An Automated Matching Algorithm for Dual Orthogonal Fluoroscopy

by

Jeffrey Thomas Bingham

Submitted to the Department of Mechanical Engineering
in partial fulfillment of the requirements for the degree of

Master of Science in Mechanical Engineering

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

June 2006

Author ....
Department of Mechanical Engineering
May 12, 2006

Certified by ....
Guoan Li
Associate Professor of Orthopaedic Surgery
Harvard Medical School
Thesis Supervisor

Certified by ....
Derek Rowell
Professor of Mechanical Engineering
Thesis Supervisor

Accepted by ....
Lallit Anand
Professor of Mechanical Engineering
Chairman, Committee on Graduate Students

An Automated Matching Algorithm for Dual Orthogonal Fluoroscopy

by

Jeffrey Thomas Bingham

Submitted to the Department of Mechanical Engineering
on May 12, 2006, in partial fulfillment of the
requirements for the degree of
Master of Science in Mechanical Engineering

Abstract

The current method of studying in vivo kinematics of human joints is a tedious and time consuming task. Current techniques require the segmentation of hundreds of raster images, often by hand, and registration of three-dimensional models with two-dimensional contours. Automation of these processes would greatly accelerate research in this field. This research presents an automated method for recovering the pose of a three dimensional model from planar contours. Validation of the algorithm and discussion of its performance and applicability are detailed herein.

Thesis Supervisor: Guoan Li
Title: Associate Professor of Orthopaedic Surgery
Harvard Medical School

Thesis Supervisor: Derek Rowell
Title: Professor of Mechanical Engineering

Acknowledgments

With the opportunity to point fingers at people who made me the way I am, I jump at the chance to accuse my benefactors. First and foremost I thank my mother and father. My successes have been achieved through the love, inspiration, knowledge, and guidance they have given me. My family, especially my little sister, has made life worth living.

I must also acknowledge the many teachers and mentors of my life, who have shown me the secrets of engineering. Mr. Walker, who taught me what an engineer should be; Prof. Schoen, who introduced me to research; and Dr. Li, who enabled this research.

Finally, I thank my friends in Idaho and MIT. Remembering all of the fun times, helped muscle through the rough spots. Besides, I like to think that together we conquer the world.

In short, I am very thankful for the opportunity to write this thesis. My life has been truly blessed because of the people in my life, without them I am nothing.

“. . . because I am involved in mankind, and therefore never send to know for whom the bell tolls; it tolls for thee.” - John Donne

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Fluoroscopic imaging techniques have been used extensively to measure in-vivo kinematics of total knee arthroplasty (TKA) because of the relatively low radiation dosage and the accessibility of the equipment [2, 9, 25]. Previous studies employed a single fluoroscope to take sagittal plane images of the knee at multiple flexion angles [2, 10, 29, 6, 32]. Using the geometry of the fluoroscope, three dimensional (3D) computer models of the tibial and femoral components were matched to the two-dimensional (2D) features of the acquired fluoroscopic images. When the features were considered to be matched, the relative poses of the 3D component models represented the in-vivo knee kinematics, where pose is defined as the position and orientation specifying the six degree-of-freedom (6DOF) location of an object. Using this technique numerous data have been reported on knee motion within the image plane of the fluoroscope. However, determining knee motion in the direction perpendicular to the fluoroscopic image plane has been questioned in recent studies [31, 15, 19].

Pursuing higher accuracies with fluoroscopy, Li et al. [19] completed a 3D study on quasi-dynamic in-vivo kinematics of the knee and more recently, Hanson et al. [9] and Suggs et al.[26] applied two fluoroscopes to formulate a dual-orthogonal fluoroscopic system to investigate in-vivo TKA kinematics. These studies utilized a computer aided drafting (CAD) program to simulate the fluoroscopic environment and manually manipulate 3D models in space so that their projected silhouettes matched outlines of the components on both fluoroscopic images. This methodology has been proven to accurately recreate the 6DOF motion of the knee at multiple flexion angles [9]. However, the manual matching procedure is laborious, impairing its application to study continuous dynamic motion. Automating the matching procedure would reduce the time required to match and improve the repeatability of the dual fluoroscope methodology in determining in-vivo kinematics.

An optimization algorithm is presented for automating the process of matching projections of 3D models of the tibia and femur to two orthogonal planar images of the patient acquired with a dual-orthogonal fluoroscopic imaging system. Accuracy and repeatability of the algorithm's ability to determine 6DOF location are discussed. Results from a complete validation demonstrate the repeatability of the algorithm for determining in-vivo TKA poses.

1.1 Previous Methods

Current research on recreating the pose of 3D objects in the orthopaedic field is driven mostly by the desire to determine joint kinematics. Pose reconstruction from radiographic data offers a non-invasive method for examining in vivo motion of joints. The method itself is conceptually simple. A computer generated three-dimensional model of the joint is obtained through MRI or CT imaging, and then radiographs are taken with the joint in motion. The next step is to match a pose of the three-dimensional computer model to the radiographs of the joint motion. This effectively gives one a three-dimensional dynamic model of the joint in question.

The three main methods requiring pose reconstruction include stereophotogrammetry, single plane fluoroscopy and biplanar fluoroscopy. Each method uses either a variation of the hypothesis and test algorithm or a template matching scheme to determine the true pose of the model. Variations of the hypothesis and test algorithm include iterative closest point, non-overlapping area, iterative inverse point matching and image matching of digitally reconstructed radiographs. These variations provide a method of formulating an objective function which is then optimized using a wide range of available minimization routines.

1.1.1 Imaging Methodology

The first classification of these pose recreation methods lies in the imaging techniques. Three dimensional imaging is used to generate very accurate anatomical models of patient geometry. Two dimensional imaging allows for acquisition of high speed motion of a patient performing functional activities, which provides a record of the in-vivo kinematics. The combination of the 2D and 3D data is then used to generate a database of highly accurate six degree-of-freedom kinematics. Several different methods of acquiring this data has been presented in the literature.

Three Dimensional Imaging

The creation of the three dimensional model is done with a variety of techniques depending on the application. For in-vivo models the majority are created by segmenting images from MRI and creating solid mesh models. A small number of researchers have also used CT scan data to create these models. Combined CT and MRI data have also been used to create voxel models as well. For patients who have had arthroplasty, models from manufacturer CAD and laser scanning of the components has been employed.

Two Dimensional Imaging

The creation of radiographs for pose recreation has been accomplished with x-ray and fluoroscopy. Fluoroscopy is distinguished from x-ray by having a lower radiation dose, which is accomplished by employing a fluorescent detector. Image quality is generally better with high dosage x-ray, but the hazard to the patient makes fluoroscopy a more desirable method. However, the greatest effect on pose recreation is the number and geometry of simultaneous radiographic images taken. Radiographic configurations currently in practice include roentgen stereophotogrammetry (RSA), single plane fluoroscopy and biplanar fluoroscopy. RSA is the oldest method and was originally accomplished by implanting radio-opaque markers into the patient, but has recently employed markerless techniques[7, 27, 28]. RSA takes two simultaneous images from separate focal points, which are projected onto a single plane. Fluoroscopy, on the other-hand, has two variants. The majority of the past research has used a single fluoroscopic image plane[2, 6, 10, 11, 15, 20, 21, 29, 30, 32]. Recently researchers have employed two fluoroscopic images, some in orthographic configurations, and others in more conventional stereo modes[1, 4, 9, 16, 19, 18, 26, 31].

1.1.2 Algorithm Type

Pose recreation can be further classified based on the algorithms used for determining pose. Most of these algorithms have their origins in machine vision algorithms and computer graphics. In general they have been modified to meet the particular requirements of the pose recreation problem. In general there are only two algorithms, template matching and hypothesis and test.

Template Matching

Template matching for pose recreation in biomechanics was pioneered by Banks and Hoff[2, 10]. In template matching the 3D model is first segmented into a library of many different projections. Each library record contains a silhouette of the projected model and the corresponding pose variables. Then by implementing shape matching techniques, outlines in the library are matched to contours segmented from fluoroscopic contours. Libraries of outlines have been created using silhouettes normalized for rotation, translation and scale. This can be done directly or by parameterizing the contours with fourier coefficients. Similarity values are computed based on similarity of normalized parameters or total amount of overlapping area.

Hypothesis and Test

Hypothesis and test is a very generic structure of pose estimation. The algorithm consists simply of two parts, determining a "hypothesis" for the position of the ideal pose and applying a "test" to determine the validity of the hypothesis[14]. In practice the variations of the algorithm are best characterized by their methods of optimization and their formulation of objective functions. Optimization being the method of determining a successively better hypothesis and the objective function a method of testing validity and guiding the next hypothesis of pose.

1.1.3 Optimization

Due to the highly non-linear nature of any feasible objective function employed to determine pose, the optimization methods used to date are either nonlinear global algorithms or heuristic direct search methods. Classical gradient methods that have been employed include: Levenberg-Marquardt Least Squares, and Feasible Sequential Quadratic Programming. Heuristic direct search methods include: Simulated Annealing, Sequential Parameter Search and Powell's Method. Brief description of these methods and several other suggested minimization routines are given below.

Levenberg-Marquardt Nonlinear Least Squares

This method was first used by Lavallee and later Zuffi and Yamazaki[16, 32, 30]. The method utilizes a trust region modification of the Gauss Newton algorithm, this optimization method switches between gradient descent and Gauss-Newton for best trade-off of speed and convergence.

Feasible Sequential Quadratic Programming

This is a further advancement of the Gauss-Newton scheme, and assumes the problem is in quadratic form and all constraints are linearized. Feasible refers to only allowing values of the dependent variables within a desired or computable range. Kanisawa, Kaptein and Valstar have used this method of optimization[11, 13, 28].

Nelder-Mead Downhill Simplex Method

Yamazaki also introduced this method for finding the optimal pose[30]. A simplex is the term given to a "figure" which has n+1 vertices in the n-dimensional search space. The simplex is used for determining search directions and is continually expanded, contracted or reflected based on minimization of the vertices. Random steps are interjected to help remove difficulties with local minima.

Powell's Method

Powell's method presents an optimization algorithm that does not require finding the partial derivatives. It increments along a set of conjugate base vectors in order to find a minimum. Simply put the algorithm starts with a set of initial vectors, computes a function step and based on the results of the step re-computes the solution vectors. This is repeated until a minimum is found. Tomazevic used Powell's Method for minimization[27].

Simulated Annealing

A popular heurestic method for solving global optimization problems and used by several authors [20, 31]. A search space is defined, a random generator modifies the search variables based on a "temperature" and at each iteration the temperature is "cooled" lowering the change in variables. The cooling schedule is predefined with rules for when to change the step sizes.

Sequential Parameter Search

A very straightforward method utilized by Hoff[10]. Each variable in the search space is sequentially minimized. Variations of this also introduce curve fitting techniques in order to determine minimum basis directions that do not directly follow a single parameter.

Swarming

A more recent heurestic search method of swarming was investigated by Schutte[23]. Random points in the search space are computed, the random cloud of search points then moves towards the minimum point with the greatest "velocity". Based on the total velocity of the cloud, these points move through the search space, until a global minimum is reached.

1.1.4 Objective Functions

The bulk of any variation of the hypothesis and test algorithm lies in defining a procedure that will test the validity of a given pose and help to determine the next best guess. This procedure is termed an objective function, cost function and sometimes performance index in literature. In this paper it is referred to as an objective function. Its purpose is to define a function which returns a scalar quantity that when minimized results in an ideal match of the model pose from the given silhouette data.

Iterative Closest Point

The Iterative Closest Point (ICP) algorithm minimizes the distance between points on the projected contour and the given contour[6, 12]. The distance compared is the minimum distance for each point to the given contour, hence the name. When this distance is minimized the projected contour should be exactly the given contour and the pose found.

Non-overlapping Area

Non-overlapping Area (NOA) is also well described by its name[28]. After projecting a silhouette from the model, any area that is not simultaneously covered by both the projected and given silhouettes is summed. The NOA is then the union minus the intersection of the projected and given silhouettes. When this area is minimized the projected silhouette should perfectly overlay the given silhouette and the ideal pose is recovered.

Iterative Inverse Point

In the Iterative Inverse Point (IIP) matching algorithm, rays connecting the focal point to points on the given contour are constructed. Then, distance between the model surface and the projected rays is minimized. When the rays form bi-tangents on the model's surface the ideal pose is recovered. Pre-computed maps describing the distance from the surface of the model have been used to reduce computational burden. These three dimensional distance maps are constructed using a specialized quad-tree decomposition first implemented by Lavallee and later adopted by others[13, 16, 30, 32].

Digitally Reconstructed Radiographs

The algorithm for Digitally Reconstructed Radiograph (DRR) matching utilizes rendering techniques in order to produce a projected raster image that simulates an x-ray[20, 27, 30]. The simulated raster image is then compared to the given x-ray image. When similarity between the two raster images is maximized the ideal pose is recovered. Image similarity measures for the raster images are accomplished using a variety of techniques used in image matching. Intensity values and edge overlap are the most widely used metrics. However, quad-tree decomposition and cross correlation have also been used successfully as measures of image similarity.

1.2 Motivation

Many factors affect the ability to recover the correct pose of a particular scene; however, the number of simultaneous radiographic views has the greatest impact. This is because sensitivity of matching out-of-plane translation is far worse than translation within the plane. In order to minimize error a procedure using two orthogonal radiographs has been developed that combines the best attributes of magnetic resonance imaging (MRI) and fluoroscopy[4, 9, 19].

This imaging procedure can be applied to most of the articulating joints in the human body. This allows for highly accurate in-vivo study of joint kinematics and dynamics, cartilage contact and ligament interaction. The technique is especially attractive for accurate measurement of in vivo kinematics of patients with orthopaedic implants because the 3D wire-frame model of the orthoses can be obtained directly from the CAD model used for manufacturing. Information gained from this imaging technique can be used for designing new medical devices, for pathological diagnosis and for planning surgical procedures as well.

Currently this procedure is accomplished manually, which is time consuming. Also, because of the vagaries of human precision, overall consistency is not well defined. Finally, the procedure necessitates the mastery of new knowledge and skills to achieve accurate results. A fully automated procedure will accelerate the matching process, stabilize repeatability and reduce the learning curve for the procedure. Research of algorithm development, algorithm implementation and validation of the procedure will ultimately realize a fully automated pose reconstruction. The purpose of this research is to present a method for automating this procedure and providing a thorough validation.

Chapter 2

Pose Reconstruction Procedure

The method of pose reconstruction is conceptually simple. A computer generated 3D model is obtained and radiographs are taken of the subject in motion. Based on the geometry of the radiographic setup it is possible to match 2D features in the radiograph to points on the 3D model. When the features are matched the pose of the 3D model is recreated, which effectively gives one a 3D dynamic model. The process of acquiring images with the dual-orthogonal fluoroscopic image system and recreating in-vivo kinematics can be separated into three stages: imaging, image analysis and matching. A flow chart of the manual process is shown in Fig. 2-1. Automating each of these tasks increases the speed of the system and improves the repeatability; however, this comes at a cost of lost sensitivity. When a balance between automation and manual interaction is reached the system is efficient, robust and accurate. To emphasize this balance, a discussion of the amount of human intervention accompanies the description of each stage of the process. These algorithms are implemented in Matlab software and listings can be found in Appendix A.

2.1 Imaging

Acquisition of the images is where the actual kinematic data is recorded. With the technological advancements in magnetic resonance imaging (MRI) and x-ray machines it is possible to recreate fully 3D anatomical models and also record human activ-

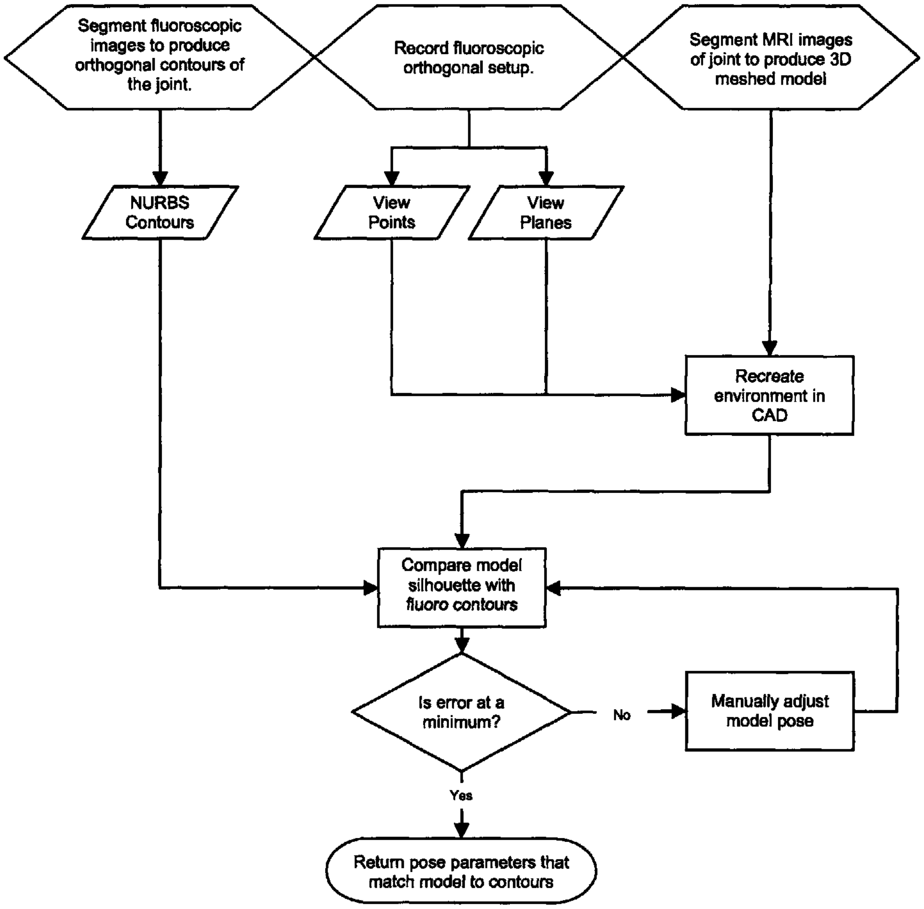

Segment fluoroscopic images to produce orthogonal contours of the joint.
Record fluoroscopic orthogonal setup.
Segment MRI images of joint to produce 3D meshed model
NURBS Contours
View Points
View Planes
Recreate environment in CAD
Compare model silhouette with fluoro contours
Is error at a minimum?
No
Manually adjust model pose
Yes
Return pose parameters that match model to contours

Figure 2-1: Flow of the manual matching process.

ity in real-time. MRI provides a tool to recreate the anatomy with sub-millimeter precision. MRI can easily be replaced with computed tomography (CT) scans, or other 3D reconstruction techniques, including computer aided design (CAD) models of metal components; however, MRI is most often the least invasive method available. Similarly, any x-ray method can be used, but pulsed fluoroscopy exposes the patient to the least amount of radiation. The general imaging requirements of the process are therefore a method of recreating the 3D geometry and the ability to capture two orthogonal images of the 3D geometry's motion. This research explored MRI and CAD for model generation and pulsed fluoroscopy for acquiring images of patient motion.

2.1.1 MRI

In order to create 3D models of living anatomy MRI has been utilized to capture the structure. For the knee joint, patients are asked to lie supine in an MRI having a 3 Tesla magnet. Using a knee coil, approximately 120 sagittal images are acquired with a section thickness of 1 mm, a field of view of 160 × 160 mm and a resolution of 512 × 512 pixels. Acquisition was accomplished with a flip angle of 25°, imaging frequency of 123.3 kHz, an echo time of 6.5 ms, and a 24 ms repetition time (Fig. 2-2).

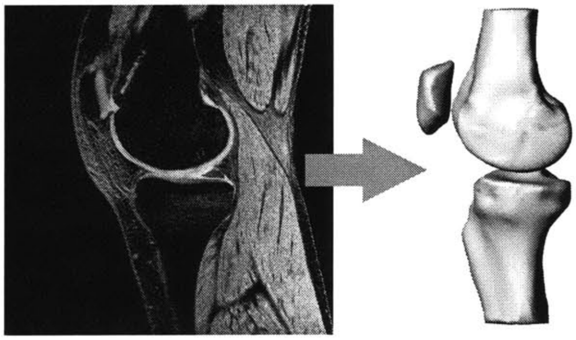


Figure 2-2: An example MRI slice on the left used to construct the 3D surface model on the right.

2.1.2 Dual-Orthogonal Fluoroscopy

The dual-orthogonal fluoroscopic image system consists of two fluoroscopes (OEC 9800 ESP, GE, Salt Lake City) positioned with the two image intensifiers perpendicular to each other (Fig. 2-3) [9]. A subject is free to move within the common imaging zone of the two fluoroscopes. The subject is then asked to move through a series of flexion angles which are imaged simultaneously by the fluoroscopes to acquire images of the knee from two perpendicular directions. During this procedure, the average subject receives 106 mrem of radiation for 20 seconds of pulsed fluoroscopy at 65 kVp and 0.80 mA. In addition to the subject images, a set of calibration images are acquired. Calibration images are taken of a perforated plate for distortion correction and a set of beads in a known configuration for the recreation of the fluoroscopic geometry. The images are stored electronically with an 8-bit gray-scale and a resolution of 1024 x 1024 pixels, corresponding to a 315 x 315 mm field-of-view. This procedure records the in-vivo poses of the knee as a series of 2D paired orthogonal fluoroscopic images.

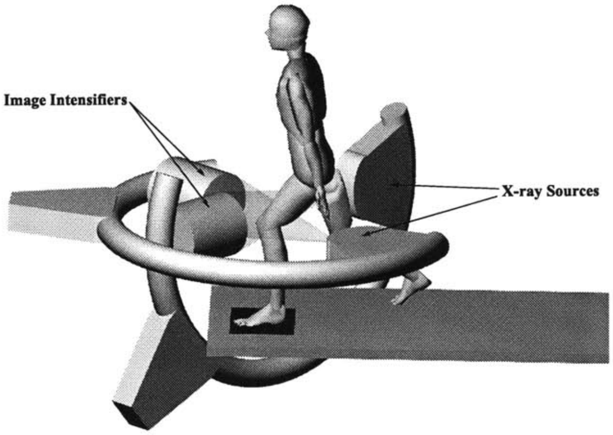


Figure 2-3: A dual-orthogonal fluoroscopic system for capturing in-vivo knee joint kinematics.

2.2 Image Analysis

In order to extract kinematic data from the acquired images the data must first be corrected for distortion, structural features then extracted and finally the fluoroscopic imaging system must be recreated in simulation. While each step can be automated, the entire process still requires a high level of human interaction. This is particularly true for segmentation, and without development of powerful machine learning algorithms, human operation will continue to provide the most efficient and reliable results.

2.2.1 Distortion Correction

The fluoroscopic images suffer from small amounts of distortion caused by the slightly curved surface of the image intensifier and environmental perturbations of the x-ray. In order to remove "swirl" caused by electro-magnetic disturbance and "fish-eye" from the curved image surface a known grid is imaged and the subsequent image is mapped to the known geometry. A global surface mapping using a polynomial fitting technique adapted from Gronenschild is used to accomplish this [8]. Linear interpolation and local distortion correction were compared, but it was found that global correction provided the most consistent results.

A plexi-glass plate with a pattern of holes in concentric circles is used for the reference geometry. This radial geometry was found to work best, because it conforms well with the circular intensifiers and allows for a high density grid that is conducive to solving the mapping problem. Distortion correction is accomplished by mapping the distorted grid to the known grid using a set of two-dimensional polynomials. The images themselves are corrected by using the spatial mapping to "move" pixels with linear interpolation of the intensity values. Matlab code used for correcting distorted images can be found in Appendix A.1.1.

2.2.2 Segmentation

Thresholding, region growing and edge detection have been explored as possible methods of extracting structural information from raster images. In order to generate contours, edge detection is currently the most useful method. Most edge detection methods are variants of a basic premise of determining the maximum gradient of image intensity. Sobel, Prewitt, Roberts and Laplacian of Gaussian algorithms return similar results for fluoroscopic and MRI images. The most favorable algorithm is the one presented by Canny [3]. This algorithm improves on previous methods by adding rules for discarding erroneous edges. The basic algorithm first smooths the image using a Gaussian filter, then it computes the gradients from a Laplacian filter. Next, the gradients are reduced by removing non-maximal values. The edges created by the maximal values are further reduced by applying a threshold and examining connectivity. Non-maximal edges connected to maximal edges are kept, while isolated non-maximal edges are removed. Canny edge detection is implemented in Matlab and used as a first pass for extracting structural information from MRI and fluoroscopic images. An example of automatic segmentation is given in Fig. 2-4.

Due to the complex geometry of most anatomical structures and the inherent lack of an edge in biological images, the outlines from the edge detection are manually reviewed. This is done by saving the segmented outlines as a list of 2D spatial points, which in turn are used to create spline curves using a periodic spline algorithm [17]. These outlines can then be edited in CAD software. Manual editing is most often required where soft tissue attaches to bone as there is a decrease in intensity gradient. These edited curves are then used to create 3D models and define matching geometry. Often it is more effecient to manually segment the images, as the results are often more accurate albeit slightly less repeatable. A recent study by Mahfouz, et al. has discussed these results[21]. The results of the segmentation are used to generate 3D models and 2D contours. To generate the 3D models, the contours are placed spatially based on the MRI information and a mesh is created using a point lofting method, a listing of which can be found in Appendix A.1.2. Splines generated from the fluoroscopic images are saved along with the starting and ending values in order to determine the valid limits of the splines. These models and contours are used later in the process for recreating the pose of the patient.

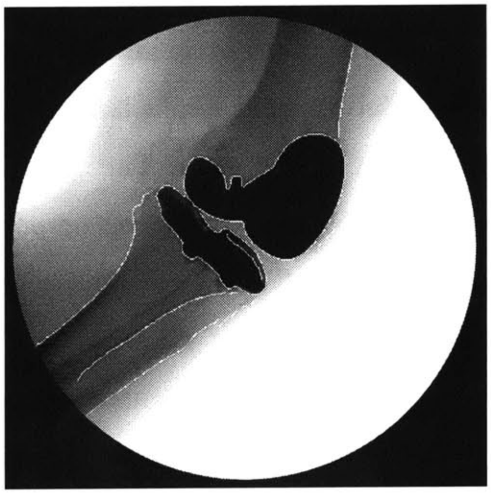


Figure 2-4: Canny automatic segmentation of a fluoroscopic image.

2.2.3 Environment Recreation

Before matching can occur a virtual replica of the dual-orthogonal fluoroscopic system is constructed. This is accomplished from calibration data, which locates the intensifier centers and the relative orientation of the fluoroscopes. By aligning the solutions of each fluoroscope the relative alignment of the fluoroscopes is determined. With this calibration data, two virtual source-intensifier pairs are created in a solid modeling program (Rhinoceros, Robert McNeel & Associates, Seattle, WA) and the virtual intensifiers are oriented so that their relative locations replicate the geometry of the real fluoroscopic system (Fig. 2-5).

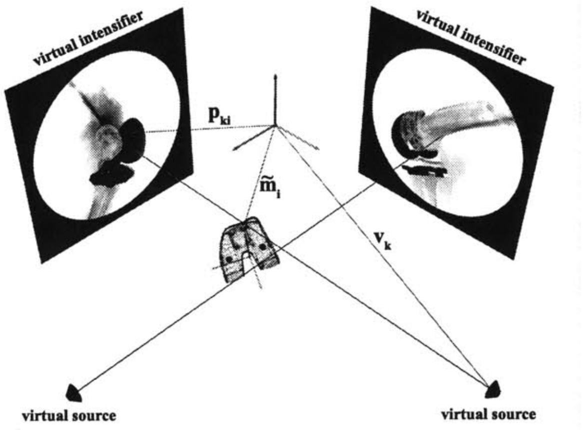


Figure 2-5: A virtual dual-orthogonal fluoroscopic system constructed to reproduce the in-vivo knee joint kinematics.

The splines of the TKA components obtained from the dual fluoroscopic images are placed on their respective virtual intensifiers. Next, 3D models of the tibial and femoral components are introduced into the virtual system. For TKA the models are obtained from the manufacturer as non-uniform rational b-splines (NURBS) surfaces, otherwise the models are obtained from MRI. Using the 3D modeling program a mesh size is selected. The surfaces are tessellated, and the vertices of the mesh are used to create a 3D point cloud of the model. Then, a local coordinate system is created for each point model. The local coordinate system is related to the global coordinates of the virtual fluoroscopic environment using a position vector and rotation matrix (Fig. 2-6).

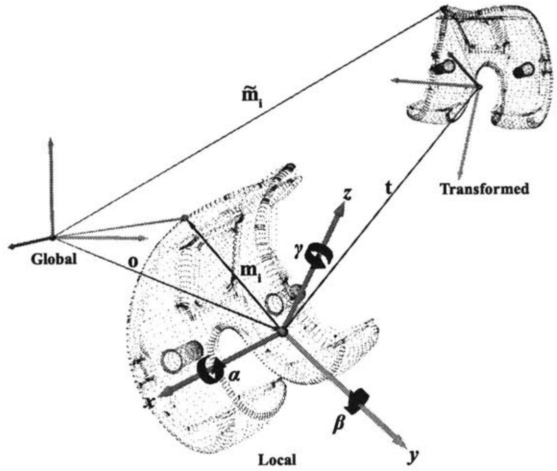


Figure 2-6: Definition of local and global coordinate systems and the transformation of model points.

Using the 3D modeling program the point model can be manipulated in the virtual environment to create an initial guess of the pose. With this CAD replica, a mathematical model of the dual-orthogonal fluoroscopic system is constructed and matching can commence.

2.3 Matching

The matching stage recreates the subject's position and orientation that was captured by the fluoroscopes. This is accomplished by registering the projected silhouettes of the 3D model with the 2D contours of the fluoroscopic images. Registration is done by translating and rotating a virtual 3D model of the subject in a simulated environment containing the fluoroscopic images. When the 3D model matches the 2D images the virtual pose should match the pose of the subject during the fluoroscopic imaging. This has been accomplished manually by employing solid modeling software [9, 19]. However, the manual process is tedious, time consuming and lacks a quantitative measure for repeatability. The qualitative aspect of manual matching requires extensive experience to obtain the accuracy described in the previous publications [9, 19]. In addition, the prospect of dynamic motion the number of images to match greatly increases, thus increasing processing time. It therefore becomes desirable, perhaps even a necessity, to automate the matching process. Automatic matching offers decreased matching time, a reduced learning curve and most importantly a quantitative matching criterion that is independent of the operator.

Chapter 3

Automated Matching

In the previous chapter the method for pose reconstruction was illustrated. The basic processes of imaging, image analysis and matching, however, can be greatly accelerated through automation. This chapter improves on the previous procedure by presenting an automated algorithm for matching. A flow chart of the enhanced process is shown in Fig. 3-1.

The automatic matching algorithm is formulated as an optimization procedure that minimizes the error between projected model silhouettes and actual fluoroscopic image outlines in order to determine the model pose. The model pose is defined by the 6DOF position and orientation of each models local coordinate system relative the global coordinate system. The objective function is expressed as a scalar function with six independent variables. The independent variables are the three components of the position vector locating the origin of the local coordinate system and the three Euler angles of the local system in the global system (Fig. 2-6). The scalar function value is the average distance between the 3D projected model silhouettes and the segmented fluoroscopic outlines. A listing of Matlab implementations can be found in Appendix A.2.

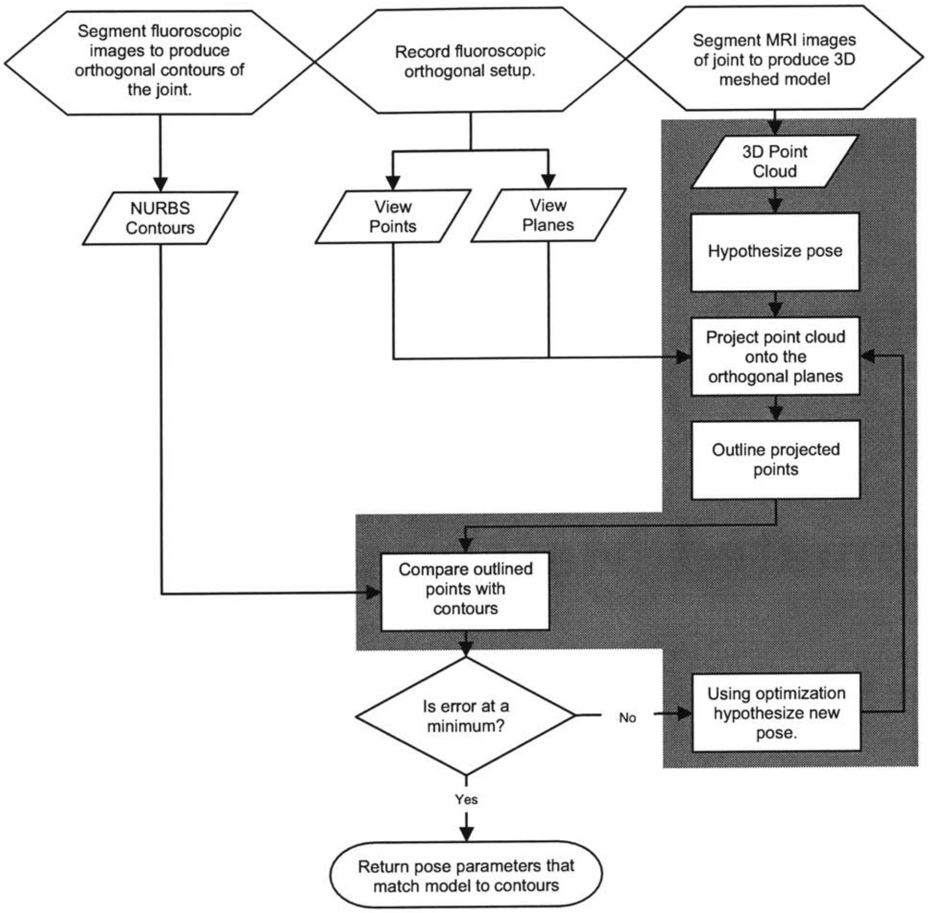

Segment fluoroscopic images to produce orthogonal contours of the joint.
Record fluoroscopic orthogonal setup.
Segment MRI images of joint to produce 3D meshed model
NURBS Contours
View Points
View Planes
3D Point Cloud
Hypothesize pose
Project point cloud onto the orthogonal planes
Outline projected points
Compare outlined points with contours
Is error at a minimum?
No
Using optimization hypothesize new pose.
Yes
Return pose parameters that match model to contours

Figure 3-1: Flowchart of the matching process, automation shaded in grey.

3.1 Objective Function

3.1.1 Transformation

The six independent variables are used to transform the points of the 3D model from an initial pose to a new position and orientation which is illustrated in (Fig. 2-6). Each point on the model, noted as $\mathbf{m_i}$, is transformed with the local coordinate system to a new location and orientation in the global coordinate system, noted as $\tilde{\mathbf{m}}_\mathbf{i}$ in Eq. 3.1.

$$\tilde{\mathbf{m}}_\mathbf{i} = \mathbf{R}(\mathbf{m_i}) + (\mathbf{t} + \mathbf{o}) \tag{3.1}$$

The vector, $\mathbf{o} = [x_o, y_o, z_o]$, locates the origin of the local system in the global coordinates. The translation vector, $\mathbf{t} = [x_t, y_t, z_t]$, is defined in the global coordinate system. The rotation matrix, $\mathbf{R}$, is defined as a Y-Z-X Euler sequence using the angles α, β, and γ shown in Eq. 3.2.

$$\begin{bmatrix} cos(\beta)cos(\gamma) & sin(\gamma) & -sin(\beta)cos(\gamma) \\ -cos(\alpha)cos(\beta)sin(\gamma)+sin(\alpha)sin(\beta) & cos(\alpha)cos(\gamma) & cos(\alpha)sin(\beta)sin(\gamma)+sin(\alpha)cos(\beta) \\ sin(\alpha)cos(\beta)sin(\gamma)+cos(\alpha)cos(\beta) & -sin(\alpha)cos(\gamma) & -sin(\alpha)sin(\beta)sin(\gamma)+cos(\alpha)cos(\beta) \end{bmatrix} \tag{3.2}$$

3.1.2 Projection

Once the 3D model is transformed to a new pose, the locations of the virtual sources are used to project the points onto the intensifiers of the virtual fluoroscopic system (Fig. 2-5). The vector equation used to project the transformed 3D model points onto a virtual intensifier is shown below in Eq. 3.3.

$$\mathbf{p}_{ki} = \mathbf{v}_k + \frac{l_k}{(\tilde{\mathbf{m}}_\mathrm{i} - \mathbf{v}_k) \cdot \mathbf{n}_k}(\tilde{\mathbf{m}}_\mathrm{i} - \mathbf{v}_k) \tag{3.3}$$

The i^{th} projected model point for the k^{th} intensifier is defined as $\mathbf{p}_{ki}$ and $\tilde{\mathbf{m}}_\mathrm{i}$ the i^{th} transformed model point. The vector $\mathbf{v}_k$ locates the k^{th} source and $\mathbf{n}_k$ the unit vector normal to the k^{th} intensifier plane. The scalar l_k is the distance between the k^{th} source and intensifier.

3.1.3 Point Selection

To decrease computation time and improve the robustness of the algorithm, only the outline points of the projected 3D model points are compared to the outlines of the TKA components. An outlining set of the projected points is determined by establishing point connectivity and following an outer contour defined by the connectivity. Connectivity is determined by automatically compartmentalizing the model points so that a connected grid is produced (Fig. 3-2A). Using a left-looking, contour-following algorithm, the grids that outline the projected points are determined (Figs. 3-2B-C). For each contour grid, the point closest to the outside of the contour is selected (Figs.

3-2D-E). This automatic procedure results in a set of points that form an outline of the projected 3D model points (Fig. 3-2F).

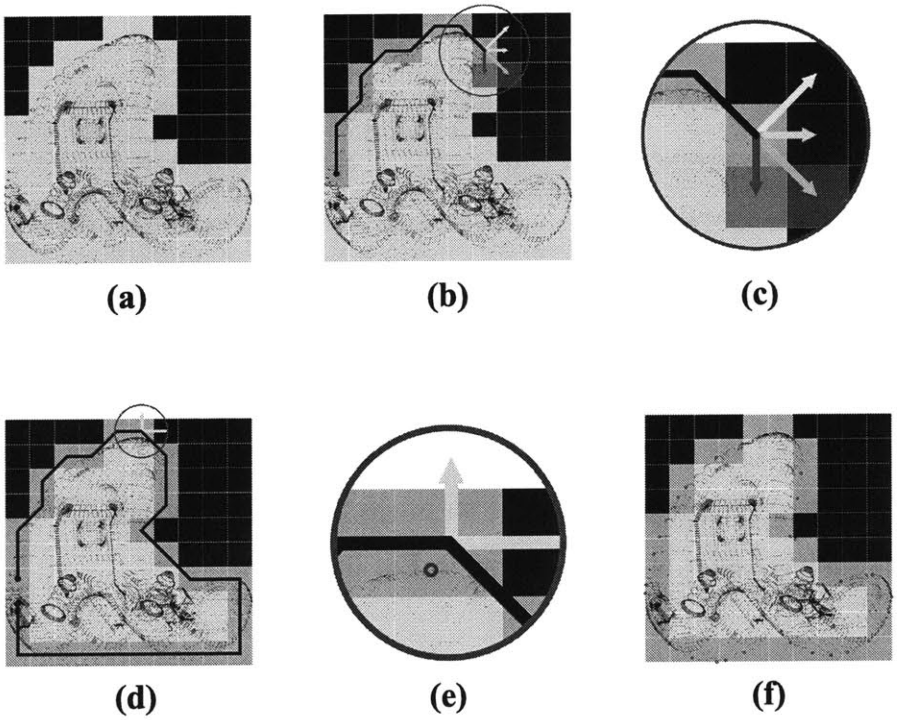


Figure 3-2: Outlining procedure. (A) Compartmentalize projected points (B-C) Determine boundary grids with left-looking outlining technique (D-E) Select point in each grid that is closest to the outer edge. (F) Completion of algorithm with selected outline points.

3.1.4 Error Computation

The minimum distance between each outlined, projected 3D model point and the fluoroscopic spline is determined (Fig. 3-3). Since the spline is represented as a parametric curve, the secant method is used to determine the minimum distance between a point and the spline (Eq. 3.4).

$$d_{ki} = \min_{t} |\mathbf{g}_k(t) - \mathbf{p}_{ki}| \tag{3.4}$$

For the k^{th} intensifier, d_{ki} is the i^{th} minimum distance, $\mathbf{p}_{ki}$ is the i^{th} projected

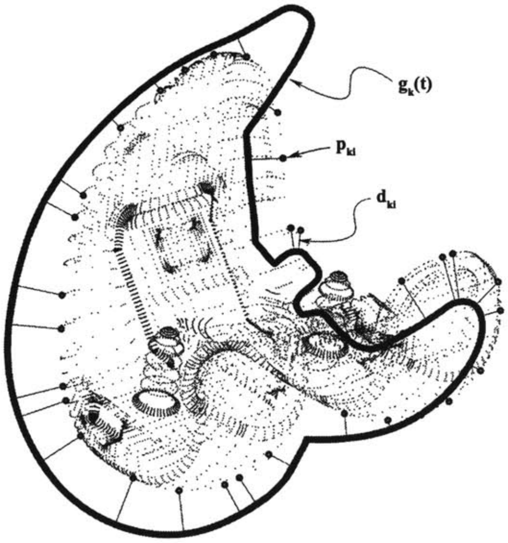


Figure 3-3: Representation of calculating the minimum distance between projected points and a fluoroscopic outline.

model point, $\mathbf{g}_k(t)$ is the parametric vector function of the fluoroscopic image spline, and t is the parametric variable, whose range corresponds to the recorded endpoints of the spline. For each intensifier, these distances are summed and divided by the total number of points, q_k, resulting in a normalized distance. The normalized distances for each intensifier are then summed and returned as the value of the objective function, I (Eq. 3.5).

$$I = \sum_{k=1}^{2} \frac{1}{q_k} \sum_{i=1}^{q_k} d_{ki} \tag{3.5}$$

3.2 Optimization

As the objective function, I, approaches zero the 3D model closely approaches the actual TKA pose. Therefore, to accurately replicate the actual TKA pose the objective function is minimized according to Eq. 3.6.

$$\min_{\mathbf{r}} I(\mathbf{r}) = \min_{\mathbf{r}} \sum_{k=1}^{2} \frac{1}{q_k} \sum_{i=1}^{q_k} \min_{t} \left| \mathbf{g}_k(t) - \mathbf{v}_k - \frac{l_k}{(\tilde{\mathbf{m}}_\mathbf{i} - \mathbf{v}_k) \cdot \mathbf{n}_k} (\tilde{\mathbf{m}}_\mathbf{i} - \mathbf{v}_k) \right| \tag{3.6}$$

The vector, $\mathbf{r}$, holds the 6 optimization variables, $r = (x, y, z, \alpha, \beta, \gamma)$, that represent the position and orientation of the models local coordinate system with respect to the global coordinate system. Any global optimization routine can be implemented to perform the minimization of the objective function, $I(\mathbf{r})$, Eq. 3.5. Heuristic search methods, such as genetic algorithms and simulated annealing are also possible; however, simple quasi-Newton methods appear to have the fastest convergence with excellent results. Minimization of this function is accomplished with the Broyden, Fletcher, Goldfarb, and Shanno (BFGS) quasi-Newton method and implemented in Matlab software [5]. Utilizing the BFGS algorithm provides constrained line-search minimization of the performance index. The basic BFGS quasi-Newton method is formulated as a minimization of the quadratic in Eq. 3.7.

$$I(\mathbf{r}) = \frac{1}{2}\mathbf{r}^T\mathbf{H}\mathbf{r} + \mathbf{c}^T\mathbf{r} + b \tag{3.7}$$

Where $\mathbf{H}$ is an approximation to the symmetric positive definite Hessian matrix of the objective function, $\mathbf{c}$ is a vector of constants defining constraints, while b is a scalar constant used to adjust for errors in approximating the Hessian. Minimization occurs by solving for the appropriate solution vector that solves Eq. 3.8.

$$\nabla I(\tilde{\mathbf{r}}) = \mathbf{H}\tilde{\mathbf{r}} + \mathbf{c} = 0 \tag{3.8}$$

Solving Eq. 3.8 results in the solution vector $\tilde{\mathbf{r}} = -\mathbf{H}^{-1}\mathbf{c}$. The method of BFGS then presents an iterative solution for the Hessian, $\mathbf{H}$, presented in Eqs. 3.9-3.11.

$$\mathbf{H}_{j+1} = \mathbf{H}_j + \frac{\mathbf{q}_j\mathbf{q}_j^T}{\mathbf{q}_j^T\mathbf{s}_j} - \frac{\mathbf{H}_j^T(\mathbf{s}_j^T\mathbf{s}_j)\mathbf{H}_j}{\mathbf{s}_j^T\mathbf{H}_j\mathbf{s}_j} \tag{3.9}$$

$$\mathbf{s}_j = \mathbf{r}_{j+1} - \mathbf{r}_j \tag{3.10}$$

$$\mathbf{q}_j = \nabla I(\mathbf{r}_{j+1}) - \nabla I(\mathbf{r}_j) \tag{3.11}$$

Upon implementation of the Hessian, the function gradients and search directions can be calculated to determine the optimal solution. Global convergence is improved by randomly perturbing the initial guess and submitting these additional guesses to the optimization routine. Convergence for each run is controlled by terminating the minimization routine when the differential change in variables meets the required tolerance or the quantity of objective function calls exceeds a specified number. Globally, runs that exceed the number of function calls and do not meet the required tolerance are deemed non-convergent. The remaining runs are used to determine a confidence interval and the match having the smallest performance index value is considered optimal.

3.3 Summary

Dual-orthogonal imaging can be applied to most of the articulating joints in the human body. This allows for highly accurate in-vivo study of joint kinematics and dynamics, cartilage contact and ligament interaction. The technique is especially attractive for in-vivo kinematics of patients with orthopaedic implants because 3D wire-frame models of the orthoses can be obtained directly from the CAD models used for manufacturing. Information gained from this imaging technique can be used for designing medical devices, clinical diagnosis and planning surgical procedures.

Automation of the procedure accelerates the matching process, stabilizes repeatability and reduces the learning curve required to produces accurate results. Automation is accomplished by applying BFGS optimization to an iterative closest point objective function. Time savings from the algorithm allows for a large number of poses to be matched, making analysis of dynamic motion feasible. By formulating the matching problem as a mathematical process that can be computed, repeatability is gained and user bias is reduced. Automation also reduces the learning curve required for determining an object's pose. Furthermore, statistical bounds of matching error are readily determined for the process. In short, automation of the matching process improves dual-orthogonal imaging by making it an easier and more robust tool to use, facilitating highly accurate biomechanics research.

Chapter 4

Validation and Application

Validation of the optimized matching algorithm was performed to demonstrate the accuracy and repeatability of recreating pose. The algorithms implemented in Matlab software were tested on a personal computer with a Pentium IV class processor (2.4 GHz, 512 MB RAM) running the Microsoft Windows XP Professional operating system. Validation consisted of running the algorithm with idealized, controlled, and real data using ten randomly generated initial pose estimates for each test. These tests were used to isolate various causes of error; idealized tests to isolate errors with geometry, standardized in-vitro tests to isolate errors caused by segmentation, and in-vivo tests to observe the combination of errors. For all tests the matching algorithm was set to record convergent solutions that did not exceed 800 objective function calls and had a differential tolerance of less than 0.0005 for each variable.

4.1 Idealized Environments

Idealized environments were created in order to determine the automated matching algorithms repeatability, accuracy, sensitivity to model point density and pose orientation, and optimal parameters under controlled conditions. These tests gave a basis for the ultimate potential of the algorithm. The idealized fluoroscopic setup was created by replicating the fluoroscopic environment using the 3D solid modeling software. Three-dimensional models were oriented in the virtual system in poses approximating a deep knee bend. From these poses the models were projected onto the virtual intensifiers, and the projections were used to create pseudo fluoroscopic outlines. Models in this configuration were then considered to be the gold standard. The error of the matched poses was then determined by comparing the matched model poses to the gold standard.

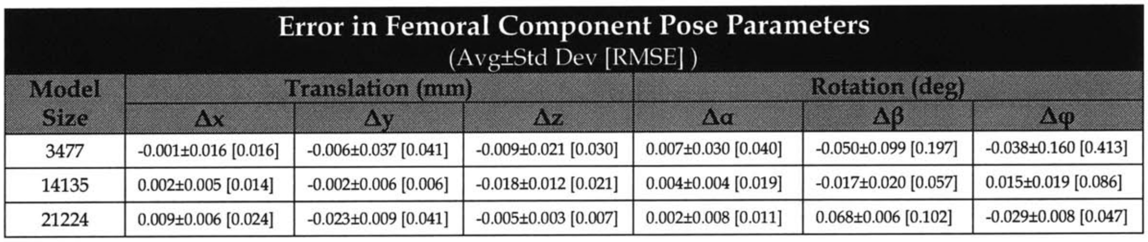
4.1.1 TKA Components

Method

For the TKA components the 3D models of the femoral and tibial components were generated from manufacturer CAD data. With the idealized fluoroscopic setup the effect of model point density was tested for one pose of the tibial and femoral components (45° flexion position) using three different densities. The low point density models used a coarse mesh with approximately 3,500 points. Medium density point models were approximately 15,000 points and high density point models were made with over 20,000 points. Three additional poses (30, 60 and 90° flexion positions) for the medium point density models were matched to test the effect of pose orientation. For each pose ten estimates for the initial guess were created for each component by perturbing the models from the gold standard. The perturbations were created by randomly generating values for the pose variables within the range of ±20 mm and ±20° using a Gaussian distribution. Next, the models were matched using fifty of the projected outline points. The accuracy and repeatability of the optimized matching algorithm in reproducing the femoral and tibial components position and orientation in 6DOF was recorded for each convergent match.

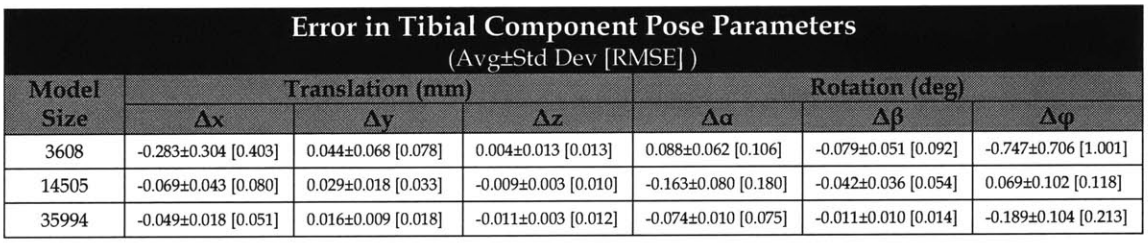
Results

Accuracy for the idealized tests was measured as the error between the body fixed local coordinate systems of the golden standard and the matched models. The sample standard deviation of these errors was selected as the measure of repeatability. Root-mean-square error (RMSE), or population standard deviation, values are also reported for comparison with previous methods. Results for different densities of the femoral component are presented in Table 4.1. The pose of the femoral component was recreated to within 0.01±0.04 mm in translation and0.05±0.16° in rotation for the low point density model, 0.02±0.01 mm and 0.02±0.02° for the medium point density model, and 0.02±0.01 mm and 0.07±0.01° for the high point density model. The average time for matching a single pose was 200 sec, 350 sec, and 510 sec for the low, medium, and high point density models, respectively. The average number of calls to the objective function was 640 for all model sizes. The results for four different flexion angles using the medium point density model are listed in Table 4.2. The average values of the pose variables were found to recreate pose to within 0.02±0.01 mm in translation and 0.02±0.03° in rotation.



Error in Femoral Component Pose Parameters
(Avg±Std Dev [RMSE])

Model	Translation (mm)			Rotation (deg)		
Size	Δx	Δy	Δz	Δα	Δβ	Δφ
3477	-0.001±0.016 [0.016]	-0.006±0.037 [0.041]	-0.009±0.021 [0.030]	0.007±0.030 [0.040]	-0.050±0.099 [0.197]	-0.038±0.160 [0.413]
14135	0.002±0.005 [0.014]	-0.002±0.006 [0.006]	-0.018±0.012 [0.021]	0.004±0.004 [0.019]	-0.017±0.020 [0.057]	0.015±0.019 [0.086]
21224	0.009±0.006 [0.024]	-0.023±0.009 [0.041]	-0.005±0.003 [0.007]	0.002±0.008 [0.011]	0.068±0.006 [0.102]	-0.029±0.008 [0.047]

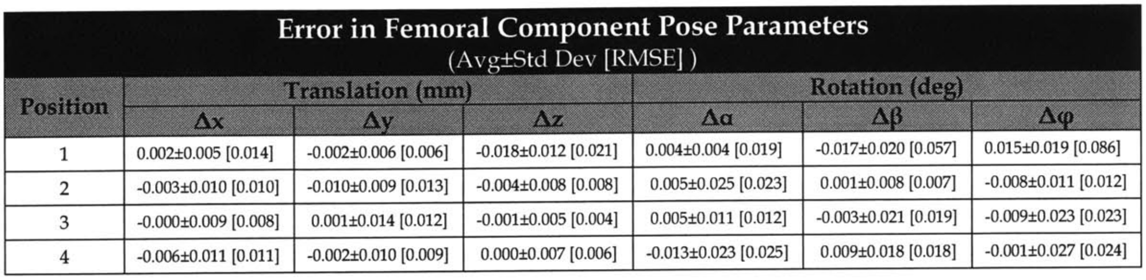


Error in Tibial Component Pose Parameters
(Avg±Std Dev [RMSE])

Model	Translation (mm)			Rotation (deg)		
Size	Δx	Δy	Δz	Δα	Δβ	Δφ
3608	-0.283±0.304 [0.403]	0.044±0.068 [0.078]	0.004±0.013 [0.013]	0.088±0.062 [0.106]	-0.079±0.051 [0.092]	-0.747±0.706 [1.001]
14505	-0.069±0.043 [0.080]	0.029±0.018 [0.033]	-0.009±0.003 [0.010]	-0.163±0.080 [0.180]	-0.042±0.036 [0.054]	0.069±0.102 [0.118]
35994	-0.049±0.018 [0.051]	0.016±0.009 [0.018]	-0.011±0.003 [0.012]	-0.074±0.010 [0.075]	-0.011±0.010 [0.014]	-0.189±0.104 [0.213]

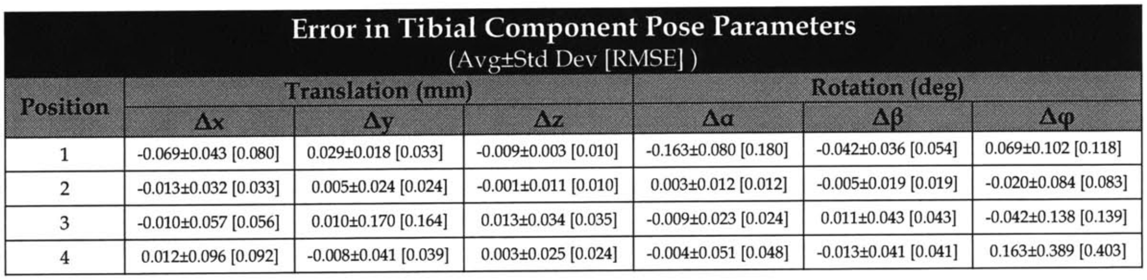
Table 4.1: Accuracy (average error values), repeatability (standard deviations) and root-mean-square errors (RMSE) of the automatic matching procedure in an idealized environment using different model point densities. Accuracy and repeatability were evaluated for ten initial positions.

Results for the tibial component are presented in Table 4.1 for different densities of the model. Pose was recreated to within 0.28±0.30 mm in translation and 0.75±0.71° in rotation for the low density model, 0.07±0.04 mm and 0.16±0.10° for the medium point density model, and 0.05±0.02 mm and 0.19±0.10° for the high point density model. The average time for matching a single pose was 110 sec, 210 sec, and 220 sec for the low, medium, and high point density models, respectively. For each model size



Error in Femoral Component Pose Parameters
(Avg±Std Dev [RMSE])

Position	Translation (mm)			Rotation (deg)		
	Δx	Δy	Δz	Δα	Δβ	Δφ
1	0.002±0.005 [0.014]	-0.002±0.006 [0.006]	-0.018±0.012 [0.021]	0.004±0.004 [0.019]	-0.017±0.020 [0.057]	0.015±0.019 [0.086]
2	-0.003±0.010 [0.010]	-0.010±0.009 [0.013]	-0.004±0.008 [0.008]	0.005±0.025 [0.023]	0.001±0.008 [0.007]	-0.008±0.011 [0.012]
3	-0.000±0.009 [0.008]	0.001±0.014 [0.012]	-0.001±0.005 [0.004]	0.005±0.011 [0.012]	-0.003±0.021 [0.019]	-0.009±0.023 [0.023]
4	-0.006±0.011 [0.011]	-0.002±0.010 [0.009]	0.000±0.007 [0.006]	-0.013±0.023 [0.025]	0.009±0.018 [0.018]	-0.001±0.027 [0.024]



Error in Tibial Component Pose Parameters
(Avg±Std Dev [RMSE])

Position	Translation (mm)			Rotation (deg)		
	Δx	Δy	Δz	Δα	Δβ	Δφ
1	-0.069±0.043 [0.080]	0.029±0.018 [0.033]	-0.009±0.003 [0.010]	-0.163±0.080 [0.180]	-0.042±0.036 [0.054]	0.069±0.102 [0.118]
2	-0.013±0.032 [0.033]	0.005±0.024 [0.024]	-0.001±0.011 [0.010]	0.003±0.012 [0.012]	-0.005±0.019 [0.019]	-0.020±0.084 [0.083]
3	-0.010±0.057 [0.056]	0.010±0.170 [0.164]	0.013±0.034 [0.035]	-0.009±0.023 [0.024]	0.011±0.043 [0.043]	-0.042±0.138 [0.139]
4	0.012±0.096 [0.092]	-0.008±0.041 [0.039]	0.003±0.025 [0.024]	-0.004±0.051 [0.048]	-0.013±0.041 [0.041]	0.163±0.389 [0.403]

Table 4.2: Accuracy (average error values), repeatability (standard deviations) and root-mean-square errors (RMSE) of the automatic matching procedure in an idealized environment using four different pose environments with a model point density of 15,000 points. Each position was evaluated for ten initial positions.

the average number of objective function calls was 650. Results from four different flexion angles for the medium point density model are listed in Table 4.2. The average values of the pose variables were found to recreate pose to within 0.07±0.09 mm in translation and 0.16±0.18° in rotation for the tibial component.

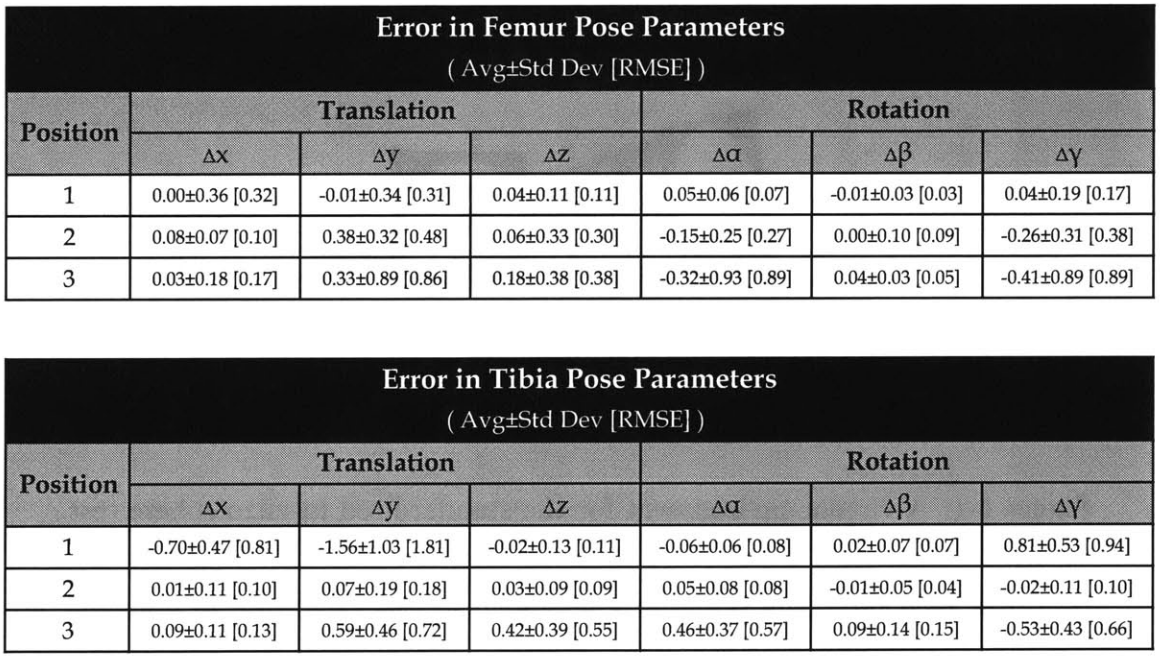
4.1.2 Natural Knee

Method

For the natural knee 3D models of the femur and tibia were generated from an MRI of a cadaver. Three poses (30, 60 and 90° flexion positions) for low point density models (3,500 points) were matched to test the effect of pose orientation. For each pose ten estimates for the initial guess were created for each model by perturbing them from the gold standard. The perturbations were created by randomly generating values for the pose variables within the range of ±20 mm and ±20° using a Gaussian distribution. Next, the models were matched using fifty of the projected outline points. The accuracy and repeatability of the optimized matching algorithm in reproducing the femoral and tibial components position and orientation in 6DOF was recorded for each convergent match.

Results

Accuracy for the idealized natural knee tests was measured as the error between the body fixed local coordinate systems of the golden standard and the matched models. The results for three different flexion angles using the medium point density model are listed in Table 4.3. The average error in recreating pose of the femur was 0.12±0.33 mm in translation and 0.14±0.31° in rotation. Average pose error for the tibia was 0.39±0.33 mm in translation and 0.23±0.20° in rotation. The sample standard deviation of these errors was selected as the measure of repeatability. The average time for matching a single pose was 30 seconds and the average number of calls to the objective function was 160.



Error in Femur Pose Parameters
(Avg±Std Dev [RMSE])

Position	Translation			Rotation		
	Δx	Δy	Δz	Δα	Δβ	Δγ
1	0.00±0.36 [0.32]	-0.01±0.34 [0.31]	0.04±0.11 [0.11]	0.05±0.06 [0.07]	-0.01±0.03 [0.03]	0.04±0.19 [0.17]
2	0.08±0.07 [0.10]	0.38±0.32 [0.48]	0.06±0.33 [0.30]	-0.15±0.25 [0.27]	0.00±0.10 [0.09]	-0.26±0.31 [0.38]
3	0.03±0.18 [0.17]	0.33±0.89 [0.86]	0.18±0.38 [0.38]	-0.32±0.93 [0.89]	0.04±0.03 [0.05]	-0.41±0.89 [0.89]

Error in Tibia Pose Parameters
(Avg±Std Dev [RMSE])

Position	Translation			Rotation		
	Δx	Δy	Δz	Δα	Δβ	Δγ
1	-0.70±0.47 [0.81]	-1.56±1.03 [1.81]	-0.02±0.13 [0.11]	-0.06±0.06 [0.08]	0.02±0.07 [0.07]	0.81±0.53 [0.94]
2	0.01±0.11 [0.10]	0.07±0.19 [0.18]	0.03±0.09 [0.09]	0.05±0.08 [0.08]	-0.01±0.05 [0.04]	-0.02±0.11 [0.10]
3	0.09±0.11 [0.13]	0.59±0.46 [0.72]	0.42±0.39 [0.55]	0.46±0.37 [0.57]	0.09±0.14 [0.15]	-0.53±0.43 [0.66]

Table 4.3: Accuracy (average error values), repeatability (standard deviations) and root-mean-square errors (RMSE) of the automatic matching procedure with an idealized natural knee using three different pose environments with a model point density of 3,500 points. Each position was evaluated for ten initial positions.

4.2 Standardized In-vitro Environments

In an attempt to quantify accuracy of the system while in a working environment, controlled experiments were developed. The principle behind these tests was to image geometry with known structure and position. This was accomplished by employing highly precise geometry to enforce a relative distance and also by accurately translating known geometry.

4.2.1 Spheres

Method

A standardized test in the manner of Short, et al was performed consisting of eight spheres 12.70 mm in diameter each having a tolerance of ±0.01 mm[24]. Five of the spheres were stainless steel, two were ceramic (spheres 2 & 7), and one was tungsten (sphere 5). The spheres were arranged in a fixed pattern (Fig. 4-2) and imaged with the dual-orthogonal fluoroscopic system.

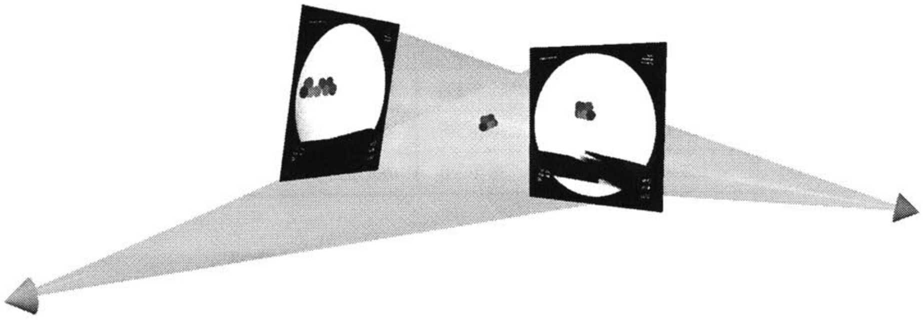


Figure 4-1: A virtual environment for the standardized in-vitro sphere test.

Using the solid modeling program a spherical model 12.70 mm in diameter was created and converted into a point cloud containing 2,500 points. Next, the model was placed centrally in the virtual fluoroscopic environment. Then, ten estimates for the initial pose were created for each component by perturbing the model from the placed configuration. The perturbations were created by randomly generating values for the pose variables within the range of ±20 mm using a Gaussian distribution. Next, the

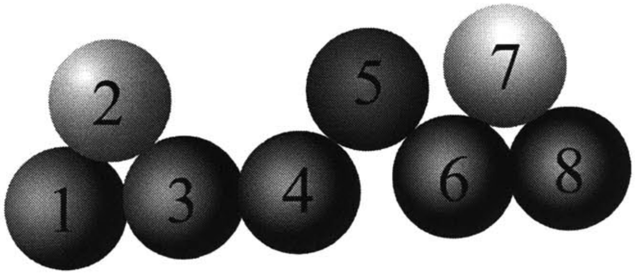
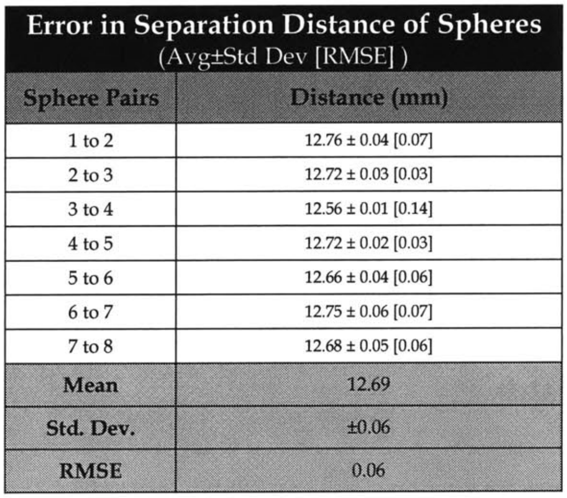


Figure 4-2: Geometry of standardized test. Spheres two and seven were ceramic, sphere five was tungsten and the remaining spheres were stainless steel. The spheres were stacked in the vertical plane.

models were matched using sixty of the projected outline points and the convergent matches were recorded. The accuracy of the matching algorithm was determined by comparing the distance between the adjacent matched spheres and the true distance of one diameter, 12.70 mm.



Error in Separation Distance of Spheres
(Avg±Std Dev [RMSE])

Sphere Pairs	Distance (mm)
1 to 2	12.76 ± 0.04 [0.07]
2 to 3	12.72 ± 0.03 [0.03]
3 to 4	12.56 ± 0.01 [0.14]
4 to 5	12.72 ± 0.02 [0.03]
5 to 6	12.66 ± 0.04 [0.06]
6 to 7	12.75 ± 0.06 [0.07]
7 to 8	12.68 ± 0.05 [0.06]
Mean	12.69
Std. Dev.	±0.06
RMSE	0.06

Table 4.4: Standardized test validating accuracy and precision of the dual-orthogonal fluoroscopic system under actual conditions with spheres of different materials using ten initial positions.

Results

For the standardized test, the distance between each adjacent pair of spheres was calculated (Table 4.4). A maximum distance of 12.76 mm was calculated between sphere 1 and sphere 2 and a minimum distance of 12.56 mm was calculated between sphere 5 and sphere 6. The average distance between matched pairs of adjacent spheres was 12.69 mm, 0.01 mm less than the actual distance data, with a standard deviation of ±0.06 mm. The average time for each match of a single sphere was 15 sec with 380 calls to the objective function.

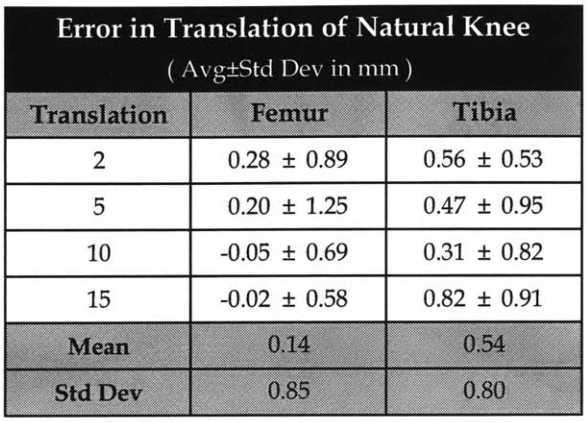
4.2.2 Natural Knee

Method

For the natural knee 3D medium point density models (15,000 points) of the femur and tibia were generated from an MRI of a cadaver. The cadaver knee was mounted in a tensile testing machine (QTest 5, MTS, Minneapolis, MN), which has a linear accuracy of 0.001 mm. The knee was then imaged as it was translated to five positions: 0, 2, 5, 10, 15 mm. The translation distance between poses was considered the gold standard. For each pose ten estimates for the initial guess were created for each model by perturbing them an initial match. The perturbations were created by randomly generating values for the pose variables within the range of ±20 mm and ±20° using a Gaussian distribution. Next, the models were matched using fifty of the projected outline points. The accuracy and repeatability of the optimized matching algorithm in reproducing the femoral and tibial position and orientation in 6DOF was recorded for each convergent match.

Results

Accuracy for the standardized natural knee tests was measured as the error between the translation of three body fixed points and the known displacement imposed by the tensile testing machine. The distance from the initial position was calculated for each position and listed in Table 4.5. The average error was found to be 0.14 mm for the femur and 0.54 mm for the tibia. Repeatability of all poses was found to be less than ±0.85 mm, where the sample standard deviation of these errors was selected as the measure of repeatability. The average time for matching a single pose was 90 seconds and the average number of calls to the objective function was 500.



Error in Translation of Natural Knee (Avg±Std Dev in mm)		
Translation	Femur	Tibia
2	0.28 ± 0.89	0.56 ± 0.53
5	0.20 ± 1.25	0.47 ± 0.95
10	-0.05 ± 0.69	0.31 ± 0.82
15	-0.02 ± 0.58	0.82 ± 0.91
Mean	0.14	0.54
Std Dev	0.85	0.80

Table 4.5: Standardized test validating accuracy and precision of the dual-orthogonal fluoroscopic system under actual conditions with a cadaver knee translated known distances.

4.3 In-vivo Environments

Testing the algorithm with actual subjects is the most important validation of performance. In-vivo validation utilizes images acquired from living subjects and provides a metric for determining the algorithm's repeatability. TKA subjects were imaged performing a lunge activity and this data was used for the validation.

4.3.1 TKA Components

Method

In-vivo images taken with the dual fluoroscopic system of the right knee of a patient after TKA. The images were acquired under IRB approval and with informed consent of the patient. The patient had a cruciate retaining component and images were taken during a lunge (NexGen CR TKA, Zimmer, Inc, Warsaw). Poses selected for matching were for images taken at 10° and 50° of flexion of the patient. Using the solid modeling program the component models were converted into point clouds containing 15,000 points. Next, the models were manually matched to the fluoroscopic contours in the virtual fluoroscopic environment. Then, for both flexion angles each model

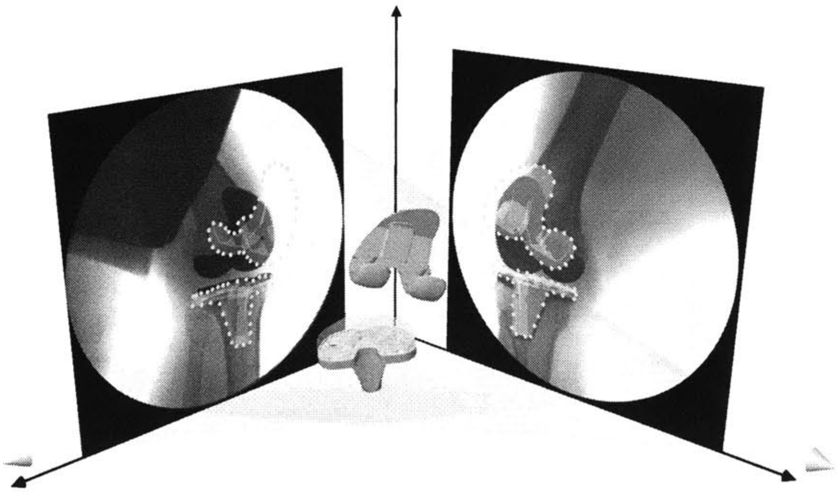
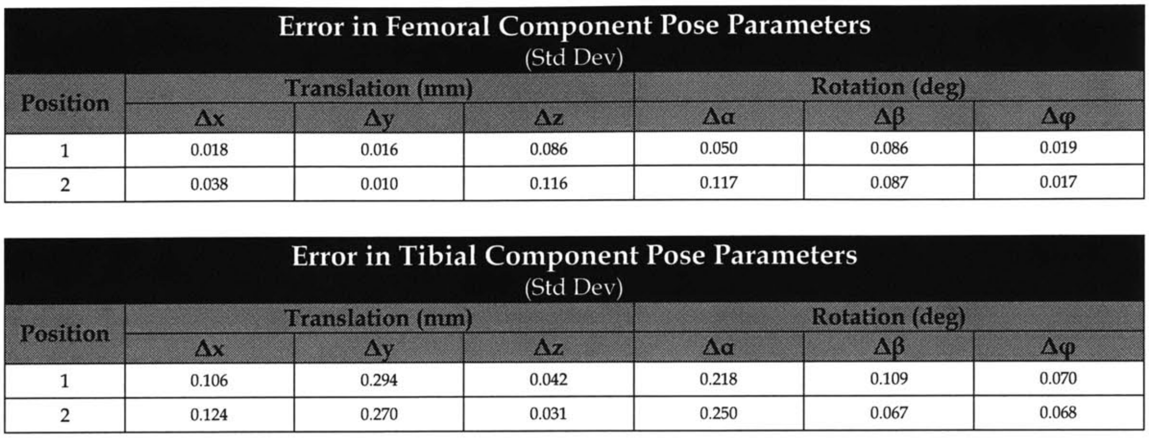


Figure 4-3: A virtual environment for an in-vivo TKA test.

was perturbed ten times to create estimates of the initial pose. The perturbations were created by randomly generating values for the pose variables within the range of ±20 mm and ±20° using a Gaussian distribution. Sixty projected outline points were used for the matches and the convergent solutions were recorded. Since the accurate position and orientation of the patient TKA is unknown, this test was designed to evaluate repeatability of the algorithm in determining the position and orientation of in-vivo TKA components.

Results

Repeatability was measured by comparing the variation of matched models when different initial guesses were used in the optimization procedure. Results are tabulated for two poses of both the tibial and femoral components in Table 4.6. Maximum translational deviation for both poses was ±0.12 mm for the femoral component and ±0.29 mm for the tibial component. Maximum angular deviation for both poses was ±0.12° for the femoral component and ±0.25° for the tibial component. The average time to match a single pose for each component was 500 sec with 600 calls to the objective function.



Error in Femoral Component Pose Parameters (Std Dev)						
Position	Translation (mm)			Rotation (deg)		
	Δx	Δy	Δz	Δα	Δβ	Δφ
1	0.018	0.016	0.086	0.050	0.086	0.019
2	0.038	0.010	0.116	0.117	0.087	0.017

Error in Tibial Component Pose Parameters (Std Dev)						
Position	Translation (mm)			Rotation (deg)		
	Δx	Δy	Δz	Δα	Δβ	Δφ
1	0.106	0.294	0.042	0.218	0.109	0.070
2	0.124	0.270	0.031	0.250	0.067	0.068

Table 4.6: Repeatability of the automatic matching procedure in reproducing the two in-vivo poses of the femoral and tibial components of a TKA patient for ten initial positions.

4.4 Summary

Results of the validation demonstrate the algorithms robustness and capability of realizing a pose from a variety of initial poses. Under idealized conditions, poses of a TKA system were recreated to within 0.02±0.01 mm and 0.02±0.03° for the femoral component and 0.07±0.09 mm and 0.16±0.18° for the tibial component. Using the idealized setup with a natural knee, pose for the femur was recreated to within 0.12±0.33 mm and 0.14±0.31° and the tibia pose was recreated to within 0.39±0.33 mm and 0.23±0.20°. By employing a standardized geometry with spheres, the translational accuracy and repeatability under actual conditions was found to be 0.01±0.06 mm. Using the standardized setup with a natural knee, pose for the femur was recreated to within 0.14±0.85 mm and the tibia pose was recreated to within 0.54±0.85 mm. Application of the optimized matching algorithm to a TKA patient showed that the pose of in-vivo TKA components can be repeatedly located, with standard deviations less than ±0.12 mm and ±0.12° for the femoral component and ±0.29 mm and ±0.25° for the tibial component. This methodology presents a useful tool that can be readily applied to the investigation of in-vivo kinematics.

Chapter 5

Discussion

5.1 Algorithm Operation

Idealized testing gave an empirical measure of the accuracy of the optimized matching algorithm and provided a controlled environment for examining the optimal point model density, number of matching points, and effect of initial poses. Results showed that increasing the model point density improved repeatability; however, accuracy remained roughly constant. Point density affects the sensitivity of rotational accuracy more than positional accuracy. For this reason model point density has a greater effect on the "axis-symmetric" tibial component. Since calculation time increases roughly linearly and error decreases quadratically, the point of diminishing returns for accuracy and repeatability of matching femoral and tibial components occurs at approximately 15,000 model points. In general, the optimal model point density occurs when the entire model surface is covered with an evenly distributed number of points that accurately capture the geometry.

The number of projected outline points used for matching affects the robustness of the matching algorithm and is dependent on model geometry. The optimal number of points should be sufficient to characterize a given projection geometry. However, characteristic points are difficult to define automatically. A simple method for circumventing manual placement is to automatically select a set of equally spaced points. Experience has shown that selecting a point every 4 mm on the projected outline of

TKA components adequately captures the geometric character. Interestingly, using large numbers of outline points does not significantly improve accuracy; however, fewer iterations are required for convergence.

The idealized testing environment also determined the effect of initial pose estimates on resulting component pose. It was found that perturbations within the range of 1-20 mm and 0.5-20° from the ideal position and orientation resulted in similar results for a range of model orientations. These data showed that the optimized matching process is forgiving of the initial pose estimate and allows for minimal operator intervention. Standardized testing allowed for evaluating the accuracy and repeatability of the optimized matching procedure with the actual dual-orthogonal fluoroscopic system.

Standardized geometries allowed for known relative poses, which is difficult to achieve using actual TKA components. Different materials for the spheres were used to simulate possible edge loss from overexposure and edge blooming from x-ray scatter due to differences in material density. In addition, cases of occlusion were also present, because of the geometry of the spheres. These image artifacts caused incorrect or incomplete segmentation of the fluoroscopic images; however, most artifacts were not severe in both views and the combined information of the two orthogonal views reduced the difficulty of matching. The results of these tests showed that the accuracy and repeatability for the standardized and idealized tests had similar orders of magnitude and that the different materials did not disrupt the optimized matching algorithm's ability to recreate the spheres' pose. Furthermore, noise, occlusion and distortion may affect the quality of the edges, but due to the geometry of the imaging system and "fitting" nature of the algorithm, these errors are often obviated. Edge quality is of greater concern when imaging natural joints and soft tissue; however, the positive results of this study allude to the possibility that this optimization method could be applied to in-vivo kinematics of intact knees with further validation.

To prove the capabilities of the optimized matching algorithm for use with TKA kinematics, the method was applied to an actual TKA patient. The data demonstrated that the optimized matching procedure was highly repeatable when different initial guesses were used. Repeatability for the femoral component was better than the tibial component because of the symmetry of the tibial component. This phenomenon was also observed in the idealized testing (Table 4.1). Symmetric objects are more difficult to match than objects of irregular geometry because of the reduced sensitivity in the projected silhouette. As sensitivity decreases, the optimization routine is less likely to converge.

5.2 Comparison with Previous Algorithms

Fluoroscopic techniques have been used extensively in recent years for determining in-vivo TKA kinematics[2, 15, 30, 32]. These techniques offer advantages over roentgen stereophotogrammetry (RSA) and conventional X-ray because of the reduced radiation exposure and non-invasive methods. In the pursuit of improved accuracy using fluoroscopy, recent studies have developed a dual-orthogonal fluoroscopic system for determining 6DOF TKA and normal knee kinematics when combined with MR image-based 3D knee models[4, 9, 18]. These methods have been shown to be accurate using a manual process, but for investigation of joint motion, which requires many fluoroscopic images, an optimized image matching process is desired[1, 19]. The automated image matching algorithm for determining 6DOF poses presented in this thesis compares favorably to previous fluoroscopic methods which have employed a variety of techniques for determining the pose of 3D objects from 2D images, see Tables 5.1 & 5.2.

These methods can be broadly grouped into either template matching or hypothesize and test methods[2, 10, 14]. Template matching techniques compare segmented outlines from fluoroscopic outlines to a library of previously calculated silhouettes of component models. Hypothesize and test methods first "hypothesize" a location and orientation of a model, and then test the validity of the pose based on fluoroscopic images.

Template matching techniques were implemented in early works by Banks and Hoff[2, 10]. With the evolution of computing power these techniques have recently been eclipsed by hypothesize and test methods. Hypothesize and test methods can be further classified based on the type of "test". The most common tests are iterative closest point (ICP) [6, 7], iterative inverse point (IIP)[16, 30, 32], and digitally reconstructed radiograph (DRR)[20, 24, 31]. The ICP method minimizes the distance between projected model points and points on the fluoroscopic outlines. IIP methods minimize the distance between a model's surface and the rays connecting points on the fluoroscopic outline to the virtual x-ray source. DRR methods use ray-tracing algorithms to render simulated fluoroscopic images of TKA components and correlate the intensity values of the pixels and matching of segmented features to the actual fluoroscopic images.

IIP and DRR methods are significantly more complex than ICP algorithms, but are gaining ground due to the recent advances in computing power and accessibility of high-power graphics software. ICP and IPP methods may be more susceptible to segmentation errors, but provide a more stable optimization problem than DRR methods[21].

Algorithm Accuracy Using Data from Simulated Cases						
Author	Accuracy				Algorithm Type	
	In-plane		Out-of-plane			
	mm	deg	mm	deg		
Bingham	0.07	0.16	0.07	0.16	Dual Plane	Hypothesis & Test (ICP)
Lavallee[16]	0.43	0.32	0.43	0.32	Dual Plane	Hypothesis & Test (IIP)
Kaptein[13]	0.15	0.07	0.15	0.07	Dual Plane	Hypothesis & Test (IIP)
Zuffi[32]	0.40	0.40	2.00	0.40	Single Plane	Hypothesis & Test (IIP)
Yamazaki[30]	0.08	0.20	0.85	0.20	Single Plane	Hypothesis & Test (IIP)
Banks[2]	0.20	0.30	2.00	0.30	Single Plane	Template Matching
Hoff[10]	0.37	0.21	1.51	1.35	Single Plane	Template Matching

Table 5.1: A comparison of the reported accuracy of the major algorithms when operating on ideal data.

It should be noted that single plane implementation of these methods are limited in their accuracy for determining accurate 6DOF TKA kinematics because of the discrepancy between in-plane and out-of-plane accuracy. By looking at Tables 5.1 & 5.2 one can see the advantage of using additional images in the matching procedure.

Algorithm Precision Using Data from In-vivo TKA Cases						
Author	Accuracy				Algorithm Type	
	In-plane		Out-of-plane			
	mm	deg	mm	deg		
Bingham	0.29	0.25	0.29	0.25	Dual Plane	Hypothesis & Test (ICP)
Kaptein[12]	0.06	0.05	0.14	0.10	Dual Plane	Hypothesis & Test (ICP)
Valstar[28]	0.16	0.34	0.16	0.34	Dual Plane	Hypothesis & Test (NOA)
You[31]	0.23	1.20	0.23	1.20	Dual Plane	Hypothesis & Test (DRR)
Mahfouz[20]	0.66	1.50	3.21	1.50	Single Plane	Hypothesis & Test (DRR)
Tomazevic[27]	1.24	1.35	N/A	N/A	Single Plane	Hypothesis & Test (DRR)
Yamazaki[30]	0.12	0.70	1.44	0.70	Single Plane	Hypothesis & Test (IIP)
Fukuoka[6]	0.28	0.33	4.17	0.26	Single Plane	Hypothesis & Test (ICP)
Banks[2]	0.50	1.10	6.60	1.10	Single Plane	Template Matching
Hoff[10]	0.13	0.30	0.26	0.30	Single Plane	Template Matching
Kanisawa[11]	1.20	0.80	4.00	0.80	Single Plane	Template Matching

Table 5.2: A comparison of the reported repeatability of the major algorithms when operating on in-vivo TKA data.

A recent parametric analysis of single imaging techniques showed that for a desired accuracy in the out-of-plane direction, the in-plane accuracy needed to be at least an order of magnitude better[9, 19]. Another recent article by Garling et al. illustrated that with an in-plane accuracy of less than 0.17 mm in translation out-of-plane error could reach 1.9 mm [7]. Other studies using single fluoroscopic techniques have also reported similar results[2, 6, 20, 30].

In order to improve on previous methods, this study implemented a modified ICP method that matches projected model points to spline curves on two orthogonal image intensifiers. Using a dual-orthogonal fluoroscopic system significantly improves accuracy over single fluoroscopic systems. This is because out-of-plane errors of one fluoroscope are the in-plane errors of the other fluoroscope. In addition, the use of two orthogonal contours for matching significantly amplifies the global minima and the use of splines smoothes the matching space, thus improving algorithm convergence. Results from this study confirm statements from similar studies that the use of dual-orthogonal fluoroscopy can dramatically enhance accuracy for true sub-millimeter accuracy of in-vivo TKA kinematics in 6DOF[1, 12, 22, 24, 28, 31]. Run-times are also favorable at around four to eight minutes and compare with similar methods[12, 16, 31].

It is important to state that using dual images is not new. Marker and model based RSA, which use two X-ray beams, has been used to determine knee, ankle, and shoulder kinematics[13, 24, 31]. Application of two X-rays has also been used to determine normal knee kinematics combined with CT image-based knee models[1, 22, 28, 31]. Not surprisingly, these studies presented accuracies similar to the dual-orthogonal fluoroscopic methodology. However, the higher radiation doses, stationary equipment and limited field-of-view associated with conventional X-ray present difficulties when determining in-vivo TKA motion. The dual-orthogonal fluoroscopic technique bridges the accuracy of RSA and the minimal invasiveness of fluoroscopy, to bring together the best attributes of both methods. This synergy produces an improved tool for investigating joint kinematics. Combining the dual-orthogonal fluoroscopic system with the optimized image matching procedure developed in this research provides a powerful tool for processing large quantities of image sets rapidly.

5.3 Single vs. Dual Fluoroscopy

The difference in accuracy of single versus dual fluoroscopy is not generally contended. It is well known that having a single viewpoint greatly reduces depth perception. If you attempt to shoot baskets with one eye covered you will have a difficult time, because you lose depth accuracy. This is similar in the case of single and dual fluoroscopy. With a single viewpoint, the out-of-plane accuracy must be worse than the in-plane accuracy; however, when utilizing two orthogonal viewpoints the out-of-plane accuracy of one view is the in-plane accuracy of the second view. A large question is to what extent the out-of-plane accuracy is degraded by using a single fluoroscope. By using the geometry from a clinical fluoroscope one can use similar triangles to show that for an arbitrary object out-of-plane error will be an order of magnitude greater than in-plane error (Fig. 5-1), which has been corroborated by Garling et al. [7].

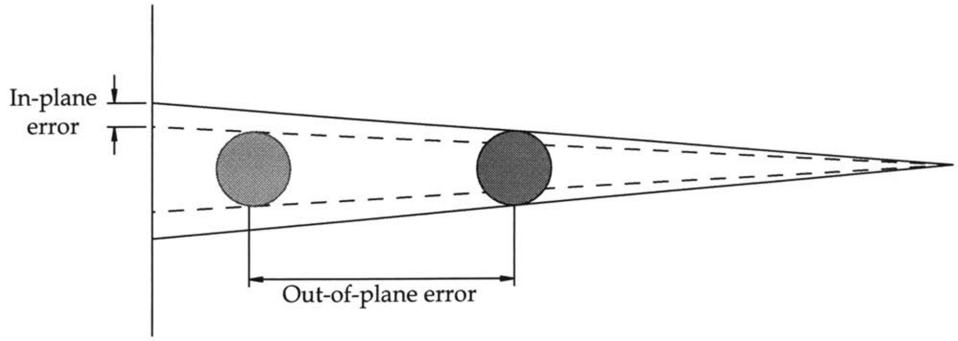


Figure 5-1: A schematic showing the relation between in-plane and out-of-plane error when employing a single viewpoint.

In order to determine the effect of multiple planes on automatic matching a series of idealized tests were run. The first test was constructed to determine the sensitivity required of the optimization routine to produce accurate results when using single or dual plane data. The second test used a TKA femoral component in several poses to determine the effect of geometry on sensitivity. The idealized virtual fluoroscopic environments for these tests were constructed with parameters taken from in-vivo measurements.

The first test used a solid modeling program to construct a spherical model 50 mm in diameter and containing 9,500 points. Next, the model was placed centrally in the virtual fluoroscopic environment and virtual outlines were created to produce the "gold standard". Then, ten initial poses were created by perturbing the model from the placed configuration. The perturbations were created by randomly generating values for the pose variables within the range of ±20 mm using a Gaussian distribution. Next, each pose was matched using a single contour and then with both contours. Additionally, for each match the tolerance of the optimization routine was adjusted sequentially with the values: 0.05, 0.005, 0.0005 and 0.000005. This effectively controls the maximum error allowed in the objective function Eq. 3.6. The accuracy and repeatability of the optimized matching algorithm in reproducing the sphere position was recorded for each convergent match.

The results of the first test, shown in Table 5.3 and Figure 5-2, confirm that with simple geometry the addition of a view greatly enhances the matching process.

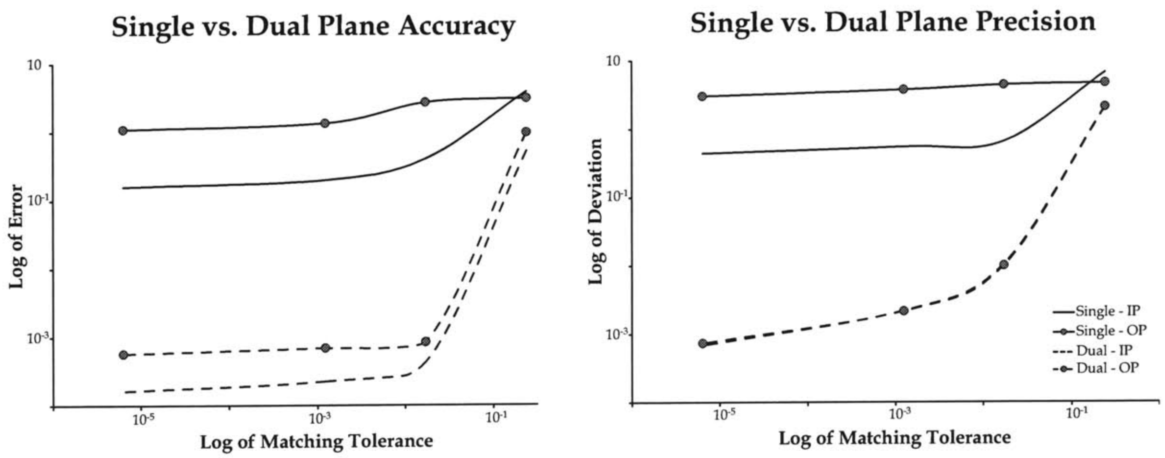
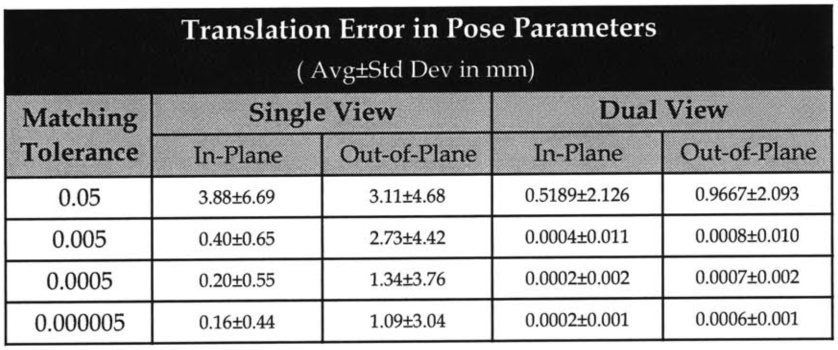


Figure 5-2: Comparison of the accuracy and repeatability of automatic matching to single vs. dual plane data when using spherical geometry.

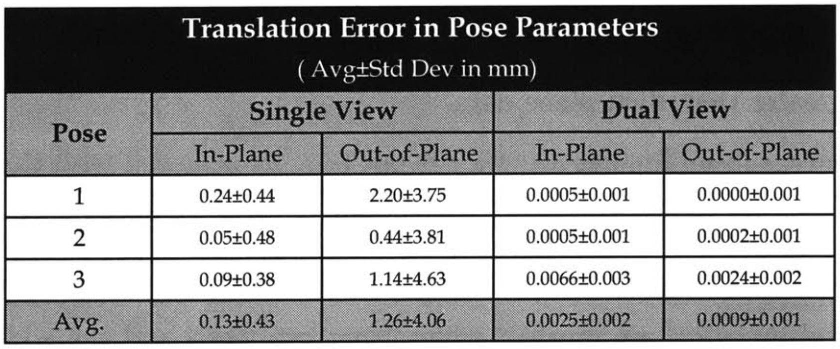


Translation Error in Pose Parameters (Avg±Std Dev in mm)				
Matching Tolerance	Single View		Dual View	
	In-Plane	Out-of-Plane	In-Plane	Out-of-Plane
0.05	3.88±6.69	3.11±4.68	0.5189±2.126	0.9667±2.093
0.005	0.40±0.65	2.73±4.42	0.0004±0.011	0.0008±0.010
0.0005	0.20±0.55	1.34±3.76	0.0002±0.002	0.0007±0.002
0.000005	0.16±0.44	1.09±3.04	0.0002±0.001	0.0006±0.001

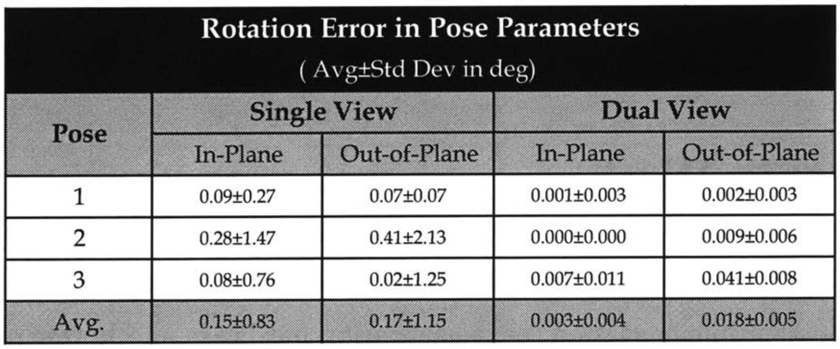
Table 5.3: Data comparing the effect of single vs. dual plane data on automatic matching when using spherical geometry.

By examining Figure 5-2, one can see out-of-plane error is consistently an order of magnitude greater than in-plane error when only a single view is used. It is also important to notice that while precision increases with tighter tolerance for the dual-view data, precision remains constant for the single view. For this particular test one can see that the accuracy of the automatic matching is compromised if only a single view is used, and sub-millimeter accuracy is feasible.

It has been argued that irregular geometry intensifies sensitivity of the matching process, which would increase the accuracy of single plane matching. The second test was developed to determine the extent of the increased sensitivity and whether this is enough to produce sub-millimeter accuracy when using single plane data. The second test was constructed by placing a TKA femoral component having 14,000 points into three different virtual environments. Next, the model was placed in a position approximating 90° of knee flexion in the virtual fluoroscopic environment and virtual outlines were created to produce the "gold standard". Then, ten initial poses were created by perturbing the model from the placed configuration. The perturbations were created by randomly generating values for the pose variables within the range of ±20 mm using a Gaussian distribution. Next, each pose was matched using a single contour and then with both contours and utilizing an optimization tolerance of 0.0005.



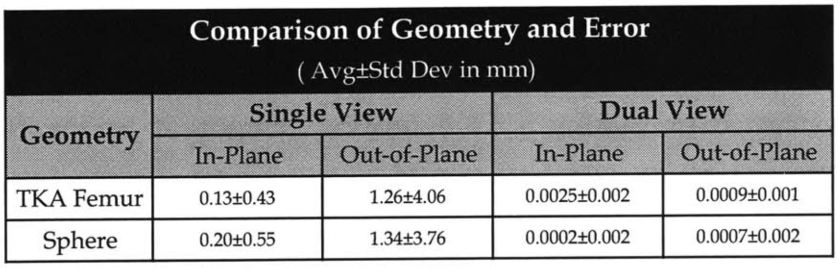
Translation Error in Pose Parameters (Avg±Std Dev in mm)				
Pose	Single View		Dual View	
	In-Plane	Out-of-Plane	In-Plane	Out-of-Plane
1	0.24±0.44	2.20±3.75	0.0005±0.001	0.0000±0.001
2	0.05±0.48	0.44±3.81	0.0005±0.001	0.0002±0.001
3	0.09±0.38	1.14±4.63	0.0066±0.003	0.0024±0.002
Avg.	0.13±0.43	1.26±4.06	0.0025±0.002	0.0009±0.001



Rotation Error in Pose Parameters (Avg±Std Dev in deg)				
Pose	Single View		Dual View	
	In-Plane	Out-of-Plane	In-Plane	Out-of-Plane
1	0.09±0.27	0.07±0.07	0.001±0.003	0.002±0.003
2	0.28±1.47	0.41±2.13	0.000±0.000	0.009±0.006
3	0.08±0.76	0.02±1.25	0.007±0.011	0.041±0.008
Avg.	0.15±0.83	0.17±1.15	0.003±0.004	0.018±0.005

Table 5.4: Data comparing the effect of single vs. dual plane data on automatic matching when using geometry from a TKA femoral component.

Table 5.4 shows the results of the second test. These results do not show a significant change in matching ability for either the single or dual plane data. This is better illustrated in Table 5.5, which presents a side by side comparison of the



Comparison of Geometry and Error

(Avg±Std Dev in mm)

Geometry	Single View		Dual View	
	In-Plane	Out-of-Plane	In-Plane	Out-of-Plane
TKA Femur	0.13±0.43	1.26±4.06	0.0025±0.002	0.0009±0.001
Sphere	0.20±0.55	1.34±3.76	0.0002±0.002	0.0007±0.002

Table 5.5: Comparison of the single and dual plane data on automatic matching when using geometry from spherical and TKA femoral component geometries.

matching results for the spherical and femoral TKA component geometries. This shows that single-plane sensitivity is not increased when matching objects of irregular geometry. It should also be noted that the rotational accuracy suffers when only a single view is employed; even when highly irregular geometries are used. The data from this test show that rotational errors with single plane data are an order of magnitude greater than dual plane data.

In summary, sub-millimeter accuracy is not feasibly achieved with this automatic matching algorithm when employing only a single view. The highest accuracy of the automatic matching algorithm occurs when two views are utilized. Furthermore, sensitivity and precision are greatest when using two views and the additional view reduces the rotational error of the automatic matching.

5.4 Future Work

This work provides a solid algorithm, which can be improved upon through more efficient software implementation, next-generation hardware and better minimization routines. The current automatic matching algorithm is coded in Matlab software, which is a run-time language. Coding the algorithm into a compilable program would greatly increase the speed of the algorithm. Also, the algorithm lends itself well to utilizing the capabilities of modern graphics hardware and software libraries. Advancements can also be made as the imaging hardware improves. Higher quality images will allow for more accurate segmentation and increased precision of the method.

As computational power increases and the code becomes more efficient search based minimization will become more feasible. Global search methods have been drastically hindered by the small search pool required by time constraints. With large populations, genetic algorithms, simulated annealing and direct search methods should produce excellent results. Using these optimization methods will possibly make texture and intensity matching a robust measure, which can then be used in conjunction with edge matching for improved automatic matching. Automatic matching will certainly progress as hardware and software improves. It should also be noted that this algorithm has its origin in machine vision. Advances in technology and application of this algorithm could potentially be used with high accuracy robotics applications. Furthermore, these algorithms might help to one day provide realtime visualization of in-vivo kinematics, and become a beneficial tool for clinicians to provide patient specific surgeries.

Appendix A

Matlab Source Code

A.1 Environment Setup

A.1.1 correctImage.m

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% Program Name:  correctImage
%% File Name   :  correctImage.m
%%
%% Arguments   :  selectedPoints .n x 2 matrix -  Ordered control points from
%%                                                distorted image.
%%                knownPoints. . .n x 2 matrix -  Ordered control points from
%%                                                original geometry.
%%                image. . . . . .p x q matrix -  Intensity values that make up
%%                                                the distorted image.
%%
%% Output      :  correctedImage .p x q matrix -  Intensity values that make up
%%                                                the corrected image.
%%
%% Description :  An ordered set of matching distorted and original control points
%%                are passed in.  The first point defines the center of the image
%%                and the second point the orientation.  A minimum of 15 control
%%                points is required.  Using these points the distorted image
%%                is corrected to match the original control points.
%%
%% Example     :  correctedImage = correctImage( selectedPoints, knownPoints, image)
%%
%/////////////////////////////////
%/ Bioengineering Laboratory     /
%/ Jeff Bingham                  /
%/ March 24, 2005                /
%/ Revision: March 24, 2005      /
%/////////////////////////////////
function correctedImage = correctImage( selectedPoints, knownPoints, image)

sz = size(image);

offset = selectedPoints(1,:)-knownPoints(1,:);

knwVec = knownPoints(2,:)-knownPoints(1,:);
knwVec = knwVec/norm(knwVec);
selVec = selectedPoints(2,:)-selectedPoints(1,:);
selVec = selVec/norm(selVec);
theta = acos(dot(selVec,knwVec));
rotation = [ [  cos(theta)  sin(theta) ]
             [ -sin(theta)  cos(theta) ] ];

knownPointsAligned = (rotation*[ knownPoints(:,1)-knownPoints(1,1) ...
                                 knownPoints(:,2)-knownPoints(1,2) ]')';
knownPointsAligned = [ knownPointsAligned(:,1)+knownPoints(1,1)+offset(1) ...
                       knownPointsAligned(:,2)+knownPoints(1,2)+offset(2)];


TFORM1 = cp2tform(selectedPoints,knownPointsAligned,'polynomial',4);

correctedImage = imtransform(img, TFORM1, 'bicubic', 'size', sz, ...
                             'udata', [ 0 sz(1) ], 'vdata', [ 0 sz(2) ], ...
                             'xdata', [ 0 sz(1) ], 'ydata', [ 0 sz(2) ]);
```

A.1.2 assembleMesh.m

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% Program Name:  assembleMesh
%% File Name   :  assembleMesh.m
%%
%% Arguments   :  pointMatrix. . .q x 1 cell    -  Each cell contains a r x 3
%%                                                 matrix containing ordered.
%%                                                 points that make up contours
%%                closed . . . . .boolean       -  A boolean value for determining
%%                                                 how the mesh should close at an
%%                                                 endpoint.
%%                gap. . . . . . .real          -  Specifies the maximum gap to
%%                                                 still create a mesh face.
%%
%% Output      :  faces. . . . . .m x 3 matrix -  A triangle made up of the
%%                                                 indicies of verts.
%%                verts. . . . . .n x 3 matrix -  The vertices of the mesh.
%%
%% Description :  From a level set of curves that have been divided into points a
%%                mesh is constructed.
%%
%% Example     :  [ faces, verts ] = assembleMesh(pointMatrix, closed, gap)
%%
%/////////////////////////////////
%/ Bioengineering Laboratory     /
%/ Jeff Bingham                  /
%/ February 23, 2006             /
%/ Revision: June 5, 2005        /
%/////////////////////////////////
function [ faces, verts ] = assembleMesh(pointMatrix, closed, gap)

verts=[];
faces=[];

sliceAbove = pointMatrix{1};
verts = sliceAbove;
aI = [ 1:size(sliceAbove,1), 1 ];
sz=0;

for i=2:length(pointMatrix)
    T=[];
    sliceBelow = sliceAbove;
    sliceAbove = pointMatrix{i};
    verts = [verts; sliceAbove];
    nBelow=size(sliceBelow,1);
    nAbove=size(sliceAbove,1);
    bI = aI;

    rB = floor(rand*(nBelow-1))+1;
    bI = [ bI(rB):nBelow, 1:bI(rB) ];
    [abV, abI] = closestPt(sliceBelow(bI(1),:), sliceAbove);
    aI = [ abI:nAbove, 1:abI ];

    if(dot(sliceAbove(aI(2),1:2)-sliceAbove(aI(1),1:2),sliceBelow(bI(2),1:2)-sliceBelow(bI(1),1:2))<0)
        if(dot(sliceAbove(aI(end),1:2)-sliceAbove(aI(end-1),1:2),sliceBelow(bI(end),1:2)-sliceBelow(bI(end-1),1:2))<0)
            aI=flipdim(aI,2);
        end
    end

    indxBelow = 1;
    nxtBelow = indxBelow+1;
    indxAbove = 1;
    nxtAbove = indxAbove+1;
    shortFS=1;
    shortBS=1;

    while(indxBelow<=nBelow || indxAbove<=nAbove)
```

```

        distFS = (sliceAbove(aI(nxtAbove),1)-sliceBelow(bI(indxBelow),1))^2 + ...
            (sliceAbove(aI(nxtAbove),2)-sliceBelow(bI(indxBelow),2))^2 + ...
            (sliceAbove(aI(nxtAbove),3)-sliceBelow(bI(indxBelow),3))^2 ;
        distBS = (sliceAbove(aI(indxAbove),1)-sliceBelow(bI(nxtBelow),1))^2 + ...
            (sliceAbove(aI(indxAbove),2)-sliceBelow(bI(nxtBelow),2))^2 + ...
            (sliceAbove(aI(indxAbove),3)-sliceBelow(bI(nxtBelow),3))^2 ;

        if(shortFS*distFS<shortBS*distBS)
            T = [T;[ aI(indxAbove)+nBelow bI(indxBelow) aI(nxtAbove)+nBelow ]];
            if(~closed)
                if((aI(indxAbove)==1&& aI(nxtAbove)==nAbove)||(aI(indxAbove)==nAbove&& aI(nxtAbove)==1))
                    T(end,:)=[];
                    distFS = 0;
                end
                if(distFS>gap && ~isempty(T))
                    T(end,:)=[];
                end
            end
            indxAbove = indxAbove+1;
            nxtAbove = indxAbove+1;
        else
            T = [T;[ aI(indxAbove)+nBelow bI(indxBelow) bI(nxtBelow) ]];
            if(~closed)
                if((bI(indxBelow)==1&& bI(nxtBelow)==nBelow)||(bI(indxBelow)==nBelow && bI(nxtBelow)==1))
                    T(end,:)=[];
                    distBS = 0;
                end
                if(distBS>gap && ~isempty(T))
                    T(end,:)=[];
                end
            end
            indxBelow = indxBelow+1;
            nxtBelow = indxBelow+1;
        end

        if(indxAbove > nAbove)
            indxAbove = nAbove+1;
            nxtAbove = nAbove+1;
            shortBS=0;
        end

        if(indxBelow > nBelow)
            indxBelow = nBelow+1;
            nxtBelow = nBelow+1;
            shortFS=0;
        end
    end
    faces=[faces;T+sz];
    sz=sz+size(sliceBelow,1);
end
```

A.2 Automatic Matching

A.2.1 FINDPOSE.m

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% Program Name:  FINDPOSE
%% File Name   :  FINDPOSE.m
%% Arguments   :  model. . . . . . . m x 3 matrix - Each row is a point
%%                                   in 3-space describing 3D model.
%%                modelAxes. . . . . 4 x 3 matrix - Each row is a point
%%                                   Row 1) Origin, 2) X-axis, 3) Y-axis,
%%                                   4) Z-axis.
%%                silhouetteXZ. . . .m x 3 matrix - Each row is a point
%%                                   in 3-space describing match-contour.
%%                silhouetteYZ. . . .m x 3 matrix - Each row is a point
%%                                   in 3-space describing match-contour.
%%                environment. . . . 8 x 3 matrix - Each row is a point.
%%                                   Row 1  : XZ Viewpoint
%%                                   Row 2-4: XZ Plane location
%%                                   Row 5  : YZ Viewpoint
%%                                   Row 6-8: YZ Plane location
%%                settings. . . . . .12 x 1 cell  - Contains all parameters
%%                                   for optimization convergance and
%%                                   vizualization. (initialGuess,
%%                                   lowerBound, upperBound, DiffMinChange,
%%                                   DiffMaxChange, TolX, TolFun, MaxIter,
%%                                   MaxFunEvals, Display, gridSize,
%%                                   OutputFcn)
%%
%% Output      :  PARAMETERS . . . . 4 x 1 cell   - Contains the
%%                                   optimization return parameters: x -
%%                                   x - vector of six scalars that
%%                                   describe the translation and rotation
%%                                   necessary to move specified model to
%%                                   a configuration matching the specified
%%                                   contours.  [ x y z alpha beta gamma ]
%%                                   fval - the final performance index
%%                                   value.
%%                                   exitflag - whether the optimization
%%                                   routine converged, or exited properly.
%%                                   output - various optimization
%%                                   information like number of iterations,
%%                                   etc.
%%
%% Description :  Function takes a set of 3D points describing a 3D model
%%                and using the algorithm listed below determines the
%%                required rotation and translation to recreate the pose of
%%                the 3D model from the supplied 2D contours.
%%
%% Algorithm   :  1) Initialize envronment geometry.
%%                2) Begin Optimization
%%                        a) Calculate new translation and rotation values.
%%                        b) Transform model.
%%                        c) Project and flatten points.
%%                        e) Outline points.
%%                        f) Compute distance error.
%%                                i) Find minimum distance of silhouette curve to a point.
%%                                ii) Repeat (i) for all outline points.
%%                                iii) Return total normalized "error" distance.
%%                        g) Test if distance is sufficiently small.  If so stop else goto (a)
%%                4) Return transformation values that fit the 3D model to the 2D outlines best.
%%
%%
%% Example     : PARAMETERS = FINDPOSE( model, modelAxes, silhouetteXZ, ...
%%                                      silhouetteYZ, environment, ...
%%                                      settings )
%%
%/////////////////////////////////
%/ Bioengineering Laboratory     /
%/ Jeff Bingham                  /
%/ November 6, 2004              /
%/ Revision: September 16, 2005  /
%/////////////////////////////////

function PARAMETERS = FINDPOSE( model, modelAxes, silhouetteXZ, silhouetteYZ, environment, settings )

centerofrotation = modelAxes(1,:);

flatSilXZ = silhouetteXZ(:,1:2);
flatSilYZ = silhouetteYZ(:,1:2);

[ viewpointXZ, normalVectorXZ, normalMagnitudeXZ, flatXZR, ...
               flatXZT ] = environSetup(environment(1:4,:));
[ viewpointYZ, normalVectorYZ, normalMagnitudeYZ, flatYZR, ...
               flatYZT ] = environSetup(environment(5:8,:));

x0 = settings.initialGuess;
LB = settings.lowerBound;
UB = settings.upperBound;
gridSize = settings.gridSize;

minOPTS = optimset('DiffMinChange', settings.DiffMinChange, ...
                   'DiffMaxChange', settings.DiffMaxChange, ...
                   'TolX', settings.TolX, ...
                   'TolFun', settings.TolFun, ...
                   'MaxIter', settings.MaxIter, ...
                   'MaxFunEvals', settings.MaxFunEvals, ...
                   'Display', settings.Display, ...
                   'LargeScale', 'off');

if(isequal(settings.OutputFcn, 'on'))
    minOPTS.OutputFcn = @lookFun;
end

[x,fval,exitflag,output] = fmincon(@objFun, x0, [],[],[],[], LB, UB, [], minOPTS);
PARAMETERS.x = x;
PARAMETERS.fval = fval;
PARAMETERS.exitflag = exitflag;
PARAMETERS.output = output;

    %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
    % OBJECTIVE FUNCTION - START
    %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

    function error = objFun( params )
        T = [ params(1) params(2) params(3) ];
        R = eulerAngles([ params(4) params(5) params(6) ]);

        newModel=transform(model,-centerofrotation);
        newModel = transform(newModel, R, centerofrotation+T);

        pXZ = project( newModel, viewpointXZ, normalVectorXZ, ...
                       normalMagnitudeXZ, flatXZR, flatXZT);
        pYZ = project( newModel, viewpointYZ, normalVectorYZ, ...
                       normalMagnitudeYZ, flatYZR, flatYZT);
        olXZ = bound(pXZ, gridSize);
        olYZ = bound(pYZ, gridSize);

        distXZ = 0;
        distYZ = 0;
        for i=1:size(olXZ,1)
            distXZ = distXZ + dist2curve( flatSilXZ, olXZ(i,:));
        end
        for i=1:size(olYZ,1)
            distYZ = distYZ + dist2curve( flatSilYZ, olYZ(i,:));
        end

        if(size(olXZ,1)<2)
```

```
            error=100;
        else
            error = distXZ/size(olXZ,1);
        end
        if(size(olYZ,1)<2)
            error=error+100;
        else
            error = error + distYZ/size(olYZ,1);
        end
    end
    %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
    % OBJECTIVE FUNCTION - END
    %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

    %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
    % VISUALIZATION - START
    %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
    function stop=lookFun(x,optimValues,state)
        stop=[];
        switch state
            case 'iter'
                T = [x(1) x(2) x(3)];
                R = eulerAngles([x(4) x(5) x(6)]);
                newModel=transform(model,-centerofrotation);
                newModel = transform(newModel, R, centerofrotation+T);
                pXZ = project( newModel, viewpointXZ, normalVectorXZ, ...
                               normalMagnitudeXZ, flatXZR, flatXZT);
                pYZ = project( newModel, viewpointYZ, normalVectorYZ, ...
                               normalMagnitudeYZ, flatYZR, flatYZT);
                olXZ = bound(pXZ, gridSize);
                olYZ = bound(pYZ, gridSize);

                figID=figure(313);
                hold off;
                subplot(2,2,1)
                plot(flatSilXZ(:,1),flatSilXZ(:,2),'k.');
                hold on;
                plot(pXZ(:,1),pXZ(:,2),'m.');
                plot(olXZ(:,1),olXZ(:,2),'go');
                title(num2str(x(1:3)))
                axis equal
                hold off;
                subplot(2,2,2)
                plot(flatSilYZ(:,1),flatSilYZ(:,2),'k.');
                hold on;
                plot(pYZ(:,1),pYZ(:,2),'m.');
                plot(olYZ(:,1),olYZ(:,2),'go');
                title(num2str(x(4:6)))
                axis equal
                drawnow;
           case 'done'
        end
    end
    %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
    % VISUALIZATION - END
    %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
end
```

A.2.2 bound.m

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% Program Name:  bound
%% File Name    :  bound.m
%% Arguments    :  points. . .m x 2 matrix - coordinates of the points to be
%%                                          bounded.
%%                 grid. . . .scalar specifying number of divisions of
%%                            projection grid.
%%
%% Output       :  BOUNDARY. .m x 2 matrix - coordinates of points on
%%                                          the outline.
%%
%% Outside
%% Functions
%% Called       : [ PERIMETER, NORMAL_DIRECTION ] = contour( BlackAndWhiteImage, StartingPixel)
%%                      Function finds perimeter pixels of bw images.
%%                  FARTHEST_POINT = farthest( points, quadrant )
%%                      Function finds the point in the specified quadrant
%%                      that is farthest from the centroid of the points.
%%
%% Description :  This function takes a cloud of 2D points, digitizes them
%%                so that each point represents a pixel on a grid, then
%%                finds the perimeter of these pixels.  Next the
%%                "outermost" point in each of the pixels is found. These
%%                points are returned as points bounding the 2D point
%%                cloud.
%%
%%
%% Example      :  BOUNDARY = bound(pt_cloud, grid)
%%
%////////////////////////////////
%/ Bioengineering Laboratory    /
%/ Jeff Bingham                 /
%/ October 22, 2004             /
%/ Revision: January 25, 2005   /
%////////////////////////////////

function BOUNDARY = bound(points, grid)

n = size(points,1);                                 % Number of points in point-cloud.

[minX minXindx] = min(points(:,1));                 % Find the left-most point and its index.
[minY minYindx] = min(points(:,2));                 % Find the bottom-most point and its index.
[maxX maxXindx] = max(points(:,1));                 % Find the right-most point and its index.
[maxY maxYindx] = max(points(:,2));                 % Find the top-most point and its index.


width = maxX-minX;                                  % Determine the width of a bounding rectangle.
height = maxY-minY;                                 % Determine the height of a bounding rectangle.

if width>height                                     %
    gridLength=height/grid;                         % Determine the smallest grid size for a specified
else                                                % number of divisions based on the orientation of the
    gridLength=width/grid;                          % point cloud.
end                                                 %

try
pts=[points(:,1)-minX points(:,2)-minY];
if(~(gridLength==0))

    rows = ceil(height/gridLength);                 % Determine the number of "pixels"
                                                    % in the height direction.
    cols = ceil(width/gridLength);                  % Determine the number of "pixels"
                                                    % in the width direction.
    pts=pts/gridLength;
    pts=[ pts(:,1)+0.5 rows-pts(:,2)+0.5 ];

    binaryImage = zeros(rows,cols);                 % Build a blank "grid canvas" to plot points on.
    cellSorted = cell(rows,cols);                   % Build a matrix of vectors to hold
                                                    % the list of points in each "pixel"

    for i=1:n
        r= round(pts(i,2));
        c= round(pts(i,1));
        if(r>rows)
            r=rows;
        end
        if(c>cols)
            c=cols;
        end
        cellSorted{r,c} = [cellSorted{r,c} i ];     % Record which points go to which pixel.
        binaryImage(r,c) = 1;                       % Plot pixels on "grid canvas".
    end

    [ol,dir] = contour(binaryImage, ...                        % Find the contour of the
        [ rows-floor((points(minXindx,2)-minY)/gridLength) ... % digitized points, also returning
        floor((points(minXindx,1)-minX)/gridLength)+1 ]);      % the normal direction for each pixel.

    for i=1:size(ol,1)                                  % Find a single point in each perimeter
                                                        % pixel that represents the outline.
        indx = cellSorted{ol(i,1),ol(i,2)};             % All points that represent
        cellPoints = [points(indx,1) points(indx,2)];   % the same pixel.
        BOUNDARY(i,:) = farthest(cellPoints,dir(i));    % Find the point that is closest to
                                                        % the "outside" of the perimeter.
    end
else
    BOUNDARY=[];
end
catch
    BOUNDARY=[];
end
```

Bibliography

[1] Asano, T., Akagi, M., Tanaka, K., Tamura, J., and Nakamura, T. In vivo three-dimensional knee kinematics using a biplanar image-matching technique. *Clinical Orthopaedics & Related Research 388* (Jul 2001), 157–66.

[2] Banks, S., and Hodge, W. Accurate measurement of three-dimensional knee replacement kinematics using single-plane fluoroscopy. *IEEE Transactions on Biomedical Engineering 43*, 6 (Jun 1996), 638–49.

[3] Canny, J. A computational approach to edge-detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence 8*, 6 (Nov 1986), 679–698.

[4] DeFrate, L. E., Sun, H., Gill, T. J., Rubash, H. E., and Li, G. A. In vivo tibiofemoral contact analysis using 3d mri-based knee models. *Journal of Biomechanics 37*, 10 (Oct 2004), 1499–1504.

[5] Fletcher, R. *Practical methods of optimization*, 2nd ed. Wiley, Chichester, New York, 1987.

[6] Fukuoka, Y., Hoshino, A., and Ishida, A. Accurate 3d pose estimation method for polyethylene wear assessment in total knee replacement. In *Proceedings of the 19th Annual International Conference of the IEEE Engineering in Medicine and Biology Society. 'Magnificent Milestones and Emerging Opportunities in Medical Engineering', 30 Oct.-2 Nov. 1997* (1997), vol. vol.4, pp. 1849–52.

[7] Garling, E. H., Kaptein, B. L., Geleijns, K., Nelissen, R. G. H. H., and Valstar, E. R. Marker configuration model-based roentgen fluoroscopic analysis. *Journal of Biomechanics 38*, 4 (2005), 893–901.

[8] Gronenschild, E. The accuracy and reproducibility of a global method to correct for geometric image distortion in the x-ray imaging chain. *Medical Physics 24*, 12 (Dec 1997), 1875–1888.

[9] Hanson, G., Suggs, J., Rubash, H., and Li, G. A. A dual orthogonal fluoroscopic system for determining in vivo tka kinematics. In *30th ORS Transactions* (2005), vol. 30.

[10] Hoff, W., Komistek, R., Dennis, D., Walker, S., Northcut, E., and Spargo, K. Pose estimation of artificial knee implants in fluoroscopy images using a template matching technique. In *Proceedings Third IEEE Workshop on Applications of Computer Vision. WACV'96, 2-4 Dec. 1996* (1996), pp. 181–6.

[11] Kanisawa, I., Banks, A. Z., Banks, S. A., Moriya, H., and Tsuchiya, A. Weight-bearing knee kinematics in subjects with two types of anterior cruciate ligament reconstructions. *Knee Surg Sports Traumatol Arthrosc 11* (2003), 16–22.

[12] Kaptein, B. L., Valstar, E. R., Stoel, B. C., Rozing, P. M., and Reiber, J. H. C. A new model-based rsa method validated using cad models and models from reversed engineering. *Journal of Biomechanics 36*, 6 (Jun 2003), 873–882.

[13] Kaptein, B. L., Valstar, E. R., Stoel, B. C., Rozing, P. M., and Reiber, J. H. C. Evaluation of three pose estimation algorithms for model-based roentgen stereophotogrammetric analysis. *Proceedings of the Institution of Mechanical Engineers Part H-Journal of Engineering in Medicine 218*, H4 (Jul 2004), 231–238.

[14] Koelzow, T., and Krueger, L. Matching of a 3d model into a 2d image using a hypothesize and test alignment method. *Proceedings of the SPIE, July 2002 4791* (2002), 222–32.

[15] Komistek, R. D., Dennis, D. A., and Mahfouz, M. In vivo fluoroscopic analysis of the normal human knee. *Clinical Orthopaedics & Related Research 410* (May 2003), 69–81.

[16] Lavallee, S., and Szeliski, R. Recovering the position and orientation of free-form objects from image contours using 3d distance maps. *IEEE Transactions on Pattern Analysis and Machine Intelligence 17*, 4 (April 1995), 378–90.

[17] Lee, E. T. Y. Choosing nodes in parametric curve interpolation. *Computer-Aided Design 21*, 6 (Jul-Aug 1989), 363–370.

[18] Li, G., DeFrate, L. E., Sun, H., and Gill, T. J. In vivo elongation of the anterior cruciate ligament and posterior cruciate ligament during knee flexion. *American Journal of Sports Medicine 32*, 6 (Sep 2004), 1415–20.

[19] Li, G. A., Wuerz, T. H., and DeFrate, L. E. Feasibility of using orthogonal fluoroscopic images to measure in vivo joint kinematics. *Journal of Biomechanical Engineering 126*, 2 (Apr 2004), 314–318.

[20] Mahfouz, M., Hoff, W., Komistek, R., and Dennis, D. A robust method for registration of three-dimensional knee implant models to two-dimensional fluoroscopy images. *IEEE Transactions on Medical Imaging 22*, 12 (Dec 2003), 1561–74.

[21] Mahfouz, M. R., Hoff, W. A., Komistek, R. D., and Dennis, D. A. Effect of segmentation errors on 3d-to-2d registration of implant models in x-ray images. *Journal of Biomechanics 38*, 2 (Feb 2005), 229–39.

[22] Sato, T., Koga, Y., and Omori, G. Three-dimensional lower extremity alignment assessment system: application to evaluation of component position after total knee arthroplasty. *Journal of Arthroplasty 19*, 5 (Aug 2004), 620–8.

[23] SCHUTTE, J., KOH, J., REINBOLT, J., HAFTKA, R., GEORGE, A., AND FREGLY, B. Evaluation of a particle swarm algorithm for biomechanical optimization. *Journal of Biomechanical Engineering 127* (2005), 465–474.

[24] SHORT, A., GILL, H. S., MARKS, B., WAITE, J. C., KELLETT, C. F., PRICE, A. J., O'CONNOR, J. J., AND MURRAY, D. W. A novel method for in vivo knee prosthesis wear measurement. *Journal of Biomechanics 38*, 2 (Feb 2005), 315–22.

[25] STIEHL, J. B., KOMISTEK, R. D., DENNIS, D. A., PAXSON, R. D., AND HOFF, W. A. Fluoroscopic analysis of kinematics after posterior-cruciate-retaining knee arthroplasty. *Journal of Bone and Joint Surgery 77B*, 6 (Nov 1995), 884–889.

[26] SUGGS, J., HANSON, G., DURBHAKULA, S., PAPANNAGARI, R., JOHNSON, T., FREIBERG, A., RUBASH, H., AND LI, G. A. Patient specific 3d analysis of in vivo knee kinematics after cruciate retaining total knee arthroplasty. In *30th ORS Transactions* (2005), vol. 30.

[27] TOMAZEVIC, D., LIKAR, B., SLIVNIK, T., AND PERNU, F. 3-d/2-d registration of ct and mr to x-ray images. *IEEE Transactions on Medical Imaging 22*, 11 (November 2003), 1407–16.

[28] VALSTAR, E. R., DE JONG, F. W., VROOMAN, H. A., ROZING, P. M., AND REIBER, J. H. C. Model-based roentgen stereophotogrammetry of orthopaedic implants. *Journal of Biomechanics 34*, 6 (Jun 2001), 715–722.

[29] WALKER, S. A., KOMISTEK, R., HOFF, W., AND DENNIS, D. 'in vivo' pose estimation of artificial knee implants using computer vision. In *Proceedings of the 1996 33rd Annual Rocky Mountain Bioengineering Symposium & 33rd International ISA Biomedical Sciences Instrumentation Symposium, Apr 12-13 1996* (1996), vol. 32, pp. 143–150.

[30] YAMAZAKI, T., WATANABE, T., NAKAJIMA, Y., SUGAMOTO, K., TOMITA, T., YOSHIKAWA, H., AND TAMURA, S. Improvement of depth position in 2-d/3-d registration of knee implants using single-plane fluoroscopy. *IEEE Transactions on Medical Imaging 23*, 5 (May 2004), 602–612.

[31] YOU, B.-M., SIY, P., ANDERST, W., AND TASHMAN, S. In vivo measurement of 3-d skeletal kinematics from sequences of biplane radiographs: Application to knee kinematics. *IEEE Transactions on Medical Imaging 20*, 6 (June 2001), 514–25.

[32] ZUFFI, S., LEARDINI, A., CATANI, F., FANTOZZI, S., AND CAPPELLO, A. A model-based method for the reconstruction of total knee replacement kinematics. *IEEE Transactions on Medical Imaging 18*, 10 (Oct 1999), 981–91.