial system (2) have no odd terms, if $\{l_1,l_2,l_3..,l_n\}$ is one solution of the polynomial system, then $\{-l_1,-l_2...,-l_n\}$ is also a solution of the same polynomial system. We only consider the real positive solutions (l_i>0,1<=i<=n) of polynomial system (2) below.

3.1 Multiple Solutions

If the original graph is unconnected, then it can be divided into several connected sub-graphs. The connected sub-graphs are independent of each other. For the original graph to be solved, each connected sub-graph has to be solved. If each connected sub-graph is solved, the original graph can then be solved. We consider only the connected graph case below.

If the graph has two vertices V_i and V_j and one edge, we can obtain only one quadratic equation $f_{ij}(l_i,l_j)$=0 from the graph topology. It is impossible to solve two variables from one equation. In order to solve the unknown variables from the equations, the number of equations must be equal or greater than the number of variables. It means that in the graph, the number of edges should be equal or greater than the number of vertices. According to basic graph theory, there is at least one cycle in the graph. If the graph has only one cycle, the system of equations (2) associated with the cycle has equal number of equations and variables. According to Bezout theorem [17,18], the polynomial system has no more than 2^n solutions in general case, where n is the number of edges in the cycle. Each vertex in the cycle can have no more than 2^{n-1} positive solutions. For an arbitrary vertex V not in the cycle, there is a path P that connects vertex V and the cycle. If the length of path P is m, then vertex V has at most 2^{n+m-1} positive solutions after solving the polynomial system associated with the cycle and the path. As the example shown in Figure 2, vertex V has at most 2^{n+2-1} positive solutions after solving the associated polynomial system, where is n is the length of cycle.

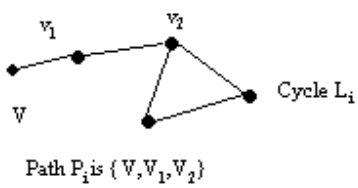

Figure 2: Vertex V has a finite solution set S_{Li} associated with cycle L_i and path P_i

3.2 Unique Solution

In practical applications, we need to find the unique solution from a graph configuration.

If there are two cycles L_i and L_j in a graph, then vertex V has one solution set S_{Li} associated with cycle L_i and path P_i and another solution set S_{Lj} associated with cycle L_j and path P_j. The true solution is in the intersection set of S_{Li} and S_{Lj}.

Consider the graph configuration in Figure 3, for an arbitrary vertex, say $\mathbf{W_{k-1}}$, it may have two positive solutions S_{k-1} and S_{k-1}'. Then vertex $\mathbf{W_k}$ may have at most four positive solutions by solving the equation associated with edge $\mathbf{W_{k-1}W_k}$. If two out of the four positive solutions are equal, we define it as special case (a), since it requires the special value of $D_{(k-1,k)}$ and $\theta_{(k-1,k)}$. Here $D_{(k-1,k)}$ is the length of edge $\mathbf{W_{k-1}W_k}$, $\theta_{(k-1,k)}$ is the 3D viewing angle subtended at the camera center **O** by point $\mathbf{W_{k-1}}$ and $\mathbf{W_k}$. One example of special case (a) is shown in Figure 4, if we

know the solutions of vertex C, after solving the equation associated with edge CD, we obtain two positive solutions of vertex D. For the two different solutions S_D and S_{D1} of vertex D, after solving the two different equations associated with edge DE, the same solution of vertex E is obtained. Most of the degenerate examples discussed in literature so far are caused by this kind of special case.

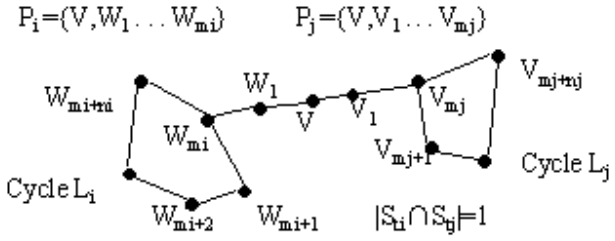

Figure 3: The intersection solution set of S_{Li} and S_{Lj} has only one positive real solution outside of two special cases

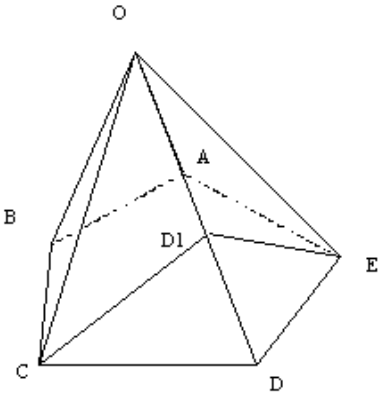

Figure 4: Special case (a): two positive solutions of vertex D correspond to one solution of vertex E

Assume that the special case (a) has been avoided in the graph configuration of Figure 3. Vertex V may have some positive solutions in solution set S_{Li}. It may also have some positive solutions in solution set S_{Lj}. Outside of the true solution S_v of vertex V, another positive solution S_v' of vertex V may exist in both solution sets and $S_v \neq S_v'$. We define it as special case (b), since it requires some special values of the edges and 3D viewing angles. When this kind of special case happens, two positive solution sets exist for all vertices. For each vertex, the solutions in the two sets are different. For this kind of special case, it has seldom been discussed in literature before.

Outside of the two kinds of special cases, vertex V has only one positive solution in the graph configuration of Figure 3. The unique solution of each vertex in the graph can thus be propagated along the paths and cycles according to the unique solution of vertex V.

In practical applications, we often consider the case where two cycles share a common vertex, as illustrated in Figure 5. For this kind of configuration, the common vertex V has a unique positive solution S_0 associated with the two cycles outside of the two special cases, and the unique solutions of other vertices along cycle L_1 and cycle L_2 can be uniquely determined by the unique solution of vertex V.

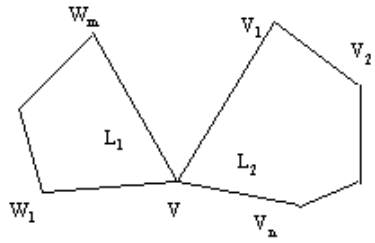

Figure 5: Two cycles share a common vertex

3.3 Avoiding Multiple Solutions

The polynomial system associated with the graph configuration of Figure 3 may have multiple solutions. This can be avoided using geometric methods and algebraic methods. We first avoid multiple solutions using geometric methods.

3.3.1 Geometric Methods

If the length of the path between two vertices is 2, then we can find the geometric configuration corresponding to special case (a). One example is shown in Figure 6. It is the side view of the example in Figure 4. Since $|CD_1|=|CD|$ and $|ED_1|=|ED|$, there is a point H on line D_1D such that $CH \perp D_1D$ and $EH \perp D_1D$. If the camera and graph are carefully configured, this kind of cases can be avoided. If the length of the path between two vertices is 3, the geometric meanings corresponding to special case (a) are also found. One example is shown in Figure 7. There is a point H_1 on line D_1D such that $CH_1 \perp D_1D$. There is another point H_2 on line F_1F such that $EH_2 \perp F_1F$. At the same time $|CD_1|=|CD|$, $|EF|=|EF_1|$ and $|D_1F_1|=|DF|$. It is difficult to avoid this kind of geometric configurations. If the length of the path between two vertices is 4 or more, the graph configurations do not have much geometric meanings. It is difficult to avoid multiple solutions when the graph configurations do not have much geometric meanings.

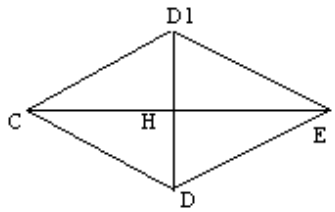

Figure 6: The length of the path between vertex C and E is 2

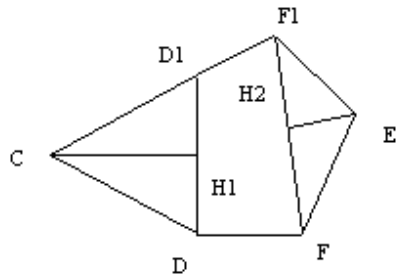

Figure 7: The length of the path between vertex C and E is 3

3.3.2 Algebraic Methods

The graph configuration of special case (b) usually has no geometric meanings. Some graph configurations of special case (a) do not have much geometric meanings either. Since it is difficult to avoid these graph configurations, we hope to avoid multiple solutions using algebraic methods. For the graph configuration in Figure 3, the polynomial system associated with the graph configuration may have multiple effective solutions. There is no general method to select the unique true solution from multiple solutions. All we can do is to add more edges to the graph configuration in Figure 3 to reduce the solution space. The step is repeated until a unique positive solution is obtained or all the distances in the graph configuration are used up. If all the distances are used up and multiple solutions still exist, additional vertex and edges should be added to the graph. The extreme case seldom happens when there are more than 4 points.

4. INITIAL SOLUTIONS

The theory declared in Section 3 is developed into a practical vision algorithm in this section. We consider the application of recovering 3D scene structure from a single calibrated view using distance constraints. The numerical problems in solving the polynomial system (2) are first considered.

4.1 Solving Polynomial Equations

Many methods exist to solve a polynomial system of equations. For example, one can use Grobner basis [19,20] to reduce variables and get one univariate polynomial equation. The univariate polynomial equation can then be solved using Laguerre's method [8]. Certain system of quadratic equations can even be solved using linear methods [9]. In real applications, the solutions of polynomial system (2) are often affected by image noise. So the common solutions of two cycles are not exactly same. Sometimes some false solutions may be very close to the true one. Sensitivity analysis of the polynomial system solutions with respect to the polynomial system coefficients should be studied [8,21]. The sensitivity analysis result in [21] shows, if the polynomial system associated with the graph configuration of Figure 3 has a unique solution, then the errors in the solution is bounded by the error in the polynomial system coefficients. In the implementation, Mathematica is used to solve the polynomial system. It is a kind of numerical Grobner basis technique. We are mainly concerned with the sensitivity of the solutions with respect to the input coefficients. A large sensitivity of the solution with respect to the noises may lead to a large error in the final result.

4.2 Sensitivity Analysis

The sensitivity analysis method used in [8] for polynomials is generalized to polynomial systems in our work. Consider the polynomial system (2), if we denote all the variables as **x**, all the coefficients as **a** and all the equations as **F**, the polynomial system can be rewritten as:

$$\mathbf{F}(\mathbf{x},\mathbf{a})=\mathbf{0}. \tag{3}$$

The sensitivity of the solutions of a polynomial system with respect to a change in the coefficients can be derived by assuming that the zero location is a function of the coefficients. For a solution **z** of equation (3), we have

$$\mathbf{F}(\mathbf{x}(\mathbf{a}),\mathbf{a})|_{\mathbf{x}=\mathbf{z}}=\mathbf{0}\,. \tag{4}$$

Differentiating (4) with respect to **a** gives

$$\left[\frac{\partial\mathbf{F}}{\partial\mathbf{a}}+\frac{\partial\mathbf{F}}{\partial\mathbf{x}}\frac{\mathrm{d}\mathbf{x}}{\mathrm{d}\mathbf{a}}\right]|_{\mathbf{x}=\mathbf{z}}=\mathbf{0}\,. \tag{5}$$

If there are equal number of equations and variables in (3), and matrix $\frac{\partial\mathbf{F}}{\partial\mathbf{x}}$ is nonsingular, we have

$$\frac{\mathrm{d}\mathbf{x}}{\mathrm{d}\mathbf{a}}|_{\mathbf{x}=\mathbf{z}}=-\left[\left[\frac{\partial\mathbf{F}}{\partial\mathbf{x}}\right]^{-1}\frac{\partial\mathbf{F}}{\partial\mathbf{a}}\right]|_{\mathbf{x}=\mathbf{z}}\,. \tag{6}$$

The matrix in (6) gives the sensitivity of each variable with respect to each coefficient. It is referred to as stability matrix later. If matrix $\frac{\partial\mathbf{F}}{\partial\mathbf{x}}$ is near singular or singular, the solution is unstable. The elements in the corresponding sensitivity matrix are very large [14].

If vector **a** consists of all points projections and distances, the sensitivity matrix of each solution with respect to each point projection and each distance can be obtained using equation (6). For small perturbations of image projections and 3D distances near solution **z**, the variation in the solution can be approximated in first order as

$$\Delta\mathbf{x}\approx\frac{\mathrm{d}\mathbf{x}}{\mathrm{d}\mathbf{a}}\Delta\mathbf{a}\,. \tag{7}$$

The covariance matrix of **x** can be calculated as

$$\mathrm{COV}(\mathbf{x})\approx(\frac{\mathrm{d}\mathbf{x}}{\mathrm{d}\mathbf{a}})\mathrm{COV}(\mathbf{a})(\frac{\mathrm{d}\mathbf{x}}{\mathrm{d}\mathbf{a}})^{\mathrm{T}}\,. \tag{8}$$

Here COV(**a**) is a diagonal matrix, the first $2n$ diagonal elements are Gaussian image noise parameter σ^2 and the last n diagonal elements are squares of 3D distance standard deviations, n is the number of equations. The square root of the diagonal items of the covariance matrix COV(**x**) gives the standard deviations of each element in **x**. For each solution vector **z** of equation (3), the standard deviations of each element around solution **z** can be computed.

For the common vertex V in Figure 5, assume the true solution is r_{true}, it has one approximate solutions r_1 from cycle L_1 and another approximate solution r_2 from cycle L_2. We have the following inequality:

$$\begin{aligned}&|r_1-r_2|\le|(r_1-r_{true})-(r_2-r_{true})|\\&\le|r_1-r_{true}|+|r_2-r_{true}|\\&=\Delta r_1+\Delta r_2\,,\end{aligned} \tag{9}$$

where Δr_i is the variance of $r_i(i=1,2)$. The distribution of Δr_i can be approximated by Gaussian distribution $\mathrm{N}(0,\sigma_i^2)$, σ_i is the standard deviation of variable r_i. From inequality (9) we have

$$|r_1 - r_2| \le \frac{\Delta r_1}{r_1} r_1 + \frac{\Delta r_2}{r_2} r_2 = (\varepsilon_1 r_1 + \varepsilon_2 r_2), \quad (10)$$

where $\varepsilon_i = \frac{\Delta r_i}{r_i}, (i = 1,2)$ can be considered as the relative drift of the approximated solution.

4.3 Stable Solutions

If the upper bound in inequality (10) is too large, the false solutions may also satisfy the inequalities. It will be difficult to differentiate the false solutions from the true one. The value of the upper bound may be caused by several factors.

From equation (8), we can see that the standard deviations of the solutions can be calculated from gaussian image noise, distance error and the sensitivity matrix. As the image noises and distance errors increase, the standard deviations of the solutions will increase. If the value of certain element in the sensitivity matrix increases, it will also increase the standard deviations of some elements.

For a relatively small error in the input coefficients, we expect a relatively small drift of the approximate solutions. More formally, this can be written as

$$\varepsilon_i \le \varepsilon, \quad (11)$$

where ε_i is the relative drift of an approximate solution, ε is some tolerable value, usually it is between 2%~4% for practical applications. If the standard deviation of one solution satisfies inequality (11), the solution is considered as a stable one. If the calculated relative drift is larger than ε, we think that the solution is unstable and may contain large errors. The difference between the two solutions can be approximated as

$$|r_1 - r_2| \le (\varepsilon_1 r_1 + \varepsilon_2 r_2) \le \varepsilon(r_1 + r_2). \quad (12)$$

Some factors that may contribute to the value of ε are studied and verified in the experiments. First the noise in the image and the error in the distances should be as small as possible, as can be verified in section 6. The depth of the graph configuration is also considered. The experiments in section 6 show that if the depth of the graph configuration is large, a small perturbation in the distance will result in a large variation in the depth. So if the graph configuration is far from the camera, the false candidate solutions and the true candidate one may all satisfy inequality (9). It is difficult to obtain an absolute accurate solution. But it is shown in the experiment that the relative drift of the solutions does not vary much when the depth of the graph configuration changes. The corresponding false solutions and the true one may all satisfy inequality (12). If only a relatively accurate solution is wanted, one of them can be selected as the candidate solution. We do not find apparent relationship between the standard deviations of the solutions and the cycle length from the experiments. Since it is slow to solve a quadratic polynomial system with more than 6 equations, the cycle length is limited under 6 in this implementation.

4.4 Determination of Common Solutions

The unique geometric solutions of the common vertices are calculated in this section. Only the solutions that satisfy inequality (11) are considered in this subsection. For two positive solutions r_1 and r_2 of one common vertex connected with two different cycles, if the inequality (12) is satisfied, then the two solutions are then considered to be sufficiently close.

The following algorithm has been used in the implementation to decide the common solutions. Assume vertex V is a common vertex connected with several different cycles as in Figure 3. It has a series of positive solutions associated each cycle. Since the true candidate solutions of the polynomial system do not drift far from the true value when inequality (12) is satisfied, the property can be utilized to determine the common solutions of different cycles. The detailed algorithm is described below.

Step 1. For each common vertex, collect the solutions of the vertex associated with different cycles. For each solution, a set is built with the solution as the element.

Step 2. If all the elements in one set are sufficiently close to all the elements in another set (inequality (12) is satisfied), the two sets are combined together. The combing operation is repeated until no sets are available to be combined.

Step 3. The set with the largest number of elements is chosen to be the set contains the common solutions of the vertex. The mean of the set is calculated and used as the unique solution of the vertex.

Step 4. If two or more sets have equal largest number of elements, additional cycles should be added and attached to the common vertex. Go back to step 1.

If geometrically two solutions exist for the graph in Figure 3, then it is difficult to differentiate the true solution from the false solutions of certain common vertex. Under these circumstances, usually three or more cycles are needed in the above algorithm to determine the unique solution of the common vertex.

4.5 Propagating Unique Solutions Along Cycles

Once the solutions of the common vertices have been determined. The solutions can be used to propagate the unique solution of vertices along the cycles and paths attached to the common vertices. The detailed algorithm is described below.

Step 1. For each cycle, the common vertex may have one or more candidate solutions that are very close to each other (inequality (12) is satisfied).

Step 2. If the common vertex has only one candidate solution in the cycle, then the unique solution of each vertex along the cycle can be determined.

Step 3. If the common vertex has two or more candidate solutions in the cycle, the solutions of the vertices along the paths and cycle are uncertain.

Step 3.1 If the different solution sets of one cycle are close enough (each pair of solutions satisfy inequality (12)), then arbitrary one set can be selected as the unique solution.

Step 3.2 The different solution sets of one cycle are not close enough. In order to differentiate the true candidate solution set from the false solution sets, additional cycles should be added to the graph and attached to one uncertain vertex in the cycle. Once

the correct solution of the uncertain vertex has been determined in the cycle using the algorithm described in subsection 4.4. Go back to step1.

One example is shown in the simulation experiment. For the common vertex, it may have two or more candidate common solutions in one cycle. This may be caused by geometric configurations or numerical problems. Under these circumstances, other vertex should be used instead to find the unique solutions of the vertices in the cycle.

5. OPTIMIZATION BY SQP

The unique solution of the points can be successfully calculated if the failure cases discussed above are avoided. If more accurate results are desired, we need to optimize the initial solutions using redundant distances in the graph.

5.1 Objective Function and Constraints

The optimization procedure must minimize the sum of the distances between the observed pixels and the projections of the 3D scene points while keeping the distance constraints satisfied. Formally it can be represented as:

$$\min \mathrm{M} = \frac{1}{2}\sum_{i=1}^{n}\{(f\,\frac{x_i}{z_i}+u_0-u_i)^2+(f\,\frac{yi}{zi}+v_0-v_i)^2\} \qquad (13)$$

subject to the following constraints:

$h_{jk}=(x_k-x_j)^2+(y_k-y_j)^2+(z_k-z_j)^2-D_{jk}^2=0.\ 1\le j\le n, j\le k\le n$.

Here n is the total points in the connected graph G, (x_k,y_k,z_k) and (x_j,y_j,z_j) are coordinates of point $\mathbf{p_k}$ and point $\mathbf{p_j}$. The objective is to minimize function M(**X**) under the constraints $h_{jk}=0$ where **X** is the vector of the 3D coordinates of the points.

5.2 Constrained minimization

For each 3D point, it has 3 degree of freedom. For two 3D points, they have 6 degree of freedom. If there is a distance between them, the degree of freedom can be reduced to 5. If there are m distances among n points, the degree of freedom can be reduced to $3\times n-m$. In theory, the redundant variables can be reduced and unconstrained optimization algorithm can be used to solve the problem. However, the constraints are all quadratic equations and it is difficult to solve the quadratic system of equations to reduce the redundant variables. To avoid this problem, constrained minimization algorithm is used here.

The basic strategy we take is to use the Sequential Quadratic Programming (SQP) method [22,23,24]. It has been proved highly effective for solving constrained optimization problems with smooth nonlinear functions in the objective and constraints. The basic principle of sequential approximations is to replace the given nonlinear problem by a sequence of quadratic sub-problems that are easier to solve. Detailed procedure is described below. Consider the equality constraint problem:

min f(**x**)

subject to **h(x)=0**.

Here **x** is the desired variable vector, f(**x**) is the objective function and **h(x)** is the vector of equality constraints. Using a Lagrange-Newton method [25], at the kth iteration, we have

$$\begin{bmatrix} \mathbf{W_k} & \mathbf{A_k}^{\mathbf{T}} \\ \mathbf{A_k} & \mathbf{O} \end{bmatrix}\begin{bmatrix} \mathbf{s_k} \\ \boldsymbol{\lambda}_{\mathbf{k+1}} \end{bmatrix} = -\begin{bmatrix} \nabla \mathbf{f_k}^{\mathbf{T}} \\ \mathbf{h_k} \end{bmatrix},$$

where $\mathbf{W}=\nabla^2\mathbf{f}$, $\mathbf{A}=\nabla\mathbf{h}$, and $\boldsymbol{\lambda}$ is the vector of Lagrange multipliers. Solving the above equations iteratively, we obtain the iterates $\mathbf{x_{k+1}}=\mathbf{x_k}+\mathbf{s_k}$ and $\boldsymbol{\lambda}_{\mathbf{k+1}}$, which should eventually approach $\mathbf{x_f}$ and $\boldsymbol{\lambda}_{\mathbf{f}}$, the optimal values.

Proper convergence properties are achieved with some modifications on the basic SQP algorithm. We may view s_k as a search direction and define the iteration as $\mathbf{x_{k+1}}=\mathbf{x_k}+\theta_k\mathbf{s_k}$, where θ_k is introduced and computed by minimizing an appropriate merit function along the search direction. Given an initial estimate **X** sufficiently close to the solution, we try to improve this by using SQP. At each step the step size $\mathbf{s_k}$ and the new Lagrange multiplier vector $\boldsymbol{\lambda}$ are recalculated. At the optimum point $\mathbf{s_k}$ vanishes and the specified constraints are all strictly satisfied. We can use this as an indication flag to stop the iteration.

6. IMPLEMENTAION AND EXPERIMENTAL RESULTS

We first outline the implementation of the method. Then we study the sensitivity of the polynomial equations. Finally we do experiments on both simulation data and real images.

6.1 Implementation

The implementation can be outlined as below:

1. Data Processing: Collect the image points (u_i,v_i), the inter-point distances D_{ij} and the camera intrinsic parameters. Compute the cosines of angle θ_{ij} from the image points.
2. Collect and select different cycles in the graph, use polynomial system solver to solve the system of equations and find the common solutions from different cycles.
3. For each cycle, the solutions of the common vertex are used to find the unique solutions of other vertices along the cycle.
4. Calculate the initial 3D point coordinates.
5. Optimize the initial 3D point coordinates using SQP.

The above steps can be iterated until the recovered 3D model satisfies the demands.

6.2 Sensitivity Analysis

We have studied the sensitivity of the polynomial solutions with respect to image noises. The camera and graph configuration are fixed. We find that the standard deviations and relative drift of the solution increase as the Gaussian noise parameter σ increases. One example is shown in Figure 8 and in Figure 9.

We have studied the sensitivity of the polynomial solutions with respect to the distance errors. The camera and graph configuration are fixed. We find that the standard deviations of the solutions increase as the distance error increases. One example is shown in Figure 10.

The standard deviations of the polynomial solutions with respect to the depth of the graph are also analyzed. We find that

when the depth of the graph configuration becomes larger, the standard deviations increase very fast. One example is shown in Figure 11. The graph configuration is kept fixed, while the depth is pushed 2 units farther every time. We can also see that the relative drift of the solutions does not have much relationship with the depth of the graph. One example is shown in Figure 12. We find that there is no apparent relationship between the cycle length and the variances of the solutions. One example is shown in Figure 13. Here cycle length changes from 3 to 6. The standard deviations of 3 common vertices of these cycles are compared in the graph.

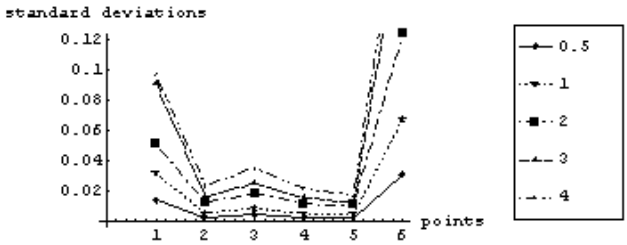

Figure 8: Relationship between the standard deviations and the gaussian noise parameter σ=0.5,1,2,3,4 pixels.

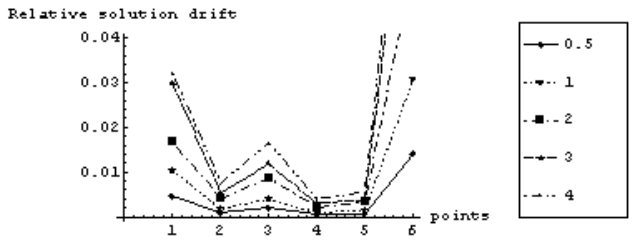

Figure 9: Relationship between the relative solution drift and the gaussian noise parameter σ=0.5,1,2,3,4 pixels.

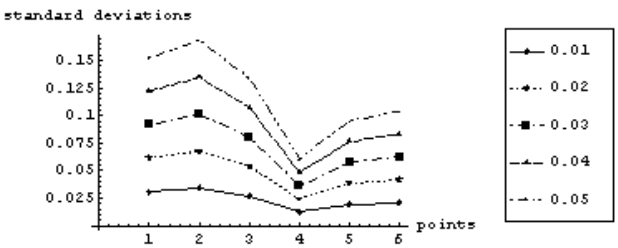

Figure 10: Relationship between the standard deviations and the distance error (the standard deviations of the distance error changes from 0.01 to 0.05).

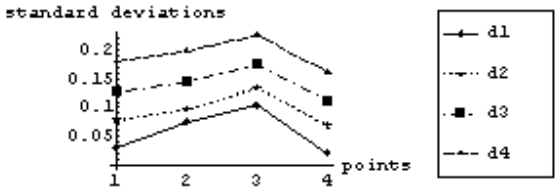

Figure 11: Relationship between the standard deviations of the solutions and the depth of the graph (dk,k=1,2,3,4 represents for the 4 different depths of the graph configuration)

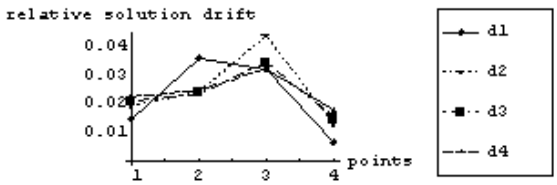

Figure 12: The relative drift of the solutions does not have much relationship with the depth of the graph(dk,k=1,2,3,4 represents for 4 different depths of the graph configuration)

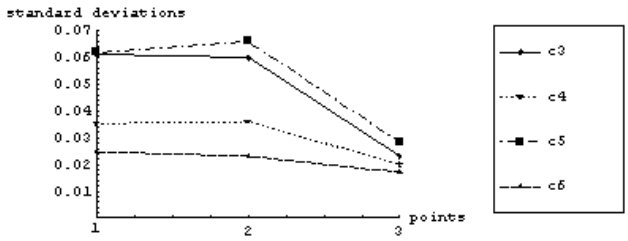

Figure 13: No apparent relationship between the standard deviations of the solutions and the cycle length(ck,k=3,4,5,6 represents for cycles with k edges)

6.3 Simulatation

The simulation data are shown in Figure 14, the image size is 500*500 pixels, focal length f is 1000 pixels, and the principle point value u_0 is 250 pixels, v_0 is 250 pixels. 6 image points are generated with isotropic Gaussian noise of σ=0.5 pixel. 9 distances are used in the experiment and they are (1 2), (1 3), (2 3), (1 4), (4 5), (5 6), (1 6), (3 5) (3 6). Three cycles are used to calculate the initial solution. They are cycle1 (1 2 3), cycle2 (1 4 5 6) and cycle3 (3 5 6).

Determining Initial Solutions

It can be seen from Figure 15 that vertex 1 has total 6 solutions. The first solution and the sixth solution are very close to each other (inequality (12) is satisfied). The two solutions are considered as the common solutions. It can also be seen that there are 4 solutions of vertex 3. The first solution and the second and the eighth solution are very close to each other (inequality (12) is satisfied). The three solutions are considered

as the common solutions.

After the solutions of the common vertices have been decided, we need to find the unique solutions of other vertices along the cycles. Vertex 1 is the common vertex of cycle 1 and cycle 2. The two common solutions of vertex 1 come from the two cycles. So we can find the unique solutions of other vertices in cycle 1 and cycle 2 without difficulty. Vertex 3 is the common vertex of cycle 1 and cycle 3. Two solutions come from cycle 1 and one solution comes from cycle 3. So we can find the unique solutions of vertices in cycle 3 without difficulty. But it is difficult to find the unique solutions of vertices in cycle 1. A new cycle should be added and attached to an uncertain vertex in cycle 1. Here cycle 2 is added and attached to vertex 1. The unique solution of each vertex in cycle 1 can then be decided by using the unique solution of vertex 1.

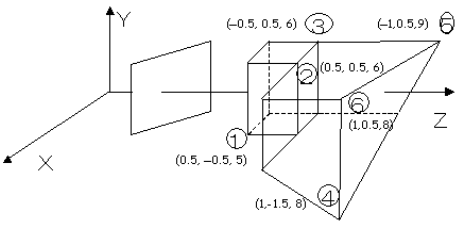

Figure 14: Graph configuration of simulation points

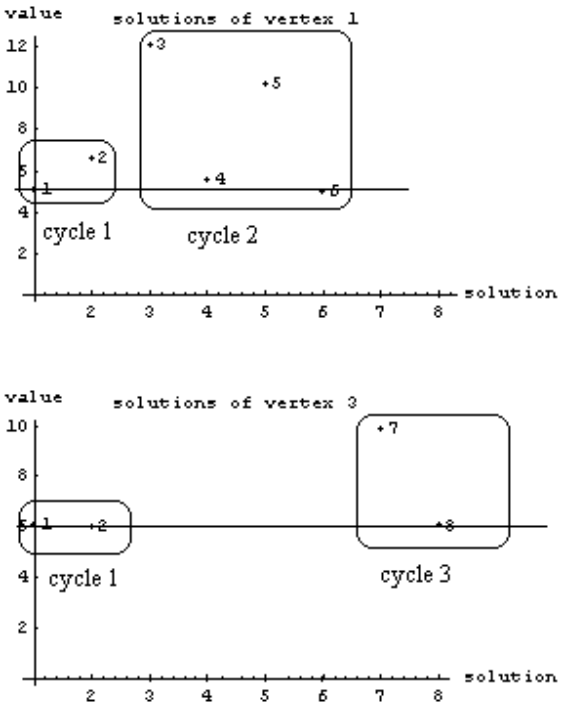

Figure 15: Determination of the common solutions

Optimization

We then feed the initial 3D coordinates into the SQP optimization step. The SQP optimization step converges in 10 steps. The specified distance constraints are all satisfied. The maximum error of the specified distances is no more than 0.000004. This shows that SQP can keep the specified constraints strictly satisfied. The total residual error M_0=0.477884(c.f. formula (5)). The total number of points n_0=6. The square root of mean residual error $\sqrt{\frac{M_0}{n_0}}$ is 0.282219 pixels.

The error of the initialized coordinates and the optimized coordinates are illustrated in Figure 16. The error is computed as $\|\mathbf{X}-\mathbf{X}_{\text{True}}\|$. $\mathbf{X}$ is the calculated coordinates of the points and $\mathbf{X}_{\text{True}}$ is the true coordinates of the points. It can be seen that the optimized coordinates are more close to the true coordinates than the initialized coordinates.

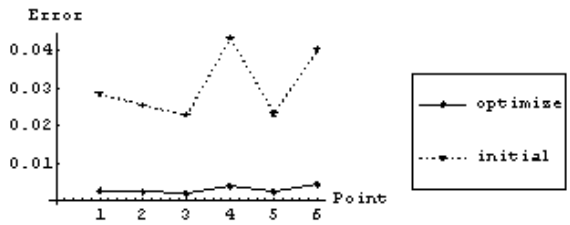

Figure 16: Initial and optimized error of the points

6.4 Real Images

For experiments on real images, two examples are given here. The camera is Power Shot Pro 70, a digital camera manufactured by Canon. It is calibrated using 3DM Calibrator of 3D Media Co., Ltd.

The first example is shown in Figure 17. The image size is 1525*1021 pixels. The focal length f is 2095.1772 pixels, and the principle point value u_0 is 718.1390 pixels, v_0 is 527.2681 pixels. There are 10 points and 14 known distances. The 14 distances are (1 2), (1 3), (2 3), (2 4), (2 5), (2 6), (3 4), (4 6), (5 6), (7 8), (7 9), (7 10), (8 10), (9 10). Six cycles are used to calculate the initial solution. They are cycle1 (1 2 3), cycle2 (2 3 4), cycle3 (2 4 6), cycle4 (2 5 6), cycle5 (7 8 10) and cycle6 (7 9 10).

Figure 17: A real image with some known distances

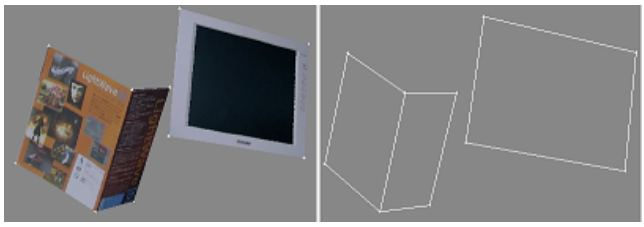

Figure 18: Textured VRML model and the wireframe view of the recovered 3D structure

Edges	Calculated dist.(cm)	Measured dist(cm)
(1,5)	25.3249	25.31
(1,4)	33.2165	33.22
(3,6)	25.4925	25.50
(4,5)	27.9465	27.95

Table 1: Calculated and measured distances of the first example

The unique common solution of different cycles can be selected. The unique solutions of all vertices in the graph are then determined. Finally we feed the initial 3D coordinates into SQP. SQP converges within 10 steps. The total residual error M_1=17.0017(c.f. formula (5)). The total number of points n_1=10. The square root of mean residual error $\sqrt{\frac{M_1}{n_1}}$ is 1.3039 pixels. The recovered 3D structure is shown in Figure 18. The specified distance constraints are all satisfied and the maximum error is no more than 0.000004(cm). This shows that SQP can keep the specified constraints strictly satisfied even for real images. The optimized 3D coordinates have been used to calculate the distances of several points. They are compared with the measured distances in Table 1.

A more complex object is shown in Figure 19. The image size is 1499*995 pixels. The focal length f is 1613.1632 pixels, and the principle point value u_0 is 671.284 pixels, v_0 is 594.9705 pixels. There are 30 points and 48 known distances. 21 cycles are used to calculate the initial solution. The initial coordinates of the points in the graph can be calculated using the described method. Finally we feed the initial 3D coordinates into SQP. SQP converges within 20 steps. The total residual error M_1=75.2903(c.f. formula (5)). The total number of points n_1=30. The square root of mean residual error $\sqrt{\frac{M_1}{n_1}}$ is 1.5842 pixels. The recovered 3D structure and the wireframe view are shown in Figure 20. The specified distance constraints are all satisfied and the maximum error is no more than 0.000005(cm). Some distances have been measured and they are compared with the estimated value in Table 2. The error in Table 2 is larger than that in Table 1. The main reason is that the object is not a rigid model and the measured distances are not as accurate as the previous example. So the unspecified distances have larger error. For a curved surface, a large number of polygons are required to approximate it. Due to the number of polygons is small, the edges between the polygons can be seen in the recovered model. And the recovered model looks a little unnatural. From the wireframe view, we can see that the recovered polygons approximate the shape well.

7. Conclusions

In this paper, we have proposed a new technique to recover 3D scene information from one calibrated image. It is simple and easy to use. It can recover 3D information by some known distances in the scene with a single calibrated view and can be adapted to a wide variety of scenes. The proposed algorithm consists of solving systems of quadratic polynomial equations followed by a SQP constrained minimization refinement. Both computer simulation and real scene have been used to test the proposed technique. The results are good and all the specified constraints are strictly satisfied.

Future work will examine several topics. More powerful technique is needed to solve quadratic polynomial equations. Only static scene is recovered in the current work, in the future work, dynamic scene can also be modeled.

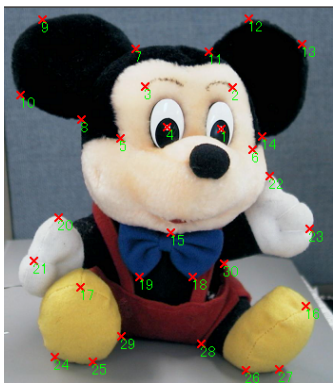

Figure 19: A toy image with some known distances

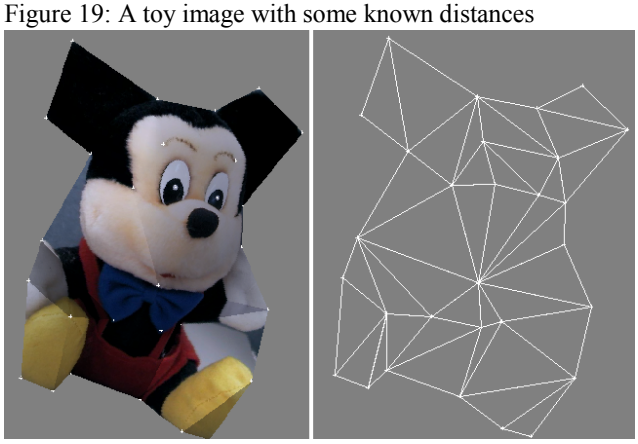

Figure 20: The VRML model of the toy and the wireframe view of the recovered 3D structure

Edges	Calculated dist.(cm)	Measured dist(cm)
(1,13)	8.8777	9.00
(16,21)	14.0889	14.10
(13,23)	10.5020	10.32
(16,17)	12.0150	12.20

Table 2: Calculated and measured distances of the toy example

References

[1] R. Hartley and A. Zisserman Multiple View Geometry, Cambridge University Press, 2000.

[2] Antonio Criminisi and Ian D. Reid and Andrew Zisserman, Single view metrology. International Journal of Computer Vision,vol 40,number 2, pp. 123-148, Sep 2000

[3] Y.Horry,K.anjyo,K.Arai," Tour into the picture:using a spidery mesh interace to make animation from a single image", ACM SIGGRAPH Proceedings,pp.225-232,1997

[4] H.-Y.Shum,M.Han, and R.Szeliski. "Interactive construction of 3D models from panoramic mosaics", IEEE Conf. On CVPR, pp. 427-433, 1998.

[5] R.Chaudhary, S.Chaudhary, J.B.Srivastava. Reconstruction-Based Recognition of Scenes with Translationally Repeated Quadrics. T-PAMI, June 2001
[6] Uncalibrated Motion Capture Exploiting Articulated Structure Constraints . D. Liebowitz and S. Carlsson. Proc. 8th Intl. Conf. Computer Vision , 2001
[7] M.A.Fischler and R.C.Bolles ”Random Sample Consensus:A paradigm for Model Fitting with Applications to Image Analysis and Automated Cartography”,Comm.Acm,vol.24,no.6,pp. 381-395,1981
[8] R.M. Haralick,C.N. Lee,K. Ottenberg, and M.Noelle,”Analysis and Solution of the Three Point Perspective Pose Estimation Problem”, Proc. IEEE Conf. Computer Vision and Pattern Recognition, pp.592-598,1991.
[9] Long Quan and Zhongdan Lan “Linear N-Point Camera Pose Determination”, IEEE Trans. Pattern Analysis and Machine Intelligence, pp.778-783, vol. 21 no.7 July 1999
[10] D.Lowe,” Fitting Parameterized Three-Dimensional Models to Images”, IEEE Trans. Pattern Analysis and Machine Intelligence, vol 13, no. 5, pp.441-450, May 1991
[11] J.S.C Yuan,” A General Photogrammetric Solution for the Determining Object Position and Orientation”, IEEE Trans. Robotics and Automation , vol. 5, no. 2, pp.129-142, Apr.1989
[12] M.Dhome,M.Richetin,J.T.Lapreste,and G.Rives,”Determining of Attitude of 3-D Objects from a Single Perspective View”,IEEE Trans. Pattern Analysis and Machine Intelligence, vol. 11, no.12, pp.1265-1278,1989
[13] A.Gruen T.S.Huang “Calibration and Orientation of Cameras in Computer Vision”, Springer Verlag, (2001) pp. 7-51
[14] E.H.Thompson,”Space Resection: Failure Cases”, Photogram -metric Record, vol.X, no.27, pp.201-204,1966
[15] B.P. Wrobel,”Minimum Solutions for Orientations”,Proc. IEEE Workshop Calibration and Orientation Cameras in computer vision, Washington, D.C., Aug. 1992
[16] Z.Y.Hu and F.C.Wu, A Note on the Number of solutions of the non -coplanar P4P Problem. IEEE Trans. Pattern Analysis and Machine Intelligence, Vol. 24, No. 4, pp 550-555, APRIL 2002.
[17] W. Triggs. Camera pose and calibration from 4 or 5 known 3D points. In Proc. ICCV, pages 278--284, 1999.
[18] "Improper intersection of algebraic curves" Abhyankar S., Chandrasekar S., Chandru V. ACM Transactions on Graphics 9: 147-159, 1990.
[19] Gianni P., Mora T. Algebraic Solution of Systems of Polynomial Equations Using Grobner Bases. AAECC-5, Minorca. L.N.Comp.Sci 356 (1989), pp. 247-257
[20] Teo Mora, An introduction to commutative and noncommutative Groebner bases, Theoretical Computer Science 134 (1994) 134-173.
[21] Adnan Ansar and Kostas Daniilidis, “Linear Pose Estimation from points or Lines”. ,IEEE Trans. Pattern Analysis and Machine Intelligence, vol. 25, no.5, pp.578-589,2003
[22] Triggs, W. and Zisserman, A. and Szeliski, R. "Vision Algorithms: Theory and Practice", Springer Verlag, pp.298--375
[23] Th.F.Coleman and A.R.Conn,“Nonlinear programming via an exact penalty function:Global analysis”,Mathematica Programming 24(1982), pp.137-161
[24] P. E. Gill, W. Murray, and M. A. Saunders, SNOPT: An SQP algorithm for large-scale constrained optimization, Numerical Analysis Report 97-1, Department of Mathematics, University of California, San Diego, La Jolla, CA, 1997.
[25] P.Y.Papalambros and D.J. Wilde, Principles of Optimal Design, Cambridge University Press, New York, 1988