Published in final edited form as:
J Comb Optim. 2005 September ; 10(2): 113–132.

Efficient Algorithms for Determining 3-D Bi-Plane Imaging Geometry[*]

Jinhui Xu[†], **Guang Xu**[†], **Zhenming Chen**[†], and **Vikas Singh**[†]
Department of Computer Science and Engineering, State University of New York at Buffalo, 201 Bell Hall, Buffalo, NY 14260, USA, jinhui@cse.buffalo.edu, guangxu@cse.buffalo.edu, zchen4@cse.buffalo.edu, vsingh@cse.buffalo.edu

Kenneth R. Hoffmann
Department of Neurosurgery, State University of New York at Buffalo, 445 Biomedical Research Building, 3435 Main St., Buffalo, NY 14214, USA, kh9@acsu.buffalo.edu

Abstract

Biplane projection imaging is one of the primary methods for imaging and visualizing the cardiovascular system in medicine. A key problem in such a technique is to determine the imaging geometry (i.e., the relative rotation and translation) of two projections so that the interested 3-D structures can be accurately reconstructed. Based on interesting observations and efficient geometric techniques, we present in this paper new algorithmic solutions for this problem. Comparing with existing optimization-based approaches, our techniques yield better accuracy, have bounded execution time, and thus are more suitable for on-line applications. Our techniques can easily detect outliers to further improve the accuracy.

1 Introduction

Cardiovascular diseases (such as vessel blockage and narrowing) have been major causes of death in the United States for a long time. Effective treatment and diagnosis procedures for this type of diseases heavily rely on accurate 3-D images of the interested vessel structure. Tomographic techniques, such as magnetic resonance imaging (MRI) and computed tomography (CT), although generate 3-D data sets for visualization and analysis (e.g., rendering), cannot yet provide the time resolution necessary for interventions and for dynamic evaluations of rapidly moving structures such as heart. Thus, projection imaging generated by image intensifier-TV (II-TV) systems is still the dominant form of imaging in this area, mainly due to its rapid image acquisition and relatively larger field of view.

In projection imaging, 3-D structures are reconstructed from one or more 2-D projections. A key problem in such reconstructions is to determine the relative translation and rotation, called *imaging geometry*, of the coordinate system associated with one projection with respect to those associated with other projections. Since the size of vessels in a vascular system is very small, inaccurate imaging geometry may cause the reconstructed 3-D vessels located in totally different positions.

As a promising approach, bi-plane imaging (in which each 3-D image is reconstructed from two projections) has received considerable attention in recent years and a number of techniques have been developed for its imaging geometry determination and 3-D reconstruction [5,6,11,

---

[*] This research was supported in part by NIH under USPHS grant HL52567.

[†] The research of this author was supported in part by an IBM faculty partnership award, and an award from NYSTAR (New York state office of science, technology, and academic research) through MDC (Microelectronics Design Center).

12,16,21]. A common approach for determining image geometry in these techniques is to first identify a set of corresponding pairs of points in the two projections, then convert the problem to a certain non-linear optimization problem, and find a feasible solution by using greedy algorithms or general optimization packages. Due to their heuristic nature, these approaches guarantee neither the quality of solutions nor the time efficiency, and thus may not be suitable for online applications.

We notice that in the field of Computer Vision, a similar problem, called *Epipolar Geometry Determination (or EGD)*, has been studied extensively [14,18,19,22]. The Epipolar Geometry Determination problem tries to determine the epipolar geometry relating two images of a single scene and plays an important role in many applications, such as scene modeling and vehicle navigation. Quite a few interesting approaches were previously proposed for solving this problem. (Excellent surveys on existing techniques can be found in articles [23,1,13].) However, almost all of them are based on iterative numerical computation which in general can not guarantee the converging speed and therefore are not suitable for online cardiovascular applications. Further, the EGD problem is slightly more general than our problem. In the EGD problem, the relative orientation of the two images could be arbitrary and unknown in advance. But in our problem, rough estimation of the imaging geometry of the two projections can be obtained from the imaging system. Consequently, algorithms for the EGD problem cannot fully exploit the special geometric structures of cardiovascular images.

To provide better solutions, we reduce the imaging geometry determination problem to the following geometric search problem: Given two sets of 2-D points $A = \{a_1, a_2, \cdots, a_n\}$ and $B = \{b_1, b_2, \cdots, b_n\}$ on two image screens (or planes) respectively with each pair of points a_i and b_i (called *a corresponding pair*) being the approximate projections of an unknown 3-D point p_i, also given the 3-D coordinate system of A, find the most likely position of the origin o_B and the orientation of the 3-D coordinate system of B in the coordinate system of A. It is easy to see that in an ideal situation where a_i and b_i are exact projections of p_i, it is sufficient to consider only a constant number of corresponding pairs. In practice, however, it is often difficult to find the exact corresponding pairs as the correspondence is established manually. Thus, a number of corresponding pairs are often considered in practice to ensure the accuracy.

In this paper, we present an efficient geometric approach to solve the above problem. Our approach first reduces the imaging geometry determination problem to an optimal cell search problem in an arrangement of surfaces in E^6. Based on interesting observations, we then simplify the rather complicated surfaces, which can not be analytically expressed, so that each of them can be implicitly expressed by an equation. The simplified surfaces are in general non-algebraic, indicating that directly computing the arrangement could be very challenging. To overcome this difficulty, we study the error sensitivity of each variable in the imaging geometry and use it to partition the feasible domain into smaller regions so that the topological structure of the arrangement in each region can be effectively captured by some lower dimensional (e.g., 2 or 3-D) arrangements. The curves and surfaces in these lower dimensional arrangements, although are still non-algebraic, have “nice” properties and can be used to efficiently find the optimal cell. Comparing with existing approaches, our techniques achieve better accuracy (as suggested by our experimental results) and have bounded running time. Our techniques can easily recognize outliers in A and B to further improve the accuracy.

The rest of this paper is organized as follows. Section 2 gives some preliminaries of the problem. Section 3 shows that the imaging geometry problem can be reduced to solving an arrangement search problem in E^6 space. In Section 4, we discuss the major difficulties of the arrangement search approach and our main ideas for overcoming them. In Section 5, we present our algorithms for finding the maximum point in lower dimensional arrangements. Experimental results and comparison with existing approach are given in Section 6.

2 Preliminaries

In bi-plane imaging, two projections of the interested 3-D object are generated by shooting two radiation (e.g., X-ray) beams (with cone shapes) from different angles. The radiation beams penetrate the 3-D object and form two planar images on an image acquisition device (or screen) which is away from the beam source by a fixed distance *D*. For each projection, a 3-D coordinate system *xyz* can be defined in such a way that the origin is coincident with the beam source, the *z* axis is orthogonal to the image screen, and the *x* and *y* axes are horizontal and vertical, respectively (see Figure 1 for an example). Each image on a screen can also be associated with another coordinate system *uvw*, with the *u*, *v*, and *w* axes parallel to the *x*, *y*, and *z* axes, respectively, and the origin being the intersection of the *z* axis and the image screen. It is easy to see that the coordinate system *uvw* can be related to the *xyz* coordinate system through the following system.

$$\begin{pmatrix} u \\ v \end{pmatrix} = \begin{pmatrix} x \times D / z \\ y \times D / z \end{pmatrix} \tag{1}$$

To distinguish the two projections, we denote them as P_A and P_B, respectively, with P_A corresponding to the point set *A* and P_B corresponding to *B*. Their associated image screens are denoted as S_A and S_B, respectively. We call the coordinate systems associated with P_A as *xyz* and *uvw*, and the ones of P_B as $x'y'z'$ and $u'v'w'$. The relation between the coordinate system *xyz* and the system $x'y'z'$ can be specified as follows by the rotation matrix *R* and the translation vector *t*,

$$\begin{pmatrix} x' \\ y' \\ z' \end{pmatrix} = \begin{pmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{pmatrix} \begin{pmatrix} x \\ y \\ z \end{pmatrix} + \begin{pmatrix} t_x \\ t_y \\ t_z \end{pmatrix} \tag{2}$$

To facilitate our computation and analysis, we use Euler angles to describe the relative rotation in the rotation matrix *R*. Let θ, ψ and φ be the three Euler angles (see Figure 2). The entries of the rotation matrix *R* can be represented by θ, ψ and φ as follows.

$$r_{11} = -\sin\psi\sin\varphi + \cos\theta\cos\psi\cos\varphi, \quad r_{12} = \sin\theta\cos\varphi, \quad r_{13} = -\cos\psi\sin\varphi - \cos\theta\sin\psi\cos\varphi,$$
$$r_{21} = -\sin\theta\cos\psi, \quad r_{22} = \cos\theta, \quad r_{23} = \sin\theta\sin\psi,$$
$$r_{31} = \sin\psi\cos\varphi + \cos\theta\cos\psi\sin\varphi, \quad r_{32} = \sin\theta\sin\varphi, \quad r_{33} = \cos\psi\cos\varphi - \cos\theta\sin\psi\sin\varphi.$$

Due to a variety of reasons (such as the movement of beam sources and data noise), the exact rotation matrix *R* and translation vector *t* between the two coordinate systems *xyz* and $x'y'z'$ are often unknown. Rough estimation can be obtained from the imaging system or computed by using technique in [16]. To accurately reconstruct the 3-D structures of small vessels, high precision imaging geometry is desired.

3 From Imaging Geometry to Arrangement Search

In this section, we show how to reduce the imaging geometry determination problem to an optimal cell search problem in an arrangement of surfaces.

Let $P = \{p_1, p_2, \cdots, p_n\}$ be the set of to-be-determined 3-D points. Let $p^a{}_i$ and $p^b{}_i$ be the exact projections (under the optimal or most likely imaging geometry) of p_i on the image screens S_A and S_B, respectively. We define

$$\Delta = max_{i=1}^{n} max\left\{dist(a_i, p_i^a), \ dist(b_i, p_i^b)\right\},$$

where *dist*(.) is the Euclidean distance of two points. Note that $p^a{}_i$, $p^b{}_i$ and Δ are all unknowns.

To determine the best possible imaging geometry G for the point set B in the coordinate system xyz, we first guess a possible value, say δ, for Δ. Clearly, if $\delta \geq \Delta$, then there exists an imaging geometry G such that each $p^a{}_i$ will be contained in the disk d_i (on S_A) of radius δ centered at a_i. Thus p_i is contained in the round cone C_i apexed at the origin o_A and with d_i as the base (see Figure 3). Under the same imaging geometry G, the projection of C_i on the screen S_B of B forms a sector SC_i.

Observe that if G is optimal, then each b_i will fall in its corresponding sector SC_i. Thus, by counting the number (called *fall-in number* and denoted by $f_{in}(A, B, G, \delta)$) of points in B which are contained in their corresponding sectors, we are able to measure the quality of G. We say an imaging geometry G is *feasible* with respect to (w.r.t.) δ, if $f_{in}(A, B, G, \delta) = n$. For a given δ, if there exists at least one feasible imaging geometry, then δ is called *feasible*. Notice that for each feasible δ, we may have infinitely many feasible solutions to G. Thus, to determine the most likely imaging geometry G for B, we need not only to find a feasible solution to G, but more importantly to minimize δ, as a minimized δ could consequently make G converge to its optimum.

Hence, to efficiently determine the imaging geometry, three problems need to be considered: (a) How to minimize δ; (b) How to determine the feasibility of δ; (c) How to find a feasible G w.r.t. a given δ.

For (a), since the feasibility of δ is monotone in the increasing direction of δ, we can perform a binary search or a parametric search [15] on δ to find the smallest feasible δ, provided that we can determine the feasibility of G w.r.t. a fixed δ. For (b) and (c), we notice that for a fixed δ, to determine the feasibility of δ and to find a feasible G w.r.t. δ, it is sufficient to find an imaging geometry G which maximizes the value of $f_{in}(A, B, G, \delta)$. Hence, our focus is hereafter on this maximization problem.

Consider an arbitrary point $b_i \in B$. Let $o^b{}_A$ be the projection of the origin o_A on screen S_B, and $\overrightarrow{o^b_A b_i}$ be the ray emitting from $o^b{}_A$ and crossing b_i. Let α_{b_i} be the angle between $\overrightarrow{o^b_A b_i}$ and the horizontal line (i.e., the v'-axis). Denote the lower and upper bounding rays of SC_i by $r^l{}_i$ and $r^u{}_i$, respectively. Each of the two bounding rays also forms an angle with the horizontal line, and is denoted by $\alpha^l{}_i$ and $\alpha^u{}_i$, respectively. In order for b_i to be contained in its corresponding sector SC_i (i.e., b_i contributes a “1” to $f_{in}(A, B, G, \delta)$), G must be in some positions so that α_{b_i} is between $\alpha^l{}_i$ and $\alpha^u{}_i$. Since both $r^l{}_i$ and $r^u{}_i$ can be parameterized by the six variables of G, the constraints on the three angles define a (possibly unbounded) region R_i for G in E^6 so that when G is inside R_i, sector SC_i contains b_i. Thus, we can define in total n regions with each corresponding to a point in B. To maximize the value of $f_{in}(A, B, G, \delta)$, it is sufficient to determine a point (called *maximum point*) for G in E^6 contained by the most number of R_i’s.

To find such a maximum point, we need to first determine the bounding surface of each R_i. Notice that the region R_i can be viewed as the loci of G in E^6 while moving b_i inside the sector SC_i, and similarly the bounding surfaces of R_i can be viewed as the loci of G while moving b_i on the two bounding rays, $r^l{}_i$ and $r^u{}_i$, of SC_i. Thus, the formula of the surface can be determined by using the fact that b_i is incident to either $r^l{}_i$ or $r^u{}_i$.

Once all the surfaces (denoted by Γ) are obtained, a direct approach for computing the maximum point of G is to first construct the arrangement $A(\Gamma)$ of Γ, and for each cell c of $A(\Gamma)$ determine the value of $f_{in}(A, B, G, \delta)$. Notice that, since all points in c are contained by the same set of regions, their fall-in numbers are the same. Thus it is sufficient to consider only one point from each cell. The maximum point of G can then be determined by finding the cell with the maximum fall-in number.

4 Main Difficulties and Ideas

To make the above arrangement-search approach work is actually quite difficult. The success of this approach relies on two key conditions: (i) The intersections of surfaces should be easily computed; (ii) The topological structure of the arrangement should be "simple" in a sense that all the cells can be relatively easily detected. Unfortunately, neither one seems to be true. The main reason is that the surfaces of R_i are determined by the two bounding rays, r^l_i and r^u_i, which are projections of a pair of rays on the boundary of the round cone C_i. When G moves in its domain, the two rays, say rc^l_i and rc^u_i, on C_i generating r^l_i and r^u_i, respectively, will change accordingly. Thus, to compute r^l_i and r^u_i, we need to first determine rc^l_i and rc^u_i, and then project them to S_B to obtain r^l_i and r^u_i. But determining rc^l_i and rc^u_i has to find roots of a polynomial of degree 6, which in general do not admit analytical solutions. Thus, the surfaces of R_i cannot be analytically expressed by the six variables in G. Consequently, the intersections of surfaces and the arrangement cannot be efficiently computed.

To overcome this difficulty, we observe that the problem is mainly caused by determining the rays (i.e., rc^l_i and rc^u_i) on C_i which generate r^l_i and r^u_i, respectively. Our idea is thus to approximate each round cone C_i by a convex facet cone FC_i with k facets for some small constant k (e.g., 3, 4, 6) so that we only need to consider k rays on C_i as candidates for rc^l_i and rc^u_i. Depending on the location of G, the projection of FC_i will create up to $O(k)$ sectors [1], $SC^1_i, SC^2_i, \cdots, SC^{O(k)}_i$, on the screen of S_B, with each SC^j_i, $1 \leq j \leq O(k)$, corresponding to and generated by a pair of edges on FC_i tangent to two planes crossing o_B. The facet cone FC_i also partitions each region R_i into $O(k)$ subregions $R^1_i, R^2_i, \cdots, R^{O(k)}_i$, with each subregion R^j_i generated by a sector SC^j_i, $1 \leq j \leq O(k)$. Since the bounding rays of each sector SC^j_i is simply the projections of a pair of fixed edges on the facet cone, the surfaces of each R^j_i can be directly determined and implicitly expressed by an equation. A typical form of the surfaces corresponding to b_i is $f = 0$, where f is the angular distance (i.e., angle) between ray $\overrightarrow{o_A^b b_i}$ and one of the two bounding rays, r^l_i and r^u_i, of a sector SC_i, and has the following form.

$$f = \frac{\dfrac{r_{21}(u_{a_i} \pm \delta) + r_{22}(v_{a_i} \pm \delta) + r_{23}D + t_y}{r_{31}(u_{a_i} \pm \delta) + r_{32}(v_{a_i} \pm \delta) + r_{33}D + t_z} - \dfrac{t_y}{t_z}}{\dfrac{r_{11}(u_{a_i} \pm \delta) + r_{12}(v_{a_i} \pm \delta) + r_{13}D + t_x}{r_{31}(u_{a_i} \pm \delta) + r_{32}(v_{a_i} \pm \delta) + r_{33}D + t_z} - \dfrac{t_x}{t_z}} - \frac{v'_{b_i} - t_y D / t_z}{u'_{b_i} - t_x D / t_z}. \qquad (3)$$

Replacing the round cones with facet cones although simplifies the surfaces (called *bounding surfaces*) of the regions corresponding to points in B, it introduces other difficulties for searching the arrangement. For example, each region R_i is now partitioned into $O(k)$ subregions R^j_i by a set of surfaces, called *separating surfaces*, corresponding to the bounding rays for the $O(k)$ sectors. The separating surfaces are generated by comparing the angles of the projections of the k edges of FC_i on the screen S_B, and have much more complicated form than the bounding surfaces, thus dramatically increasing the difficulty of constructing the arrangement. In next section, we will show that by using a different way to count the fall-in number for each cell in the arrangement, we can actually remove the set of separating surfaces. Thus we can focus on how to efficiently construct the arrangement of the bounding surfaces.

---

[1] Note that due to the fact that the beam source o_B is often very far away from the 3-D points in P, the number of sectors is $O(k)$ instead of $O(k^2)$.

Notice that all bounding surfaces are non-algebraic and have 6 tightly coupled variables. Directly constructing the arrangement of those implicitly expressed surfaces in E^6 would be very difficult. One reason is that finding the intersections of these surfaces needs to solve high degree non-linear systems. Another reason is that the topological structure of the arrangement is rather complicated and thus it is challenging to find all cells of the arrangement.

To further simplify the problem, we study the error sensitivity of each variable in the imaging geometry. The following lemma shows that when the 3-D object is roughly in the middle of the image systems (which is typically the case in practice), error is much less sensitive to the three translational variables than to the rotational variables.

Lemma 1

Let p be any point with coordinates $(x, y, z)^T$ *and* $(x', y', z')^T$, *satisfying z and* $z' \in [\frac{D}{4}, \frac{3D}{4}]$, $|x|$, $|x'|$, $|y|$ *and* $|y'| \le \epsilon_0$ *for some small constant* ϵ_0. *Assume that the xyz and x′y′z′ coordinate systems have the following relation*: $\theta \le \epsilon_1$, ψ *and* $\varphi \in [\pi/4, 3\pi/4]$, $\varphi + \psi \in [\pi/4, 3\pi/4]$, $t_x, t_z \in [D/4, 3D/4]$, *and* $|t_y| \le \epsilon_0$, *where* ϵ_1 is another small constant. Then the partial derivatives of the angular distance f w.r.t. each variable has the following orders:

$$\frac{\partial f}{\partial t_x} = O(\frac{1}{D^2}), \quad \frac{\partial f}{\partial t_y} = O(\frac{1}{D}), \quad \frac{\partial f}{\partial t_z} = O(\frac{1}{D^2}), \quad \frac{\partial f}{\partial \theta} = O(1), \quad \frac{\partial f}{\partial \varphi} = O(\frac{1}{D}), \quad \frac{\partial f}{\partial \psi} = O(\frac{1}{D}),$$

where $D >> \epsilon_0 + \epsilon_1$ *is the distance between the origins of the coordinate systems image screens*.

Proof—First, we list the inequalities obtained directly from the assumption.

$$D/4 \le t_x \le 3D/4 \tag{4}$$

$$|t_y| \le \epsilon_0 \tag{5}$$

$$D/4 \le t_z \le 3D/4 \tag{6}$$

$$|u'_{b_i}| \le \epsilon_0 \tag{7}$$

$$|v'_{b_i}| \le \epsilon_0 \tag{8}$$

Next, we introduce three additional variables to simplify our proof. Let

$$\begin{aligned} T_1 &= r_{11}(u_{a_i} \pm \delta) + r_{12}(v_{a_i} \pm \delta) + r_{13}D + t_x, \\ T_2 &= r_{21}(u_{a_i} \pm \delta) + r_{22}(v_{a_i} \pm \delta) + r_{23}D + t_y, \\ T_3 &= r_{31}(u_{a_i} \pm \delta) + r_{32}(v_{a_i} \pm \delta) + r_{33}D + t_z. \end{aligned}$$

Let p^a and p^b be the projections of p on screens S_A and S_B, respectively. Notice that T_1 is the x' coordinate of p^a in the $x'y'z'$ coordinate system. From the assumption about the position of p, we have

$$-3D/4 \le T_1 \le -D/4. \tag{9}$$

Since T_2 is the y' coordinate of p^a in the $x'\,y'\,z'$ coordinate system. From the assumption about the position of p, we have

$$\left| T_2 \right| \leq 4\epsilon_0 \ . \tag{10}$$

T_3 is the z' coordinate of p^a, hence

$$D \Big/ 4 \leq T_3 \in 3D \Big/ 4. \tag{11}$$

We begin with estimating $\frac{\partial f}{\partial t_x}$.

$$\frac{\partial f}{\partial t_x} = \frac{\frac{T_2}{T_3} - \frac{t_y}{t_z}}{\left(\frac{T_1}{T_3} - \frac{t_x}{t_z}\right)^2}\left(\frac{1}{T_3} - \frac{1}{t_z}\right) - \frac{v'_{b_i} - t_y D \Big/ t_z}{\left(u'_{b_i} - t_x D \Big/ t_z\right)^2}\frac{D}{t_z}. \tag{12}$$

From equations (4)–(11), the following estimations about the terms appeared in the expression of $\frac{\partial f}{\partial t_x}$ can be easily obtained.

$$\left| \frac{T_2}{T_3} \right| \leq \frac{4\epsilon_0}{D/4} \leq \frac{16\epsilon_0}{D}, \quad \left| \frac{t_y}{t_z} \right| \leq \frac{\epsilon_0}{D/4} \leq \frac{16\epsilon_0}{D}, \quad -3 \leq \frac{T_1}{T_3} \leq -1 \Big/ 3, \text{ and } 1 \Big/ 3 \leq \frac{t_x}{t_z} \leq 3$$

. Plugging these estimations into (12), we have

$$\left| \frac{\partial f}{\partial t_x} \right| \leq \frac{48\epsilon_0}{D}\frac{3}{2D} + \frac{5\epsilon_0}{(D/3 - \epsilon_0\ )^2}4.$$

From the assumptions about ϵ_0 and D, we have $D/6 \leq D/3 - \epsilon_0 \leq D/3$. Thus, we obtain the desired estimation of $\frac{\partial f}{\partial t_x}$, i.e.,

$$\left| \frac{\partial f}{\partial t_x} \right| \leq \frac{800\epsilon_0}{D^2} = O\left(\frac{1}{D^2}\right).$$

For $\frac{\partial f}{\partial t_y}$, we have the following equation.

$$\frac{\partial f}{\partial t_y} = \frac{\frac{1}{T_3} - \frac{1}{t_z}}{\frac{T_1}{T_3} - \frac{t_x}{t_z}} - \frac{\frac{D}{t_z}}{u'_{b_i} - \frac{t_x D}{t_z}}. \tag{13}$$

Inserting inequalities (4)–(11) into (13), we have $\left| \frac{\partial f}{\partial t_y} \right| = O\left(1 \Big/ D\right)$.

For $\frac{\partial f}{\partial t_z}$, we have

$$\frac{\partial f}{\partial t_z}=\frac{t_y \Big/ t_z^2-T_2 \Big/ T_3^2}{T_1 \Big/ T_3-t_x \Big/ t_z}-\frac{T_2 \Big/ T_3-t_y \Big/ t_z}{(T_1 \Big/ T_3-t_x \Big/ t_z)^2}\left(t_x \Bigg/ t_z^2-T_1 \Bigg/ T_3^2\right)-\frac{\frac{t_yD}{t_z^2}}{u' b_i-\frac{t_xD}{t_z}}$$

$$+\left(v' b_i-t_yD \Bigg/ t_z\right)\frac{1}{(u' b_i-t_xD \Big/ t_z)^2}\frac{t_xD}{t_z^2}.$$

Inserting inequalities (4)–(11) into the above equation, we have $\left|\frac{\partial f}{\partial t_z}\right| = O(1 \Big/ D^2)$.

For the rotational variables, we first estimate $\frac{\partial f}{\partial \varphi}$. From Equation (3), we have the following equation.

$$\frac{\partial f}{\partial \varphi}=\frac{\frac{\partial T_2}{\partial \varphi}\frac{1}{T_3}-\frac{T_2}{(T_3)^2}\frac{\partial T_3}{\partial \varphi}}{\frac{T_1}{T_3}-\frac{t_x}{t_z}}-\frac{\frac{T_2}{T_3}-\frac{t_y}{t_z}}{(\frac{T_1}{T_3}-\frac{t_x}{t_z})^2}\left(\frac{\partial T_1}{\partial \varphi}\frac{1}{T_3}-\frac{T_1}{T_3^2}\frac{\partial T_3}{\partial \varphi}\right). \quad (14)$$

In (14), we have two new terms, $\frac{\partial T_2}{\partial \varphi}\frac{1}{T_3}-\frac{T_2}{T_3^2}\frac{\partial T_3}{\partial \varphi}$ and $\frac{\partial T_1}{\partial \varphi}\frac{1}{T_3}-\frac{T_1}{T_3^2}\frac{\partial T_3}{\partial \varphi}$, to estimate.

$$\frac{\partial T_2}{\partial \varphi}=\frac{\partial r_{21}}{\partial \varphi}(u_{a_i}\pm\delta)\frac{\partial r_{22}}{\partial \varphi}(v_{a_i}\pm\delta)+\frac{\partial r_{23}}{\partial \varphi}D.$$

From the definition of r_{21}, r_{22} and r_{23}, we have $\left|\frac{\partial r_{21}}{\partial \varphi}\right| \leq 3$ $\left|\frac{\partial r_{22}}{\partial \varphi}\right| \leq 3$, and $\frac{\partial r_{23}}{\partial \varphi} = 0$. Hence,

$$\left|\frac{\partial T_2}{\partial \varphi}\right| \leq 6(\epsilon_0+\delta).$$

We also have

$$\left|\frac{\partial T_3}{\partial \varphi}\right| \leq 2D+6(\epsilon_0+\delta), \quad \text{and}$$

$$\left|\frac{\partial T_1}{\partial \varphi}\right| \leq 2D+6(\epsilon_0+\delta).$$

$$\text{Thus,} \quad \left|\frac{\partial T_2}{\partial \varphi}\frac{1}{T_3}-\frac{T_2}{T_3^2}\frac{\partial T_3}{\partial \varphi}\right| \leq C\left(\frac{\epsilon_0+\delta}{D}\right)+\frac{\epsilon_0}{D^2}(\epsilon_0+\delta+D)=O(1 \Bigg/ D).$$

Similarly, we can estimate $\frac{\partial T_1}{\partial \varphi}\frac{1}{T_3}-\frac{T_1}{T_3^2}\frac{\partial T_3}{\partial \varphi}$ and have $\frac{\partial f}{\partial \varphi}=O(1 \Big/ D)$.

For $\frac{\partial f}{\partial \psi}$, we have

$$\frac{\partial f}{\partial \psi} = \frac{\frac{\partial T_2}{\partial \psi}\frac{1}{T_3} - \frac{T_2}{T_3^2}\frac{\partial T_3}{\partial \psi}}{\frac{T_1}{T_3} - \frac{t_x}{t_z}} - \frac{\frac{T_2}{T_3} - \frac{t_y}{t_z}}{(\frac{T_1}{T_3} - \frac{t_x}{t_z})^2}(\frac{\partial T_1}{\partial \psi}\frac{1}{T_3} - \frac{T_1}{T_3^2}\frac{\partial T_3}{\partial \psi}).$$

We get

$$\frac{\partial f}{\partial \psi} = O(1 / D).$$

The estimation of $\frac{\partial f}{\partial \theta}$ is somewhat different. First,

$$\frac{\partial f}{\partial \theta} = \frac{\frac{\partial T_2}{\partial \theta}\frac{1}{T_3} - \frac{T_2}{T_3^2}\frac{\partial T_3}{\partial \theta}}{\frac{T_1}{T_3} - \frac{t_x}{t_z}} - \frac{\frac{T_2}{T_3} - \frac{t_y}{t_z}}{(\frac{T_1}{T_3} - \frac{t_x}{t_z})^2}(\frac{\partial T_1}{\partial \theta}\frac{1}{T_3} - \frac{T_1}{T_3^2}\frac{\partial T_3}{\partial \theta}).$$

By previous estimations, we have

$$\frac{\frac{T_2}{T_3} - \frac{t_y}{t_z}}{(\frac{T_1}{T_3} - \frac{t_x}{t_z})^2}(\frac{\partial T_1}{\partial \theta}\frac{1}{T_3} - \frac{T_1}{T_3^2}\frac{\partial T_3}{\partial \theta}) \le C\frac{\epsilon_0}{D^2}.$$

Since the dominating term of $\frac{\partial T_2}{\partial \theta}\frac{1}{T_3} - \frac{T_2}{T_3^2}\frac{\partial T_3}{\partial \theta}$ is $\frac{\partial T_2}{\partial \theta}\frac{1}{T_3}$, and $\frac{\partial T_2}{\partial \theta}\frac{1}{T_3} = O(1 / D)$, we obtain

$$\frac{\partial f}{\partial \theta} = O(1).$$

Remark—In the above lemma, all the assumptions on point *p* and on the relation of the two coordinate systems hold in practice.

The above lemma shows that when *p* is well placed (i.e., in the middle of the two projections), it is much easier for p^b to move from inside to outside or from outside to inside of its corresponding sector *SC* on S_B, when *G* moves in the directions of the variables with larger partial derivatives (such as θ t_y, ψ, and φ) than it is in other directions This means that the topological structure of the arrangement is more likely to change (i.e., crossing some bounding surfaces) when *G* moves one unit of distance in these directions. Consequently, an error in the computation of their corresponding variables is more likely to lead to an larger error for the imaging geometry and 3-D reconstruction. Thus the partial derivatives of these varibales can be viewed as a measure of their error sensitivities.

To compute the maximum point of *G*, we notice that we only need to find one point from each possible cell in the arrangement. Thus, it is sufficient to consider a set of crossing sections (i.e., lower dimensional arrangements) of the 6-dimensional arrangement as long as the set of crossing sections intersects every cell of the arrangement. For an error insensitive direction (or variable), since the topological structure of the arrangement has less changes (within a unit distance), we may select a few observing points and compute the crossing sections through the selected points (i.e., using hyperplanes crossing these observing points and orthogonal to this direction to cut the arrangement). In this way, we may avoid considering this direction continuously, and hence reduce the dimensions of the arrangement.

To implement this idea, we can select a few “good” variables with larger partial derivatives as the variables of the arrangement, and place a grid in the subspace of the domain induced by those unselected variables. We say a set of variables are good if the bounding surfaces induced by setting other variables to constants has simple forms or nice structures. Notice that in the imaging geometry determination problem, the domain can be assumed to be a small hyperbox as a rough estimation of the optimal solution can be obtained by using some previous techniques [16] or from the imaging system. The sizes of the grid may vary in different directions, consistent with their partial derivatives.

Hence it is possible to compute the maximum point through traversing a set of lower dimensional arrangements. Following is the main steps of our algorithm for computing the maximum point.

1. Select 2 or 3 good variables, and place a grid in the subspace of the domain induced by other variables.
2. For each grid point,
    a. determine the expression of each bounding surface in the subspace induced by the selected variables;
    b. compute the maximum point in the arrangements of the set of lower dimensional surfaces.
3. Return the maximum point found among all grid points.

5 Finding the Maximum Point in a Lower Dimensional Arrangement

To solve the maximum point problem, we need to first select the set of variables. By Lemma 1, we know that θ is the most error-sensitive variable, and thus should be chosen. Three other variables, t_y, ψ, and φ has the same order. Since t_y is loosely coupled with θ, we pick t_y over the other two rotational variables.

To select other possible variables, we first observe that if two rotational variables are selected simultaneously, the surfaces will be of the form $\frac{g_1(\alpha_1,\ \alpha_2)}{g_2(\alpha_1,\ \alpha_2)} + c = 0$, where, α_1 and α_2 are the two rotational variables, and $g_1()$, $g_2()$ are two functions containing products of trigonometric functions of α_1 and α_2. The surfaces will be rather complicated, and more importantly, their intersections will not be easily computed. Hence, in our algorithm, we only select one rotational variable.

Since our approach needs to traverse an arrangement on each grid point, the more variables we select, the less grid points we need to consider. Thus on one hand, it is better to select more variables to ensure time efficiency as long as the associated arrangements can still be easily traversed. On the other hand, selecting more variables will complicate the surfaces and make it harder to traverse the arrangement. Nevertheless, our analysis shows that it is possible to select another translational variable t_x and still obtain relatively simple surfaces.

With the selected variables, at each grid point the bounding surfaces have the following form.

$$\begin{aligned} &t_x\left\{v'_{b_i}\left(r_{31}(u_{a_i} \pm \delta) + r_{32}(v_{a_i} \pm \delta) + r_{33}D\right) - D\left(r_{21}(u_{a_i} \pm \delta) + r_{22}(v_{a_i} \pm \delta) + r_{23}D\right)\right\} + \\ &t_y\left\{D\left(r_{11}(u_{a_i} \pm \delta) + r_{12}(v_{a_i} \pm \delta) + r_{13}D\right) - u'_{b_i}\left(r_{21}(u_{a_i} \pm \delta) + r_{22}(v_{a_i} \pm \delta) + r_{23}D\right)\right\} = \\ &t_z\left\{v'_{b_i}\left(r_{11}(u_{a_i} \pm \delta) + r_{12}(v_{a_i} \pm \delta) + r_{13}D\right) - u'_{b_i}\left(r_{31}(u_{a_i} \pm \delta) + r_{32}(v_{a_i} \pm \delta) + r_{33}D\right)\right\}. \end{aligned} \tag{15}$$

In the above equation, those unselected variables are treated as constants. The following lemmas show some nice properties of such surfaces.

Lemma 2

Let t_x, t_y and $\alpha \varepsilon \{\theta, \psi, \varphi\}$ be the three selected variables. Then, at any fixed grid point, each bounding surface S_i is monotone in the directions of t_x and t_y. Furthermore, the intersection of S_i and any plane parallel to the $t_x t_y$ plane is a straight line.

Proof—Consider equation (15). Since there is only one selected rotational variable, (15) can be rewritten as

$$c_0(\alpha)t_x + c_1(\alpha)t_y = c_2(\alpha)t_z, \tag{16}$$

where $c_i(\alpha) = m_i \cos\alpha + n_i \sin\alpha + l_i$, $i = 0, 1, 2$ and m_i, n_i, l_i are constants depending on the three fixed variables and the corresponding pair α_i and b_i. If α is fixed, the equation becomes a linear equation of t_x and t_y. Thus, the intersection of S_i and any plane parallel to $t_x t_y$ plane is a straight line. For each given value of t_y and α , the equation becomes a linear equation of t_x. Thus, there is a unique solution to t_x, and the surface is t_x-monotone. Similarly, we can show that the surface is t_y-monotone.

Remarks—Although we select θ as the rotational variable, the above lemma holds for any rotational variable. From Equation (16), we know that once the three rotational variables are fixed, the three translational variables are linearly correlated. Thus, to find the maximum point, we only need to place grid points in the subspace induced by the two non-selected rotational variables.

Lemma 3

Let t_x, t_y and α be defined as in Lemma 2. Then each bounding surface S_i is continuous in the directions of t_x, t_y and α .

Proof—From Equation (16), we know that for each given α , the surface is a straight line and thus is continuous in the directions of t_x and t_y. Since the coefficients of Equation (16) are all continuous functions of α , S_i is continuous at each α.

The above lemmas are keys for us to design efficient algorithms for computing a maximum point in the arrangement.

Recall that to find a maximum point in the arrangement $A(\Gamma)$, we can first use the technique in [16] (or information from the imaging system) to obtain an approximation of G so that the optimal solution to G is contained in an axis-aligned hyperbox H in E^6 space. Thus our search for the maximum point can be focused on the portion of $A(\Gamma)$ inside H. At each grid point, the three unselected variables become constants, and the hyperbox H is reduced to a 3-D axis-aligned box. Without causing any ambiguity, we also call the 3-D box as H, the set of bounding surfaces as Γ and the arrangement as $A(\Gamma)$.

Hence our task is to find the maximum point in $A(\Gamma) \cap H$. As mentioned previously, all points in each cell of $A(\Gamma)$ share the same fall-in number. For two neighboring cells c_1 and c_2 separated by a bounding surface S_i, the two sets of fall-in points (i.e., the points of B contained in their corresponding sectors) in the two cells differ only by one point b_i, since crossing the surface S_i means turning b_i from a fall-in point to a non-fall-in point (or vice versa). Hence the difference of the fall-in numbers of the two cells is 1. To find the cell with the maximum fall-in number, our main idea is to design a plane sweep algorithm which extracts one or more points from each cell and efficiently determine their fall-in numbers.

To better illustrate our plane sweep algorithm, we first assume that there are only two bounding surfaces generated from each facet cone FC_i with each S_i corresponding to a bounding ray of the sector SC_i. (Later on we will show how to remove this assumption.) Thus in the arrangement $A(\Gamma)$, when G crosses S_i, its fall-in number either increases or decreases by 1. Equivalently, each surface S_i can be viewed as an oriented surface. When G crosses S_i in the direction of its orientation, the fall-in number of G increases by 1.

To efficiently search all cells in $A(\Gamma) \cap H$, we sweep a plane P parallel to the $t_x t_y$-plane through H. P starts from the bottom of H and moves in the increasing direction of θ (see Figure 5). Let $[\theta_{0,} \theta_1]$ be the range of θ in H, and let P_{θ} be the intersection of P and $A(\Gamma)$ n H when P moves to the position of θ. By Lemma 2, we know that the intersections of Γ and P are a set of lines. Hence P_θ is the portion of an arrangement of lines inside a rectangle. The following lemma shows that the fall-in number of each cell in P_{θ_0} can be efficiently computed.

Lemma 4

The fall-in number of each cell in P_{θ_0} *can be computed in* $O(n \log n + K_0)$ *time, where* K_0 *is the number of cells in* P_{θ_0}.

Proof—Since P_{θ_0} is the portion of an arrangement of lines inside a convex polygon (e.g., rectangle), we can use topological peeling [4] (or topological walk [3]) to generate the set of cells as well as the intersections of P_{θ_0} in $O(n \log n + K_0)$ time. The fall-in number of the first cell c encountered by topological peeling can be computed by first selecting an arbitrary point in c as (a fixed) G and then checking whether each point in B is contained in its corresponding sector. The time needed for checking each point is $O(1)$ once G is fixed. Thus the total time for computing the fall-in number of the first cell is $O(n)$. For each later encountered cell (by topological peeling), we can compute its fall-in number from its neighboring cell in $O(1)$ time, since topological peeling generates cells in a wave propagation fashion. Thus the total time needed for computing all fall-in numbers is $O(n + K_0)$. Thus the lemma follows.

To compute the fall-in numbers for those cells in $A(\Gamma) \cap H$ which have not yet intersected by P, we detect all the events in which P encounters a cell or finishes a cell while moving from bottom to top. Notice that there are several types of events which could change the topological structure of P_θ: (a) A surface which is previously outside of H enters H and generates a line on $P \cap H$; (b) A surface leaves H, and hence its corresponding line on $P \cap H$ moves outside of H; (c) A new cell is encountered by P; (d) A cell is finished by P.

For type (a) and (b) events, we can compute for each surface $S_i \in \Gamma$ its intersections with the boundary of H, and insert the events into an event queue (such as priority queue) for the plane sweep. Since the intersections can be computed in constant time for each surface, and inserting each event into an event queue takes $O(\log n)$ time, the total time for type (a) and (b) events is $O(n \log n)$ time.

For type (c) and (d) events, we have the following lemma.

Lemma 5

For each cell which is not discovered by a type (a) or (b) event, and does not intersect P_{θ_0}*, its first intersection with* P *occurs at a vertex generated by three bounding surfaces or the boundary of* H.

Proof—By Lemma 2, we know that all surfaces in γ are monotone in t_x and t_y directions. Suppose there is such a cell c which is first intersected by P at an interior point on one of its bounding surfaces S_i. By Lemma 3, we know S_i is continuous. Thus if move P up slightly, say by a sufficient small constant ϵ, then S_i will generate a closed curve on P, contradicting Lemma 2.

To efficiently detect all type (c) and (d) events, we consider a type (c) event (type (d) events can be similarly handled). Let c be the cell encountered by P. By Lemma 5, the first encountered point is a vertex v of c. Let S_1, S_2 and S_3 be the three surfaces generating v. Consider the moment just before P meets v. By Lemma 3, all the three surfaces S_1, S_2 and S_3 are continuous. Thus each of them produces a line on P. The three lines generate at least two vertices on P, say v_1 and v_2, which are neighboring to each other on one of three lines and converge to v when P moves to v. Thus to detect this event, it is sufficient to compute v at the time when v_1 and v_2 becomes neighbors to each other at their first time.

To detect all such events, we can start from P_{θ_0} and compute for each pair of neighboring vertices in P_{θ_0} the time when they converge (or merge) (if they do), and store such time in the event queue if it is in the range of H. Then we use the event queue to sweep the arrangement. When a new vertex is generated on P or two vertices become neighbors to each other at their first time, we check whether there is a possible event. In this way, we can capture all the events and thus detect all the cells in $A(\Gamma) \cap H$.

The following lemma shows that each type (c) or (d) event can be detected efficiently.

Lemma 6

The intersection of three bounding surfaces can be computed by solving a polynomial of degree 6.

Proof—Followed from Equation (16) and Lemma 2.

The following lemma bounds the total time used for detecting all events.

Lemma 7

All events can be detected in $O(n \log n + K(T_6 + \log n))$ *time, where* K *is the total number of vertices in* $A(\Gamma) \cap H$*, and* T_6 *is the time needed for finding roots of a polynomial of degree 6.*

Proof—From Lemma 4, we know that the total time for constructing the arrangement P_{θ_0} is $O(n \log n + K_0)$. In P_{θ_0}, there are $O(K_0)$ pairs of neighboring vertices. Each pair of neighboring vertices takes $O(T_6)$ time (by Lemma 6) to find the possible event and $O(\log n)$ time to insert the event into the event queue. For each new vertex in $A(\Gamma) \cap H$, the plane sweep algorithm takes $O(T_6 + \log n)$ time to find the event, to insert into, and to delete from the event queue. Thus the total time for generating all vertices is $O(n \log n + K(T_6 + \log n))$, and the lemma follows.

The fall-in number of each cell c can be computed in $O(1)$ time at the moment when P intersects c at the first time by using the already computed fall-in numbers of its neighboring cells.

So far, we have assumed that each facet cone FC_i contributes only two surfaces to Γ. For a *k*-edge facet cone, it could generate *k* bounding surfaces, with each corresponding to the projection (ray) of an edge of FC_i. Let $S^j{}_i$, and $r^j{}_i$, $1 \leq j \leq k$, be the *k* surfaces and projection rays, respectively. Depending on the position of *G*, each of the *k* rays could be a bounding ray of the sector SC_i. When computing fall-in numbers, we change the fall-in number only when the surface corresponding to a bounding ray is crossed. Thus if a surface (called *shadow surface*) whose corresponding ray is not a bounding ray of SC_i is crossed, the fall-in number need not be changed. As mentioned in last section, one way to solve this problem is to introduce separating surfaces to partition each region R_i into several subregions and to consider a more complicated arrangement. A better way is to keep the *k* surfaces simultaneously in γ, and change the way of computing the fall-in number. During sweeping the arrangement, if a new cell involves some shadow surfaces, then the fall-in numbers of the two cells separated by a shadow surface should be the same. A shadow surface may become a bounding surface when its corresponding ray exchanges its role with a bounding ray. This means the two straight lines (corresponding to the two surfaces of the same facet cone) on the sweep plane *P* overlap with each other. By checking the order of the two rays on S_B, we can correctly determine which surface is now the bounding surface and its orientation.

Computing the fall-in number in this way increases the number surfaces by a factor of *k*. Thus the total time for finding the maximum point can be bounded by the following theorem.

Theorem 1

A maximum point in A(γ)∩ H can be computed by a plane sweep algorithm in $O(nk \log(nk) + K(T_6 + \log(nk)))$ *time, where k is the number of edges of a facet cone and K is the number of vertices in the arrangement A(γ)∩ H of O(nk) surfaces, and* T_6 *is the time needed for finding roots of a polynomial of degree 6.*

Proof—Followed from the above discussion.

5.1 Finding the Maximum Point in 2-D Arrangements

When the range of t_x is small, we may also consider selecting only t_y and θ as the variables so that the arrangement is further simplified. The following lemma shows some nice structure of the arrangement induced by two variables.

Lemma 8—*Let* t_a, $a \in \{x, y, z\}$ *and* $\alpha \in \{\theta, \psi, \varphi\}$ *be the two selected variables. Each bounding curve* S_i *is of the form*

$$T_a = \frac{a_i \cos(\alpha) + b_i \sin(\alpha) + c_i}{d_i \cos(\alpha) + e_i \sin(\alpha) + f_i}, \tag{17}$$

where $\alpha \in [0, 2\pi]$ *or* $[0, \pi]$, a_i, b_i, c_i, d_i, e_i *and* f_i *are constants, and can be break into up to* 3 *continuous pieces. Any pair of curves has no more than 4 intersections.*

<u>**Proof:**</u> The form of the bounding curve S_i follows from Equation (16). The denominator of the right hand side of Equation (17) has up to two roots, say α_1 and α_2, which make S_i non-continuous. By breaking S_i at these two points, S_i can be partitioned into three continuous pieces.

To determine the number of intersections between any pair of curves S_1 and S_2, we consider the following equation.

$$\frac{a_1 \cos(\alpha) + b_1 \sin(\alpha) + c_1}{d_1 \cos(\alpha) + e_1 \sin(\alpha) + f_1} = \frac{a_2 \cos(\alpha) + b_2 \sin(\alpha) + c_2}{d_2 \cos(\alpha) + e_2 \sin(\alpha) + f_2}.$$

Replacing cos(α) by z and sin(α) by $\pm\sqrt{1 - z^2}$, we can convert the above equation into a polynomial equation of degree 4. Thus the number of intersections between S_1 and S_2 is at most 4.

Lemma 8 enables us to find a maximum point at each grid point by using some 2-D arrangement traversal algorithms [2]. The following lemma bounds the running time.

Lemma 9—*The maximum point of a 2-D arrangement can be found in* $O(n \log n + K)$ *time, where K is the number of vertices in the 2-D arrangement inside H.*

Proof: By Lemma 8, we know that the pairwise intersections are no more than 4. Thus by using the algorithm in [2], we can traverse the whole arrangement in $O(n \log n + K)$ time.

To find the "global" maximum in the 2-D or 3-D arrangements, we determine a maximum point for each grid point and return the best as our solution.

5.2 Removing Outliers

After the binary search on δ has finished, the accuracy can be further improved by removing a few outliers from the point sets A and B. Notice that the correspondence between A and B are often established manually, and may not be consistent with each other. By removing a few outliers, we could further reduce δ and consequently reduce the error of G. The main idea is as follows. Once δ is reduced to be infeasible, we can find a maximum point for G, and check which points in B are not contained in their corresponding sectors. If the number of such points is small, we can just discard those non-fall-in points from A and B. By doing so, δ becomes feasible again and therefore may be further reduced. Hence, the error of G could be decreased.

6 Experimental Results

To evaluate the performance of our technique, we implemented our algorithm using C++ and LEDA 4.4 on a DELL OPTIPLEX GX260 PC with 2.26GHz running Redhat Linux 8.0, and compare it with a well-recognized algorithm [11] in Cardiovascular community. We conduct our experiment with the same configuration as those in [11]. Our experiment randomly generates a biplane imaging geometry in a small neighborhood of the following settings: $\psi = \pi/2$, $\theta = 0$, $\varphi = 0$, $|t_x| = |t_z| = 0.5D$, $t_y = 0$, $D = 140cm$; and the input noise (or image errors) for image data are up to 0.15cm. A set of 3-D object points are placed close to the center of the two coordinate systems. The object points are projected onto the screen S_A and S_B, respectively. A and B are then chosen by adding random noise to the projections of P.

Our experiment shows that the absolute errors for the translational variables and the Euler angles are either much smaller than or comparable with those in [11]. For instance, in most of our examples, errors are as small as $0.05cm$ for the translational variables and 0.5° for the Euler angles while the errors in [11] could be as much as $0.15cm$ for the translational variables and 3° for the rotational variables. Our experiments indicate that the errors are consistent with the sensitivity analysis stated in Lemma 1. As expected, Figure 6(a) shows that the errors of t_y tend to decrease when there are more corresponding pairs. Figure 6(b) shows that the errors of t_y slowly increase with an increase in the input image errors.

References

1. Aggarwal J, Nandhakumar N. "On the computation of motion from sequences of images-A review,". Proc IEEE 1988;76(8):917–935.
2. Amato NM, Goodrich MT, Ramos EA. "Computing the arrangement of curve segments: Divide-and-conquer algorithms via sampling,". Proc 11th Annual ACM-SIAM Symposium on Discrete Algorithms 2000:705–706.
3. Asano T, Guibas LJ, Tokuyama T. "Walking in an arrangement topologically,". International Journal of Computational Geometry and Applications 1994;4(2):123–151.
4. D.Z. Chen, S. Luan, and J. Xu,"Topological Peeling and Implementation," Proc. 12th Annual International Symposium on Algorithms And Computation (ISAAC), Lecture Notes in Computer Science, Vol. 2223, Springer Verlag, 2001, pp.454–466.
5. Chen SYJ, Metz CE. "Improved determination of biplane imaging geometry from two projection images and its application to three-dimensional reconstruction of coronary arterial trees,". Med Phys 1997;24:633–654. [PubMed: 9167155]
6. Esthappan J, harauchi H, Hoffmann K. "Evaulation of imaging geometries c alculated from biplane images,". Med Phys 1998;25(6):965–975. [PubMed: 9650187]
7. Fusiello A. "Uncalibrated Euclidean reconstruction: a review,". Image and Vision Computing 2000;18:555–563.
8. Grondin CM, Dyrda I, Pasternac A, Campeau L, Bourassa MG, Lesperance J. "Discrepancies between cineangiographic and postmortem findings in patients with coronary artery disease and recent myocardial revascularization,". Circulation 1974;49:703–708. [PubMed: 4544728]
9. Hoffmann KR, Doi K, Chan HP, Chua KG. "Computer reproduction of the vasculature using an automated tracking method,". Proc SPIE 1987;767:449–453.
10. Hoffmann KR, Doi K, Chan HP, Takamiya M. "Three dimensional reproduction of coronary vascular trees using the double-squre-box method of trackings,". Proc SPIE 1988;914:375–378.
11. Hoffmann KR, Metz CE, Chen Y. "Determination of 3D imaging geometry and object configurations from two biplane views: An enhancement of the Metz-Fencil technique,". Med Phys 1995;22:1219–1227. [PubMed: 7476707]
12. Hoffmann KR, Sen A, Lan L, Kok-Gee Chua, Esthappan J, Mazzucco M. "A system for determination of 3D vessel tree centerlines from biplane images,". The International Journal of Cardiac Imaging 2000;16:315–330.
13. Huang T, Netravali A. "Motion and structure from feature correspondences: A review,". Proc IEEE 1994;82(2):252–268.
14. Luong QT, Faugeras OD. "The fundamental matrix: Theory, algorithms and stability analysis,". The International Journal of Computer Vision 1996;1(17):43–76.
15. Megiddo N. "Applying parallel computation algorithms in the design of serial algorithms,". Journal of ACM 1983;30(4):852–865.
16. Metz CE, Fencil LE. "Determination of three-dimensional structure in biplane radiography without prior knowledge of the relationship between the two views,". Med Phys 1989;16:45–51. [PubMed: 2921979]
17. D. P. Nazareth, K. R. Hoffmann, A. Walczak, J. Dmochowski, L. Guterman, S. Rudin, D. R. Bednarek, "Determination of biplane geometry and centerline curvature in vascular imaging," Manuscript, 2002.
18. Quan L. "Invariants of six points and projective reconstruction from three uncalibrated images,". IEEE Transaction on Pattern Analysis and Machine Intelligence 1995;17(1)
19. Torr P, Zisserman A, Maybank S. "Robust detection of degenerate configurations for the fundamental matrix,". Proc of the 5th Int Conf on computer Vision 1995:1037–1042.
20. Vlodaver Z, Frech R, Van Tassel RA, Edwards JE. "Correlation of the antemortem coronary angiogram and the postmortem specimen,". Circulation 1973;47:162–169. [PubMed: 4686593]
21. Wahle A, Wellnhofer E, Mugaragu I, Sauer HU, Oswald H, Fleck E. "Assessment of diffuse coronary artery disease by quantitative analysis of coronary morphology based upon 3-D reconstruction from biplane angiograms,". IEEE Transactions on Medical Imaging 1995;14:230–241.

22. Zhang Z, Luong QT, Faugeras O. “Motion of an uncalibrated stereo rig: Self-calibration and metric reconstruction,”. IEEE Trans Robotics and Automation 1996;12(1):103–113.
23. Zhang Z. “Determining the Epipolar Geometry and its Uncertainty: A Review,”. International Journal of Computer Vision 27(2):161–195.

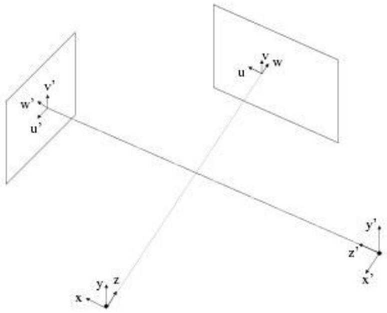


Figure 1.
Coordinate systems of bi-plane imaging.

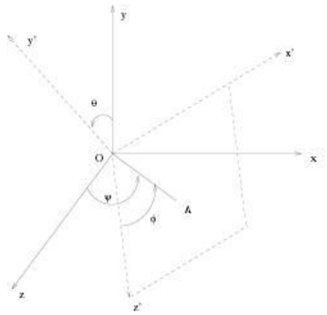


Figure 2.
The three Euler angles.

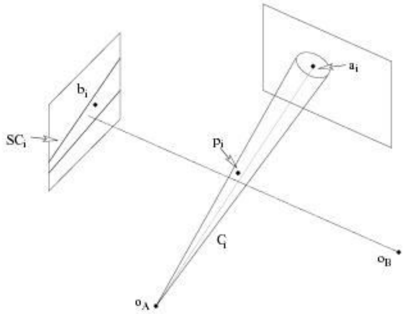


Figure 3.
A corresponding pair, a_i and b_i, and the associated round cone.

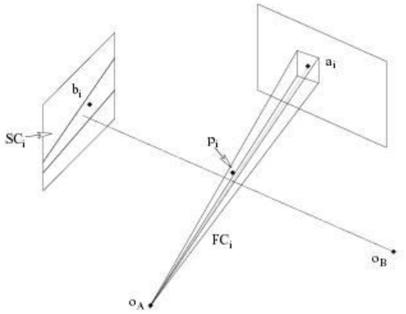


Figure 4.
Facet cone.

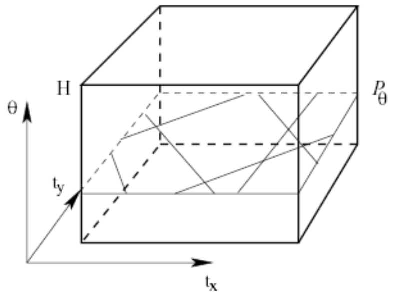
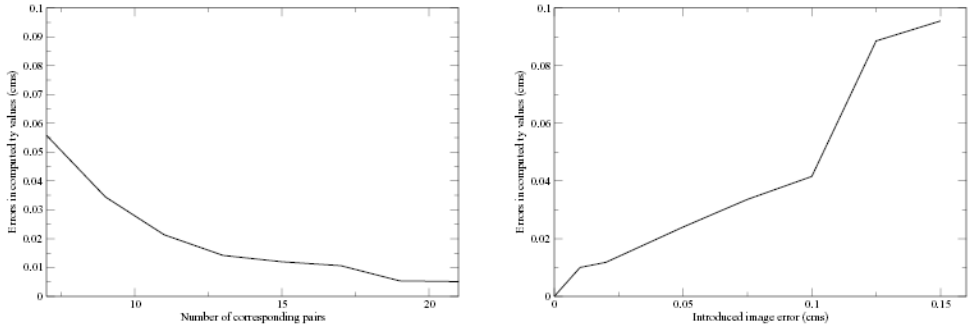


Figure 5.
Sweep the arrangement $A(\Gamma) \cap H$.



(a) (b)

Figure 6.

(a) Errors of t_y vs. number of corresponding pairs (b) Errors of t_y vs. input image errors