A Kinematic Model of the Human Arm Using Triangular Bézier Spline Surfaces

Deepak Tolani, Norman Badler, Jean Gallier

Department of Computer and Information Science
University of Pennsylvania
Philadelphia, PA 19104, USA
`jean@saul.cis.upenn.edu`

March 15, 2000

Abstract. This paper presents a kinematic model of the human arm in which the workspace of the elbow is modeled as a triangular Bézier spline surface. It is also explained how this model is used for solving forward and inverse kinematics in computer animation systems. In order to solve inverse kinematics problems, it is necessary to find a curve on the surface modeling the elbow workspace. This curve is obtained as the intersection of the surface and a sphere. We present an algorithm for computing this curve, using the fact that triangular Bézier patch can be efficiently subdivided, and using an oct-tree data structure that prunes the search space efficiently.

1 Introduction

This paper presents a new kinematic model of the human arm in which the workspace of the elbow is modeled as a triangular Bézier spline surface. It is also explained how this model is used for solving forward and inverse kinematics in computer animation systems. In order to solve inverse kinematics problems, it is necessary to find a curve on the surface modeling the elbow workspace. This curve is obtained as the intersection of the surface and a sphere. We present an algorithm for computing this curve, using the fact that triangular Bézier patch can be efficiently subdivided, and using an oct-tree data structure that prunes the search space efficiently.

1. Most of our algorithms are repeatable, in that the solution does not depend upon the starting posture of the arm. Repeatability is a significant issue in interactive applications where a user spends a great deal of time manipulating the solver into finding a satisfactory final posture. In these cases, it is very important to the user that the solver behave consistently and not be sensitive to minor perturbations of the starting state.

2. We use a parametric surface to accurately model the workspace of the elbow (the tip of the humerus). In a first step, we use motion capture to obtain a set of points modeling the workspace of the tip of the humerus (see Section 2). In a second step, we interpolate this set of points using a triangular Bézier spline surface with C^1-continuity (see Section 3). This approach allows us to construct a model based purely on experimental data, and it guarantees that the end effector behavior will be consistent with empirical observations.

3. Our model automatically enforces complex joint limits and couplings that are difficult to characterize using traditional methods. Additionally, our model permits both forward and inverse kinematics analysis.

4. Obviously, there is a tradeoff between obtaining an accurate kinematic model of the body and computational expense. One of our goals is to find a suitable balance between these two objectives so that visually acceptable results can be obtained in real time.

Advantages of our model are that it encodes both positional and orientation information, transforms a system described by many coupled degrees of freedom into one described by a smaller set of independent variables, and produces an accurate simulation of the reachable workspace. However, the surface-based model is subject to the same computational inefficiencies and problems of other numerical approaches to inverse kinematics. Another limitation, though not necessarily a disadvantage, is that the model is specific for an individual and is not readily adapted to handle figures with different anthropometries.

In order to solve inverse kinematics problems, it is necessary to find a curve on the surface modeling the elbow workspace. This curve is obtained as the intersection of the surface and a sphere. In Section 4, we present an algorithm for computing this curve using the fact that a triangular Bézier patch can be efficiently subdivided and using an oct-tree data structure that prunes the search space efficiently. The resulting algorithm is interesting in itself, and proves to be very efficient in practice.

The paper is organized as follows. A model of the arm based on empirical data is presented in Section 2. In Section 3, we explain how the elbow workspace is modeled using a triangular Bézier spline surface. The use of an interpolating triangular spline surface with C^1-continuity is novel. Methods for solving direct and inverse kinematics problems using this new model are presented in Section 4. In particular, we present an efficient algorithm for computing the interection curve of the modeling surface with a sphere. We conclude this Section by discussing other types of problems amenable to our methods, as well as the limitations of our methods. Section 5 is an Appendix devoted to a review of basics on Bézier curves and (triangular) Bézier surfaces, including subdivision.

2 A Model of the Arm Based on Empirical Data

2.1 Introduction

Many of the joints in the human body act together as functional units rather than independently. For example, the motions of the shoulder, scapula, and clavicle act in a concerted fashion to move the humerus. An even more extreme example is the spine which consists of twenty four joints whose motions are tightly coupled. The motions of these joints are further complicated by the fact that their mobility is restricted not just by mechanical limits on the joints but also by the surrounding muscle mass and soft tissue. It is virtually impossible to accurately model the workspace of the shoulder or spine using a conventional robotics model. If the joints are treated as independent degrees of freedom the model will have too much redundancy. Additionally, imposing simple linear inequality limits on the degrees of freedom does not realistically simulate the complex anatomical coupling between these joints.

We seek a compromise scheme that allows a realistic kinematic model while still permitting sufficient computational simplicity so that forward and inverse kinematics can be computed quickly. Several authors have attempted to model the behavior of complex joints such as the shoulder and spine. Otani [16] devised a formula for distributing elevation, abduction, and twist of the humerus into joint angles for the shoulder and clavicle. Monheit and Badler [14] have developed a model of the human spine that translates gross motions of the torso into individual degrees of freedom at the spine joints. Our work is distinguished from previous work in that our model is based purely on empirical data and does not rely on a detailed understanding of how the biokinematic structures actually work. Moreover, our model will guarantee that the end effector behavior is always consistent with the data, at the possible expense of less than perfect behavior at the joint level.

In our approach, we record position and orientation data of the elbow end effector to construct the boundary of the shoulder and upper arm workspace. Such data can be obtained through a motion capture system and from biomechanical studies. We use a piecewise parametric surface to interpolate the data to construct the reachable workspace of the end effector. We want to emphasize the difference between our scheme and another interpolation scheme that is used in some animation systems. Wiley and Hahn [19] have developed a system that prerecords inverse kinematics solutions by storing a table of predetermined joint angles indexed by the cartesian coordinates of the desired goal. The data is obtained by motion capture studies of a subject and is specific for that individual. To solve an arbitrary inverse kinematics problem the system finds the nearest neighboring grid points to the desired goal and performs interpolation on the corresponding set of joint angles to obtain an approximate solution. Although this method may produce "natural" looking solutions, it has several significant drawbacks. Because the system uses interpolation to obtain an answer the method is not accurate unless a large number of data points are used. If m is the number of joints and n is the number of grid points the scheme requires $O(m * n^3)$ storage if position information is used and $O(m * n^6)$ storage if both position and orientation constraints are required. For example, if the workspace is a 100cm cube and we represent orientation as Euler angles then with 1cm position and 5 degree orientation sampling this approach would require $100^3 * \left(\frac{360}{5}\right)^3$ data points. Not only are the storage requirements expensive, but it is also tedious and impractical to collect such a large data sample. By contrast, our method requires only linear storage in terms of the number of sample points. In our approach, we are sampling achievable end effector positions and orientations rather than desired goal states. Since the data lies on a constraint surface rather than in a volume only 100–200 points are required to produce a reasonable surface model. Wiley's interpolation system only gives one solution to an inverse kinematics problem. Moreover, unless the goal coincides with a grid point, joint angles from neighboring grid points must be interpolated to obtain an approximate answer. By contrast, the approach that we will describe generates an accurate answer and also allows the user to find all possible solutions. Other relevant related work includes Aydin and Nakajima [1] and Multon et al [15].

In our approach, the skeleton is still modeled as a system of rigid links connected by rotating joints. However, the joints are not independent but coupled by the fact that the end effector is constrained to lie on the surface. The forward and inverse kinematics are performed in terms of the surface parameters rather than the joint angles. We will illustrate these ideas in more detail in subsequent sections. In this section we describe our experiments for determining the reachable workspace of the arm.

2.2 Experimental Procedure

We used the Flock of Birds^{TM} (FOB) position and orientation measurement system by Ascension Technologies to obtain experimental data for our model. The FOB system is a six degree-of-freedom measuring device that can track the position and orientation of multiple receivers by a transmitter. A transmitter generates a magnetic field that is measured by each receiver which in turn computes its position and orientation.

	Female	Male
Distance from clavicle to shoulder	10 cm	15 cm
Distance from shoulder to elbow	26 cm	31 cm
Average distance of elbow to clavicle	29 cm	35 cm
Shortest distance of elbow to clavicle	20 cm	21 cm
Furthest distance of elbow to clavicle	35 cm	46 cm
Standard deviation	3.9 cm	6.5 cm

Table 1: Link lengths and distances of elbow to clavicle for both subjects.

Sensors were attached on the right arm at locations roughly corresponding to the joint centers of the shoulder, clavicle, elbow and wrist. Each subject moved his/her arm along longitudinal and latitudinal directions tracing the workspace of the elbow relative to the clavicle. To measure twist each subject kept his/her elbow fixed and rotated the lower arm about the axis of the upper arm with the elbow joint flexed at approximately ninety degrees. The range of twist was recorded for a variety of arm elevations. Two subjects, one female and one male, were used.

2.3 Experimental Results

In order to describe our results, we will use the following terminology. The rest position of the arm is defined as the arm fully extended with the fingers pointing downwards and the palms facing the body. The frontal plane is the plane that divides the body into posterior and anterior sections. The sagittal plane divides the body bilaterally into two symmetrical halves. The transverse plane passes through the clavicle and is perpendicular to the other two planes.

Our experiments indicate that the workspace traced by the tip of the humerus is crudely a hemisphere on one side of the sagittal plane with the center located at the clavicle. However, the radius of the surface varies greatly throughout the range of the workspace. For both subjects, the distance from the humerus to the clavicle reaches its minimum value when the elbow is close to the sagittal plane, is in front of the frontal plane and slightly above the transverse plane. Likewise, in both subjects the distance is maximized when the elbow is close to the sagittal plane, is behind the frontal plane and slightly below the transverse plane.

Figures 1, 2, 3, 4, show the density distribution of the distance of the elbow to the clavicle. The male subject results appear on the left column and the female subject results are shown on the right. Fig. 1 shows all of the data collected. Fig. 2 shows the third of the data that is closest to the clavicle. Fig. 3 and 4 show the third of the data in the middle and furthest most ranges from the clavicle, respectively. It is apparent that the distance of the tip of the humerus from the clavicle is more dependent on the amount of frontal plane abduction than it is on elevation. It is also clear that as the arm abducts behind the frontal plane the distance from the humerus to the clavicle increases.

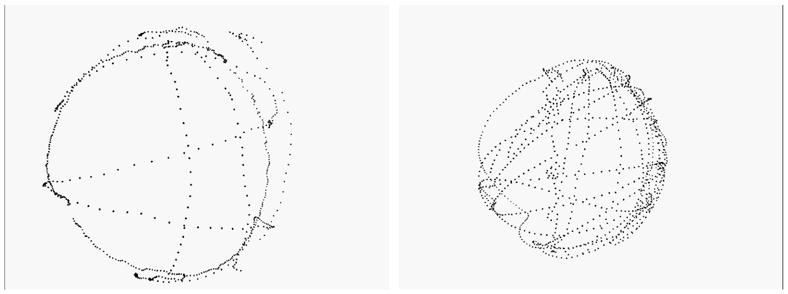

Figure 1: Distance of the elbow from the humerus, first data range.

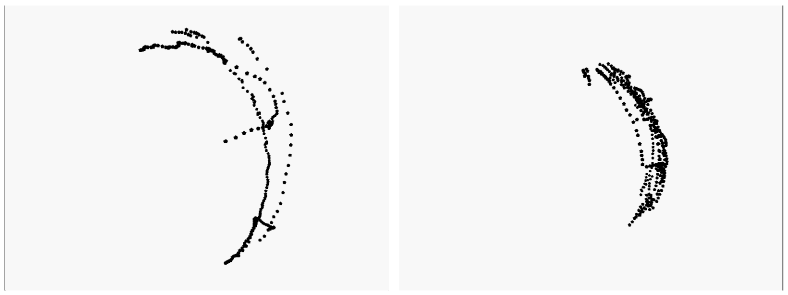

Figure 2: Distance of the elbow from the humerus, second data range.

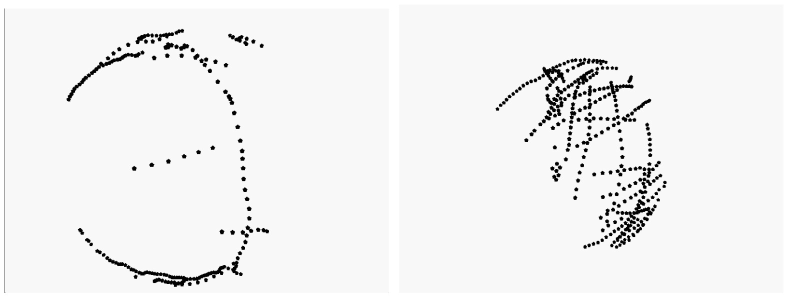

Figure 3: Distance of the elbow from the humerus, third data range.

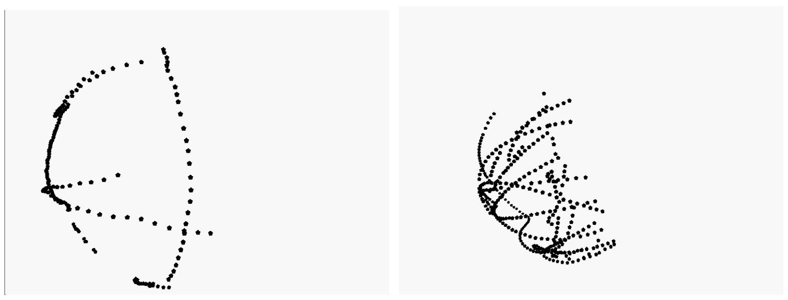

Figure 4: Distance of the elbow from the humerus, fourth data range.

Additionally, in both subjects, the elbow traces a sharp corner in the region behind the frontal plane and beneath the transverse plane. Despite the difference in link lengths between the two subjects, our results suggest that the general shape and properties of their workspaces are similar, but clearly a larger sample size is necessary before any general conclusions about the nature of the arm workspace can be reached.

Twist was difficult to measure precisely because it is not possible for the subject to keep his/her elbow perfectly still while twisting. However, the experiments also revealed that both subjects have substantially more available twist near the rest position than when the arm is fully elevated. Figures 5 and 6 shows the linear and quadratic least squares fit showing the relationship between the range of available twist and the elevation of the elbow.

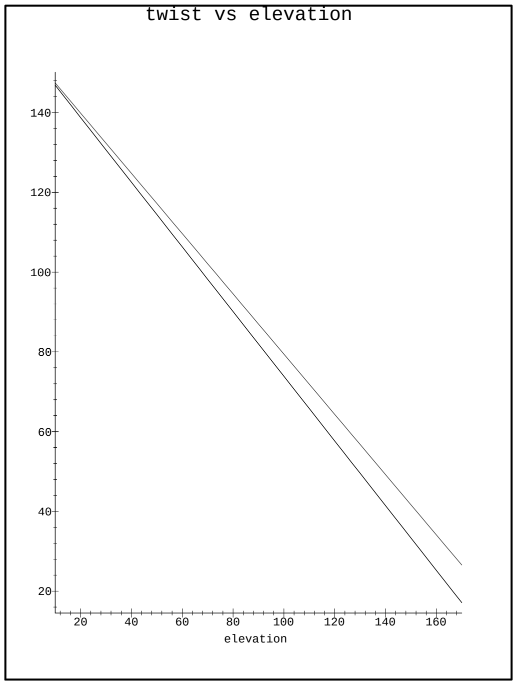

Figure 5: Linear Least Squares fit of range of twist versus elbow elevation.

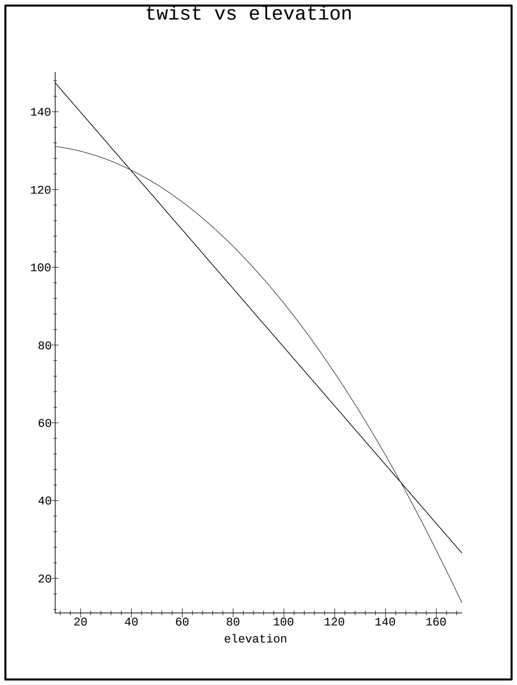

Figure 6: Quadratic Least Squares fit of range of twist versus elbow elevation.

2.4 Caveats

It should be noted that the experiments cannot be performed with great precision. A small amount of noise in any motion tracking system is inevitable. Additionally, it is impossible to place the sensors precisely on the joint centers and the sensors will tend to shift as the skin and tissue stretch. Thus, the distances from the shoulder to the clavicle sensors and from the elbow to the shoulder sensors are not constant. This may cause problems when we attempt to fit these results to an underlying model that uses a rigid skeleton that assumes fixed link lengths. Additionally, it is difficult to obtain a uniform distribution of data, and some areas of the workspace are clearly undersampled relative to other regions. Oversampling is not a significant problem since excess data can be decimated. However, insufficient data can produce poor interpolation in the undersampled regions as we observe in section 3.

3 Workspace Surface Interpolation

The shoulder girdle consists of the clavicle and scapula which move in a closed-loop about the sternum and rib cage. Additionally, the humerus can also move as an open chain about the scapula and clavicle frame. Altogether, the shoulder complex consists of twelve interdependent degrees-of-freedom, distributed across one open and one closed kinematic loop.

It is difficult to model the workspace of the shoulder complex using conventional robotics models. For example, if a spherical joint is used with linear joint limits, the traced workspace is a spherical rectangle with uniform radius and uniform twist. Alternatively, if we model the shoulder complex as a more complex series of joints coupled by nonlinear constraints, it is very difficult to devise a set of constraints that compel the end effector to trace the shape of the workspace and satisfy the valid twist ranges.

3.1 Triangular Bézier Patches for Surface Interpolation

Although it is difficult to construct a mathematical model of the shoulder workspace that is empirically correct, it is relatively easy to fit an interpolatory surface through the workspace data and then to solve for an appropriate set of joint angles using a relatively simple kinematic model for the clavicle and shoulder. Imagine a surface that encodes the locations of four points, $\mathbf{s}(u,v)$, $\mathbf{e}(u,v)$, $\mathbf{w}_0(u,v)$, and $\mathbf{w}_1(u,v)$, all measured relative to the clavicle. The variables $\mathbf{s}$ and $\mathbf{e}$ denote the shoulder and elbow sensors; $\mathbf{w}_0$ and $\mathbf{w}_1$ are the positions of the wrist with the elbow flexed at approximately ninety degrees at the extreme twist ranges of the upper arm. Using this information, a forward and inverse kinematics model of the shoulder complex can be devised based on empirical data rather than a hypothetical model of the joint behavior.

3.2 Why use Triangular Bézier Patches ?

Many surface fitting techniques are possible, and perhaps, the choice of a surface interpolation technique is ultimately not important, provided a good fit can be obtained. However, we chose to use piecewise triangular cubic C^1-Bézier polynomial patches for the following reasons:

1. Polynomial surface interpolation leads to a well behaved linear system of equations. By contrast, rational surface interpolation, while more accurate and possessing extra degrees of freedom, results in nonlinear systems of equations that may in practice be much more difficult to solve. We use cubic surfaces because cubics are the lowest possible degree that permit us to guarantee C^1-continuity while giving us enough remaining degrees of freedom to do useful interpolation.

2. Triangles are a more natural domain for interpolating a spherical data point set than rectangles. If rectangular control patches are used, then some control points must be duplicated to collapse a degenerate edge to a single vertex, or it is necessary to use multiple rectangular patches to cover regions that can be represented by a single triangular patch. Additionally, triangular patches have lower total degree than rectangular patches. For example, a tensor product surface that is cubic in both of its arguments has total degree six compared to three for its triangular counterpart. We would like to keep the total degree of the surface as low as possible to minimize computational expense and the number of control points. Moreover, the intersection algorithm we

use in section 4 requires subdivision of the surface into polygonal segments that are intersected with a sphere. Subdividing a tensor product surface produces quadrilaterals which are generally non-planar thereby complicating the intersection routine.

3. The surface should be visually smooth and hence the tangent planes should be continuous across patch boundaries. Ideally, only geometric or G^1-continuity is required. Ironically, even though G^1-continuity is less restrictive than C^1-continuity, enforcing G^1-continuity results in a nonlinear system of equations that are more difficult to solve than the C^1-constraints. We therefore use the C^1-conditions even though they impose more conditions on the surface than are necessary for our application.

4. Subdivision surfaces have undergone a recent resurgence in popularity in computer graphics. Unlike Bézier and other parametric surfaces, a subdivision surface cannot be explicitly evaluated from u, v-coordinates, but instead, are generated recursively by subdividing a polyhedral mesh. A subdivision surface is produced in two steps. In the splitting phase, a set of additional vertices are generated by taking the midpoints of all edges in the current mesh. In the averaging phase, the new values of all the vertices are computed as affine combinations of their neighbors. This process is performed recursively, and in the limit, produces a C^1-surface. Subdivision methods can be defined for either triangular or rectangular patches; additionally, either interpolatory and approximating schemes can be used [17]. There are compelling reasons to use subdivision surfaces. One of their principal virtues is they can be used on meshes of arbitrary topology. Additionally, if an interpolatory scheme is used, the subdivision surface can interpolate all the data points. However, subdivision surfaces have some properties that are problematic for this particular application. Because subdivision algorithms work by refining a polyhedral mesh, the collected data points must first be triangulated. In general, the problem of triangulating a set of arbitrary three dimensional points is not trivial and is an active area of research [2]. Additionally, with a subdivision approach, one has the option of using either an approximating or an interpolating method. The disadvantage of using an approximating method is that an approximating subdivision surface, like a uniform B-spline surface, does not interpolate the exterior vertices. Thus the resulting surface is smaller than the surface spanned by the original set of data points and a substantial fraction of the workspace is lost. On the other hand, if the data is noisy, then an interpolatory scheme will incorporate the noise rather than smoothing out perturbations in the surface caused by errors. Finally, in a subdivision surface, there is no convenient way to associate points with explicit coordinates. As we shall see, this does not present a problem for using the surface in inverse kinematics, but it does pose a difficulty in forward kinematics where it is necessary to parameterize the joint angles in terms of explicit coordinates.

3.3 Solving the Interpolation Problem

The next step is to determine the number of patches and their layout. After experimenting with several schemes, we chose a hexagonal network consisting of six patches as shown in figure 7.

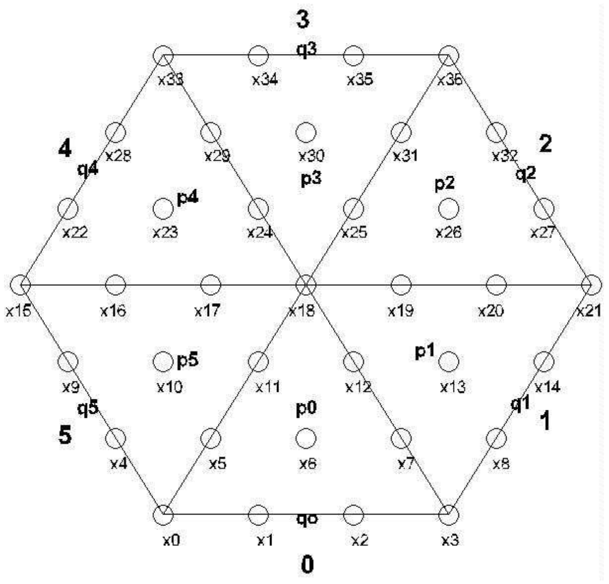

Figure 7: A hexagonal triangular patch configuration.

The hexagonal layout was chosen since it establishes an approximate match between the parameter shape and the object shape. In principle, we could construct more elaborate hexagons with a larger number of patches to obtain a closer correspondence to the surface if greater accuracy is desired.

3.4 Formulating the Constraints

Referring to section 5.8, the C^1-conditions impose the following constraints on the control points:

$$\begin{aligned}
x_{22} + x_9 &= x_{15} + x_{16} \\
x_{23} + x_{10} &= x_{16} + x_{17} \\
x_{24} + x_{11} &= x_{17} + x_{18} \\
x_{25} + x_{12} &= x_{18} + x_{19} \\
x_{26} + x_{13} &= x_{19} + x_{20} \\
x_{27} + x_{14} &= x_{20} + x_{21} \\
x_1 + x_4 &= x_0 + x_5 \\
x_6 + x_{10} &= x_5 + x_{11}
\end{aligned}$$

$$\begin{aligned}
x_{12} + x_{17} &= x_{11} + x_{18} \\
x_{19} + x_{24} &= x_{18} + x_{25} \\
x_{26} + x_{30} &= x_{25} + x_{31} \\
x_{32} + x_{35} &= x_{31} + x_{36} \\
x_{28} + x_{34} &= x_{33} + x_{29} \\
x_{23} + x_{30} &= x_{29} + x_{24} \\
x_{17} + x_{25} &= x_{24} + x_{18} \\
x_{11} + x_{19} &= x_{18} + x_{12} \\
x_{6} + x_{13} &= x_{12} + x_{7} \\
x_{2} + x_{8} &= x_{7} + x_{3}.
\end{aligned}$$

Since Bézier patches automatically interpolate their corner control points, we choose to specify the seven corner points $\mathbf{x}_0, \mathbf{x}_3, \mathbf{x}_{15}, \mathbf{x}_{18}, \mathbf{x}_{21}, \mathbf{x}_{33}$, and $\mathbf{x}_{36}$. Additionally, it also seems to make sense to specify interior points corresponding to $[u, v, w]^T$ coordinates of $\left[\frac{1}{3}, \frac{1}{3}, \frac{1}{3}\right]$ for each patch. Enumerating the patches starting from the bottom triangle and winding counterclockwise, we designate these points as $\mathbf{p}_0, \mathbf{p}_1, ..., \mathbf{p}_5$. The requirement that the ith patch interpolates $\mathbf{p}_i$ at $\left[\frac{1}{3}, \frac{1}{3}, \frac{1}{3}\right]$ yields the additional six equations:

$$\begin{aligned}
\mathbf{p}_0 &= \left(\frac{1}{27}\right)(x_0 + x_3 + x_{18} + 3(x_1 + x_2 + x_7 + x_{12} + x_{11} + x_5) + 6x_6) \\
\mathbf{p}_1 &= \left(\frac{1}{27}\right)(x_3 + x_{21} + x_{18} + 3(x_7 + x_8 + x_{14} + x_{20} + x_{19} + x_{12}) + 6x_{13}) \\
\mathbf{p}_2 &= \left(\frac{1}{27}\right)(x_{18} + x_{21} + x_{36} + 3(x_{19} + x_{20} + x_{27} + x_{32} + x_{31} + x_{25}) + 6x_{26}) \\
\mathbf{p}_3 &= \left(\frac{1}{27}\right)(x_{18} + x_{36} + x_{33} + 3(x_{24} + x_{25} + x_{31} + x_{35} + x_{34} + x_{29}) + 6x_{30}) \\
\mathbf{p}_4 &= \left(\frac{1}{27}\right)(x_{18} + x_{33} + x_{15} + 3(x_{16} + x_{17} + x_{24} + x_{29} + x_{28} + x_{22}) + 6x_{23}) \\
\mathbf{p}_5 &= \left(\frac{1}{27}\right)(x_{18} + x_{15} + x_0 + 3(x_4 + x_5 + x_{11} + x_{17} + x_{16} + x_9) + 6x_{10}).
\end{aligned}$$

Finally, we presume that we have six additional points $\mathbf{q}_0, ..., \mathbf{q}_5$ that we would like the surface to interpolate along the middle of the outer edges of the patch boundaries corresponding to barycentric coordinates of $\left(u = v = \frac{1}{2}, w = 0\right)$, $\left(u = w = \frac{1}{2}, v = 0\right)$, or $\left(v = w = \frac{1}{2}, u = 0\right)$. The equations relating these points to the control points are given by:

$$
\begin{aligned}
\mathbf{q}_0 &= \frac{1}{8}(x_0 + 3x_1 + 3x_2 + x_3) \\
\mathbf{q}_1 &= \frac{1}{8}(x_3 + 3x_8 + 3x_{14} + x_{21}) \\
\mathbf{q}_2 &= \frac{1}{8}(x_{21} + 3x_{27} + 3x_{32} + x_{36}) \\
\mathbf{q}_3 &= \frac{1}{8}(x_{36} + 3x_{35} + 3x_{34} + x_{33}) \\
\mathbf{q}_4 &= \frac{1}{8}(x_{33} + 3x_{28} + 3x_{22} + x_{15}) \\
\mathbf{q}_5 &= \frac{1}{8}(x_{15} + 3x_9 + 3x_4 + x_0).
\end{aligned}
$$

We cannot assume that all the constraint equations are independent and consistent. To make sure that the problem is well posed, we first write the thirty equations into matrix form **Ax=b**, where **x** is a vector of the unknown control points $[x_1, x_2, x_4, x_5, ..., x_{35}]$, **A** is a coefficient matrix of constants, and **b** is a vector consisting of constants, the corner control points $[x_0, x_3, x_{15}, x_{18,}x_{21}, x_{33,}x_{36}]$, and the specified interpolation points $\mathbf{q}_i$ and $\mathbf{p}_i$. Forming the augmented matrix $[\mathbf{A}|\mathbf{b}]$ and performing Gaussian elimination yields two zero rows and one equation of the form

$$\frac{15}{32}(x_0 + x_3 + x_{15} + x_{18} + x_{21} + x_{33} + x_{36}) + \frac{27}{32}\sum_{i=0}^{7} p_i - \sum_{i=0}^{7} q_i = 0.$$

This indicates that two of the equations are redundant, and also that the system is inconsistent since the equation violates our assumption that we are free to choose the corner and central control points and $\mathbf{p}_i$ and $\mathbf{q}_i$. The redundancy arises from the fact that only four of the six C^1-continuity constraint equations around the point x_{18} are independent. For a proof of this see Gallier [8].

In order to generate a consistent system of equations, we must relax the restriction that the surface interpolates the $\mathbf{q}_i$ precisely. If the corner control points are specified, it is not possible to interpolate all the **q**'s and **p**'s without breaking the C^1-continuity conditions. Although is possible to interpolate a subset of the **q**'s and **p**'s, we choose to eliminate the constraints on $\mathbf{q}_i$ altogether. Eliminating the appropriate equations from our original system and performing Gaussian elimination again gives a new constraint matrix. Let $\hat{\mathbf{A}}$ and $\hat{\mathbf{b}}$ denote the matrix and right hand side of the new augmented matrix after Gaussian elimination and after the zero rows have been removed. The system is consistent and $\hat{\mathbf{A}}$ is of rank twenty two, indicating that we have a total of 37 - 7 - 22 = 8 extra degrees of freedom. Since it is not possible to distribute these extra degrees of freedom symmetrically over six patches by defining constraints on the control points, we resolve the redundancy by specifying a quadratic objective function that should be minimized subject to the C^1-constraints and to the interpolation constraints. Let $f(\mathbf{x}) = \mathbf{x}^T\mathbf{Q}\mathbf{x}$ denote such a function where $\mathbf{Q}$ is a positive definite symmetric matrix. Forming the Lagrangian

$$L = \mathbf{x}^T\mathbf{Q}\mathbf{x} + \lambda^T(\hat{\mathbf{A}}\mathbf{x} - \hat{\mathbf{b}})$$

and setting the partial derivatives to zero gives

$$\begin{aligned} Q\mathbf{x} + \hat{\mathbf{A}}^T\lambda &= 0 \\ \hat{\mathbf{A}}\mathbf{x} &= \hat{\mathbf{b}} \end{aligned}$$

or

$$(1) \quad \begin{bmatrix} \mathbf{Q} & \hat{\mathbf{A}}^T \\ \hat{\mathbf{A}} & \mathbf{0} \end{bmatrix} \begin{bmatrix} \mathbf{x} \\ \lambda \end{bmatrix} = \begin{bmatrix} \mathbf{0} \\ \hat{\mathbf{b}} \end{bmatrix}$$

which can be solved for $\mathbf{x}$ and λ. For the objective function we use

$$(2) \quad f(\mathbf{x}) = \sum_{i=0}^{5} h_i^2 + \alpha \sum_{i\in\{5,6,7,10,11,12,13,16,17,19,20,22,23,24,25,26,28,29,30,31\}} x_i^2$$

where α is a positive scalar value and

$$\begin{aligned} h_0 &= \mathbf{q}_0 - \frac{1}{8}(x_0 + 3x_1 + 3x_2 + x_3) \\ h_1 &= \mathbf{q}_1 - \frac{1}{8}(x_3 + 3x_8 + 3x_{14} + x_{21}) \\ h_2 &= \mathbf{q}_2 - \frac{1}{8}(x_{21} + 3x_{27} + 3x_{32} + x_{36}) \\ h_3 &= \mathbf{q}_3 - \frac{1}{8}(x_{36} + 3x_{35} + 3x_{34} + x_{33}) \\ h_4 &= \mathbf{q}_4 - \frac{1}{8}(x_{33} + 3x_{28} + 3x_{22} + x_{15}) \\ h_5 &= \mathbf{q}_5 - \frac{1}{8}(x_{15} + 3x_9 + 3x_4 + x_0). \end{aligned}$$

The first summation tends to keep the surface as close as possible to the desired interpolation points $\mathbf{q}_i$, and the terms in the second summation are needed to ensure that $\mathbf{Q}$ is full rank. Figures 8, 9, and 10, show the results obtained using this interpolation scheme on two sets of data.

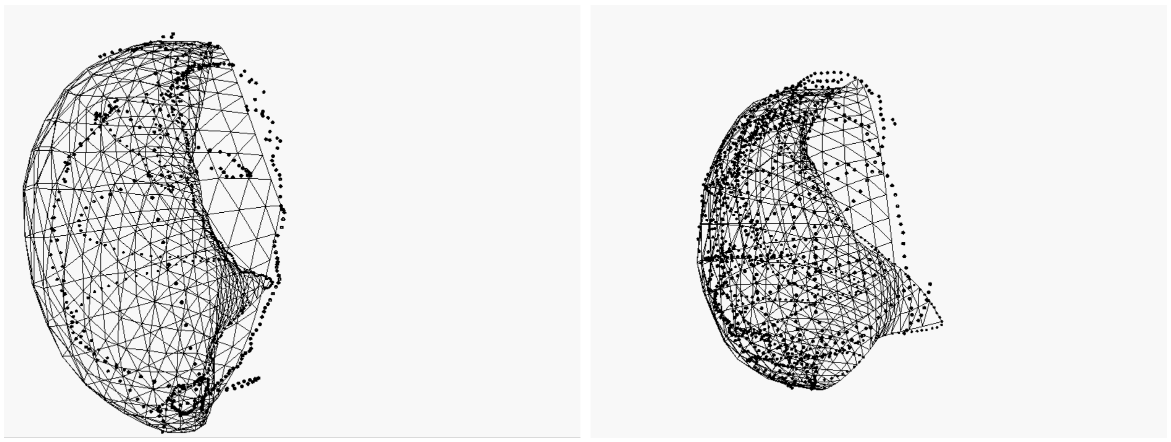

Figure 8: Front of a Bézier surface fit for the male (left) and female (right) subjects.

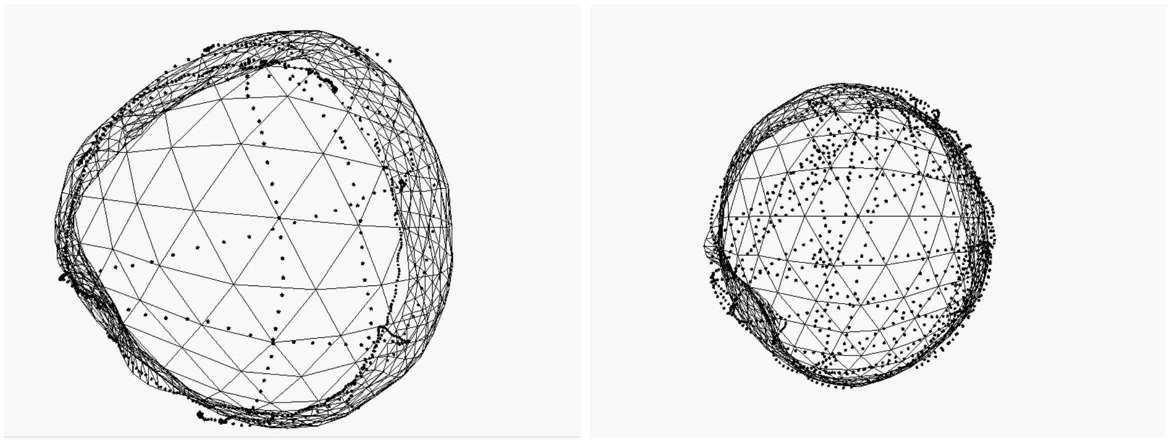

Figure 9: Side of a Bézier surface fit.

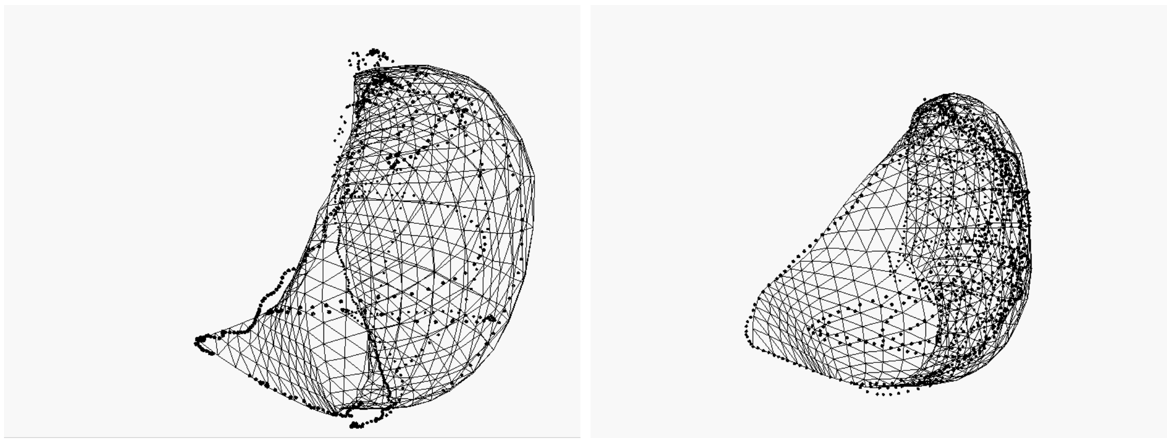

Figure 10: Rear views of a Bézier surface fit.

There is one caveat that should be mentioned. Strictly speaking, we have only guaranteed C^1-continuity on the interior of the surface. It is possible for C^1-continuity to be violated at a corner vertex connecting two exterior edges. In order to guarantee continuity at these corners, it is necessary for the Bézier curves defining the edges to be piecewise continuous. This in turn requires that the two neighboring control points and the common vertex of the Bézier curves are all collinear, and introduces six extra constraint equations into our system. Unfortunately, we have found that adding these additional equations interferes with the other interpolation objectives. Relaxing these conditions seems like a justifiable sacrifice since C^1-continuity is only violated at six extreme points on the surface.

3.5 Adding an Extra Patch

After preliminary attempts at fitting the data, we found that for both subjects, better interpolation could be achieved by adding one extra patch attached to P_5, as shown in figure 11.

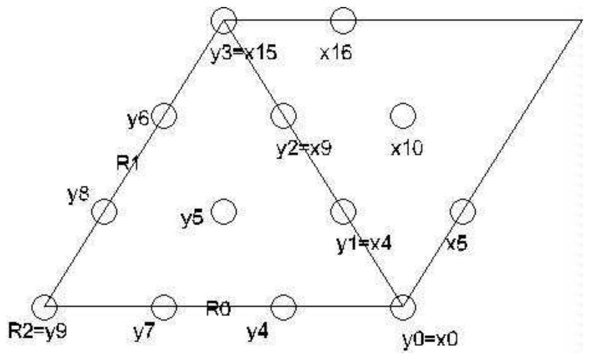

Figure 11: Adding an extra patch.

This patch is used to capture a sharp corner of the workspace located behind the frontal plane and beneath the transverse plane. Denote the control points of this patch by $y_0, ..., y_9$. The C^0 and C^1-continuity constraints completely determine $y_0, ..., y_6$:

$$\begin{aligned} y_0 &= x_0 \\ y_1 &= x_4 \\ y_2 &= x_9 \\ y_3 &= x_{15} \\ y_6 &= x_{15} + x_9 - x_{16} \\ y_5 &= x_4 + x_9 - x_{10} \\ y_4 &= x_0 + x_4 - x_5. \end{aligned}$$

This leaves three extra degrees of freedom in $y_{7,}y_{8,}$ and y_9. Let R_0, R_1, R_2 denote three additional points to interpolate along the two edges and at the corner respectively. Since the patch interpolates its corners, we can choose $y_9 = R_2$. Along each edge, the surface becomes a cubic Bézier curve of the form

$$(1-t)^3 y_9 + 3(1-t)^2 t y_7 + 3(1-t)t^2 y_4 + t^3 y_0$$

or

$$(1-t)^3 y_9 + 3(1-t)^2 t y_8 + 3(1-t)t^2 y_6 + t^3 y_3.$$

If we arbitrarily set $t = \frac{1}{3}$, we can solve for y_7 and y_8

$$\begin{aligned} y_7 &= \frac{9}{4}\left(R_0 - \frac{8}{27}y_9 - \frac{2}{9}y_4 - \frac{1}{27}y_0\right) \\ y_8 &= \frac{9}{4}\left(R_1 - \frac{8}{27}y_9 - \frac{2}{9}y_6 - \frac{1}{27}y_3\right). \end{aligned}$$

3.6 Results

The fitted surfaces for both subjects are shown in Figs. 8, 9, and 10. The three viewpoints are in front of the frontal plane, to the right of the sagittal plane, and behind the frontal plane. The surface fitting procedure is imperfect, some of the points on the boundary of the workspace are not included in the surface, and the surface tends to have sharp corners at some of the control vertices. Error was measured by calculating the minimum distance of each data point from the surface. For the male subject, the average error of a point was 1.64 cm and the maximum error was 5.32cm. For the female subject, the average error was 1.34 cm and the maximum error was 4.42cm. However, the severity of the error is exaggerated by a handful of data points around the exterior of the surface. It is unclear if these points are significant or merely noise. In addition to errors in interpolating individual points, the global shape of the surface is slightly more concave than the actual workspace. This is most noticeable at the sharp corner behind the frontal plane where the curvature of the workspace changes rapidly. We believe that most of these difficulties can be overcome easily by adding more degrees of freedom to the surface. For example, we could replace one or more of the triangular patches with four smaller patches of equal degree. This would introduce additional interior control points that can be used to interpolate more data points. Another alternative is to use higher order patches for the entire surface. Using a hexagonal pattern of quartic patches gives us sixty three control vertices and only twenty two independent C^1-constraints, giving us thirty four extra degrees of freedom for interpolation even after the seven corner control points are chosen. A quintic surface gives ninety one control vertices and only twenty eight independent C^1-constraints for a total of 91-28-7=56 extra degrees of freedom.

Another problem is that quality of the fit is highly dependent upon the choice of interpolation points and some regions are undersampled, making it difficult to select appropriate interpolation values for portions of the workspace. Here, the solution is clearly to obtain more data.

Our objective is not to obtain the best possible surface, but to demonstrate that an adequate fit can be obtained and improved if necessary. As noted in the previous section, the data itself is not completely reliable and it is difficult to determine whether the error introduced by the surface approximation is actually much worse than errors caused by imprecise measurements. At some point, it is possible that producing a better "fitting" surface actually incorporates error rather than eliminating it.

3.7 A Comparison with a Biomechanical Model

Much of the existing research on the arm focuses on individual joints rather than the behavior of the entire shoulder complex. Additionally, most of the work is descriptive in nature and is not useful for building a predictive or mathematical model of the arm. Using sonic emitters, Engin and Chen [3] [4] collected statistical data for ten male subjects of ages 18 to 32 measuring the behavior of the composite shoulder complex sinus. Based on the data collected from this study, Engin and Tumer [5] [6] developed a kinematic model consisting of three two degree-of-freedom univeral joints located at the sternoclavicular, claviscapular, and the glenohumeral joints and two one degree-of-freedom sleeve (twisting) joints. Since their model is redundant, an optimization procedure is used to determine suitable joint angles.

Inman et al [12] studied the concerted movement of the humerus, scapula, and clavicle. They observed that beyond a setting phase of approximately thirty degrees of abduction or sixty degrees of flexion, the humerus, scapula, and clavicle move in concert. The ratio of glenohumeral to scapular rotation is approximately two to one, although the relationship is not linear. The glenohumeral rotation predominates primarily at the beginning and at the end of arm elevation. Otani [16] developed a kinematic model of the arm based on the Inman study.

In this section, we compare our empirical results with those predicted by Otani's model [16]. Otani viewed the motion of the arm as three abstract degrees of freedom: elevation, abduction, and twist, which he termed the group joints. In his model, the group joints are physically realized by five internal joints in the clavicle and the shoulder. Based on clinical data, he obtained the following formula for distributing elevation and abduction to the shoulder and clavicle:

$$\begin{aligned}
\varphi_c &= \cos(\theta)\beta_1 + (1-\cos(\theta))\beta_2 - 90 \\
\theta_c &= .2\theta \\
\varphi_s &= \varphi - \varphi_c \\
\theta_s &= \theta - \theta_c \\
\beta_1 &= \begin{cases} .2514\varphi + 91.076 & \text{for } 0 \le \varphi \le 131.4 \\ \text{-}.035\varphi + 128.7 & \text{for } \varphi > 131.4 \end{cases} \\
\beta_2 &= \begin{cases} .21066\varphi + 92.348 & \text{for } 0 \le \varphi \le 130 \\ 120 & \text{for } \varphi > 130 \end{cases}
\end{aligned}$$

where φ and θ are the total elevation and abduction of the shoulder-clavicle complex and the subscripts c and s denote the contributions by the clavicle and the shoulder respectively. The model assumes a constant range of twist of approximately two hundred degrees and a default twist posture for each (φ, θ).

Figures 12, 13, 14, show how the empirical data matches the surface predicted by Otani's shoulder model using the lengths of the shoulder and clavicle links of the female test subject.

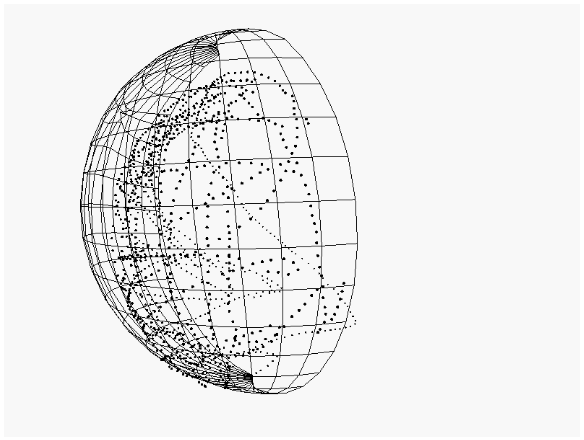

Figure 12: Workspace of the female subject compared to workspace predicted by Otani's model, front.

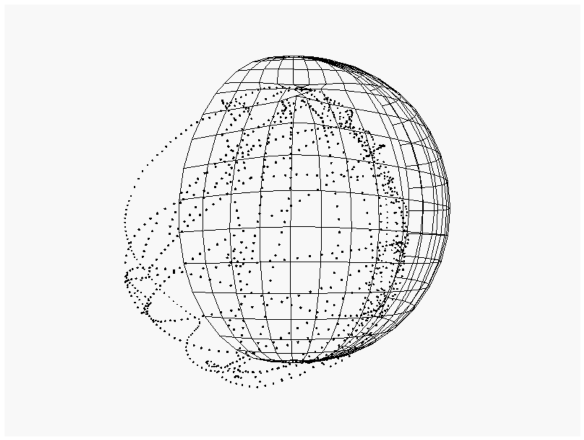

Figure 13: Workspace of the female subject compared to workspace predicted by Otani's model, side.

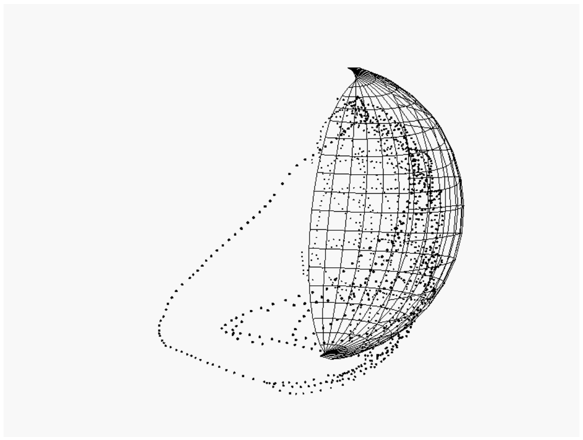

Figure 14: Workspace of the female subject compared to workspace predicted by Otani's model, rear.

There is clearly a great deal of discrepancy between Otani's model and the physical data. In particular, the model seems to exaggerate the contribution of the clavicle, predicting a greater displacement of the elbow than actually occurs; this is particularly noticeable when the arm is fully elevated.

Additionally, Otani's model fails to cover a substantial fraction of the workspace located behind the frontal plane. Finally, Otani's model predicts a uniform range of twist regardless of the arm elevation, and practical experience indicates that this is clearly not the case. We reiterate our belief that it is difficult to find a mathematical model of the arm that is kinematically correct and feel that surface interpolation of the data is a more promising technique for building an accurate model.

4 Forward and Inverse Kinematics for Bézier Patch Models

Most animation systems still model the skeleton of an articulated figure as a chain of rigid links connected by rotating joints. In order for our model to be useful to such a system, we must be able to relate points on the surface to joint angles of an underlying skeleton. We use a simple linkage to represent the shoulder complex in which the clavicle and the shoulder are treated as two and three degree of freedom joints connected in a serial chain. However, these joints are not independent but constrained by restrictions on the position and orientation of the humerus as defined by a Bézier patch surface. At each point on the surface, the humerus can twist about its longitudinal axis. In the previous section we described how to construct a surface

$$\mathbf{B}(u,v,w) = [\mathbf{s}(u,v,w), \mathbf{e}(u,v,w), \mathbf{w}_0(u,v,w), \mathbf{w}_1(u,v,w)]^T,$$

where $\mathbf{s}$ denotes the position of the shoulder, $\mathbf{e}$ the position of the elbow, and $\mathbf{w}_0$ and $\mathbf{w}_1$ the positions of the wrist at configurations of minimum and maximum twist with the elbow bent at approximately ninety degrees. We now describe a forward and inverse kinematics scheme based on the Bézier patch surface representation of the arm's workspace.

4.1 Forward Kinematics

Ideally, the user would like to think of the forward kinematics of the arm in terms of abstract degrees of freedom representing the elevation, abduction, and twist of the arm. The internal representation of a point in the workspace as (u, v, w)-coordinates on a Bézier patch is unintuitive for a user and should be concealed. The elevation and abduction parameters can be interpreted as spherical coordinates for a point on a unit hemisphere. These coordinates can in turn be projected onto a point on the unit disc, and it is easy to devise a mapping from a point on the disc to barycentric coordinates of a point on a corresponding triangle in the parameter space of the surface. This establishes a simple mapping from the user's abstract degrees of freedom to a patch and its corresponding (u, v, w) values. We represent twist by a fourth parameter α, where $0 < \alpha < 1$, and where $\alpha = 0$ and $\alpha = 1$ represent configurations of the arm corresponding to $\mathbf{w}_0$ and $\mathbf{w}_1$.

Of course, the Bézier surface point $\mathbf{s}(u, v, w)$ does not necessarily satisfy the constraint that the distance of the shoulder to the clavicle is constant. We therefore take the actual position of the shoulder $\mathbf{s}^*$ relative to the clavicle as

$$(3) \quad \mathbf{s}^*(u,v,w) = L\frac{\mathbf{s}(u,v,w)}{\|\mathbf{s}(u,v,w)\|},$$

where L is the distance from the clavicle to the shoulder sensor.

For each (u, v, w, α), we can find a transformation matrix $\mathbf{E}$ that relates the position and orientation of the end effector to the clavicle frame

$$(4) \quad \mathbf{E} = \begin{bmatrix} \hat{\mathbf{x}}(u,v,w) & \hat{\mathbf{y}}(u,v,w) & \hat{\mathbf{z}}(u,v,w) & \mathbf{s}^*(u,v,w) \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

As a convention, we assume take the $\hat{\mathbf{z}}$ axis as the twist axis and set it equal to the unit vector along the shaft of the upper arm

$$(5) \quad \hat{\mathbf{z}}(u,v,w) = \frac{\overrightarrow{\mathbf{s}^*\mathbf{e}}(u,v,w)}{\left\|\overrightarrow{\mathbf{s}^*\mathbf{e}}(u,v,w)\right\|}.$$

To define the range of twist, it is also necessary to find an expression for the $\hat{\mathbf{x}}$ axis as a function of α. We define $\hat{\mathbf{x}}_0$ and $\hat{\mathbf{x}}_1$ as the unitized projections of $\overrightarrow{\mathbf{ew}_0}$ and $\overrightarrow{\mathbf{ew}_1}$ onto the plane perpendicular to $\hat{\mathbf{z}}(u, v, w)$

$$\begin{aligned} \hat{\mathbf{x}}_i &= \frac{\overrightarrow{\mathbf{ew}_i} - (\overrightarrow{\mathbf{ew}_i} \cdot \hat{\mathbf{z}})\hat{\mathbf{z}}}{\|\overrightarrow{\mathbf{ew}_i} - (\overrightarrow{\mathbf{ew}_i} \cdot \hat{\mathbf{z}})\hat{\mathbf{z}}\|} \\ i &= 0, 1 \end{aligned}$$

and set $\hat{\mathbf{x}}(\alpha) = \cos(\alpha\theta)\hat{\mathbf{x}}_0 + \sin(\alpha\theta)\,(\hat{\mathbf{z}} \times \hat{\mathbf{x}}_0)$, where θ is the angle between $\hat{\mathbf{x}}_0$ and $\hat{\mathbf{x}}_1$ in the direction of $\hat{\mathbf{z}}$.

In principle, only the surface model is needed to describe the forward kinematics of the shoulder-clavicle complex. If we are interested in only the position of the humerus with respect to the clavicle, then no other information is required. However, for most applications, the skeleton is associated with a set of polygonal segments that define the surface of the arm, and in order to draw these segments, it is necessary to determine the rigid body transformations associated with each joint. In order to determine a set of appropriate transformations, we also utilize an underlying rigid body model for the clavicle-shoulder complex as shown in figure 15.

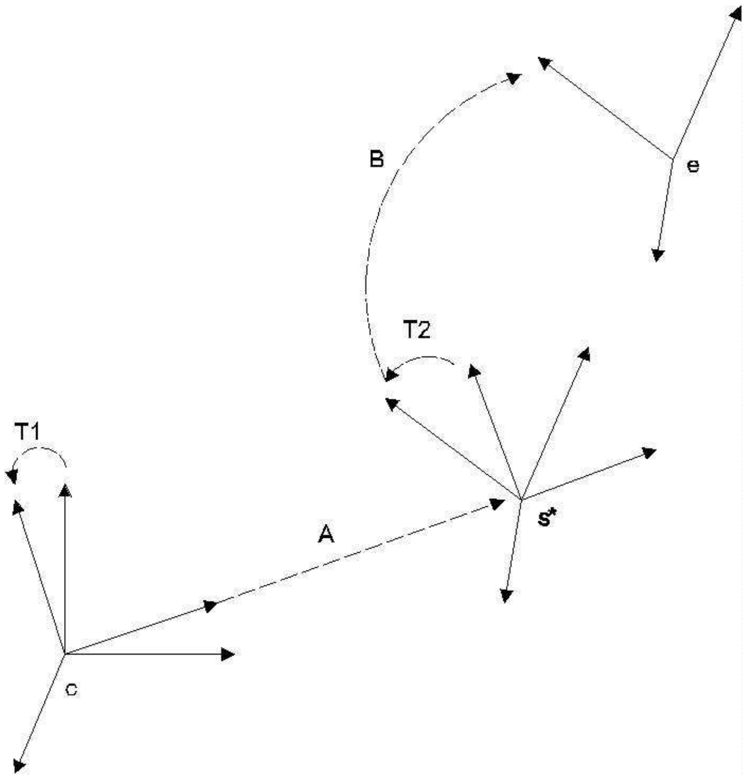

Figure 15: A model of the shoulder clavicle complex.

The forward kinematics equation for the tip of the humerus can be written as

(6) $\mathbf{T}_1(\theta_1, \theta_2)\mathbf{A}\mathbf{T}_2(\theta_3, \theta_4, \theta_5)\mathbf{B}$,

where $\mathbf{T}_1 = \begin{bmatrix} \mathbf{R}_1(\theta_1, \theta_2) & 0 \\ \mathbf{0} & 1 \end{bmatrix}$ and $\mathbf{T}_2 = \begin{bmatrix} \mathbf{R}_2(\theta_3, \theta_4, \theta_5) & 0 \\ \mathbf{0} & 1 \end{bmatrix}$ are the rotation matrices induced by the clavicle and shoulder joints and $\mathbf{A} = \begin{bmatrix} \mathbf{R}_a & \mathbf{t}_a \\ \mathbf{0} & 1 \end{bmatrix}$ and $\mathbf{B} = \begin{bmatrix} \mathbf{R}_b & \mathbf{t}_b \\ \mathbf{0} & 1 \end{bmatrix}$ are constant rigid body transformations from the clavicle to the shoulder frame and from the shoulder frame to the tip of the humerus respectively.

Given a quadruple (u, v, w, α), the corresponding joint angles θ_1 and θ_2 can be determined by solving the equation

$$\mathbf{R}_1(\theta_1, \theta_2)\mathbf{t}_a = \mathbf{s}^*(u, v, w). \tag{7}$$

Finally, $\theta_3, \theta_4, \theta_5$ can be obtained by applying Euler extraction to the right hand side of the equation below

$$\mathbf{R}_2(\theta_3, \theta_4, \theta_5) = (\mathbf{R}_1\mathbf{R}_a)^{-1} \left[\ \hat{\mathbf{x}}(u, v, w, \alpha),\ \ \hat{\mathbf{z}}(u, v, w) \times \hat{\mathbf{x}}(u, v, w, \alpha),\ \ \hat{\mathbf{z}}(u, v, w)\ \right] \mathbf{B}^{-1}. \tag{8}$$

4.2 Inverse Kinematics

The system has one extra degree of freedom determined by the elbow position. Since the elbow workspace is modeled as a surface that is nonspherical, the elbow does not move on a circle corresponding to the intersection of two spheres, (as in the kinematic model described in Tolani [18]). Instead, the elbow moves about a nonplanar curve corresponding to the intersection of the surface with a sphere centered at the goal and whose radius is equal to the length of the lower arm. The first step in devising an inverse kinematics algorithm is to find a parametric representation for the elbow position curve.

4.3 Intersection Computation

One approach to determining the intersection curve is to directly substitute the equation of the sphere into the parametric equation of the spline surface. This leads to an implicit equation of total degree 6 in terms of u, v, and w. Ideally, the equation should be in parametric rather than implicit form as our approach requires a parameterization of the elbow position. In general, it is impossible to convert an implicit curve into parametric form. One could pretend the implicit equation is parametric in terms of one of u, v, and w, and to trace the curve by iterating over this parameter and solving for the other two unknowns, but this approach is not practical since we are only interested in real values of u, v, and w. Moreover, there is no general purpose closed form solution to a polynomial equation of degree greater than four, which means that a slow and potentially unreliable numerical solver must be utilized.

Traditionally, surface intersection algorithms were based on divide and conquer methods that use polyhedral subdivision and approximation. For example, the intersection of two curves can be determined with a recursive algorithm where bounding boxes are used to approximate the curves. If the bounding boxes of two curves intersect, then each bounding box is in turn broken into two or more smaller boxes, and the process is repeated until a numerical threshold has been reached. Lasser [13] developed an intersection scheme for Bézier rectangular patches based on approximating the Bézier surface with its subdivided control mesh, reducing the problem into a polygon intersection problem. More recent surface intersection algorithms used in commercial CAD systems are based on curve-tracing techniques [10]. These algorithms first find an initial point on the curve, and by stepping along the curve tangent, they obtain a crude guess for the next point. The point is then refined so that it actually lies on the intersection curve, and the procedure is repeated.

Our approach to solving the intersection problem is similar to Lasser's method, except that we use Bézier triangles instead of rectangular patches, and only one of the two surfaces is subdivided. As a result, our algorithm produces piecewise circular arcs rather than line segment approximations to the intersection curve. These approximations can be easily parameterized and refined to an arbitrary degree of accuracy. It is well known that successively subdividing a Bézier control mesh produces a new set of control meshes that converges in the limit to the surface. The rate of convergence is very rapid, which makes subdivision a practical and efficient method for drawing or approximating the surface with a control mesh. We take advantage of this fact to approximate the intersection curve by subdividing the surface into a set of triangular control polygons and then by stitching together the circular arcs of all triangles that intersect the sphere.

It is clearly impractical to test all of the triangles for intersection, and in order to make this approach practical, it is necessary to use a data structure that prunes the search space efficiently. As a first step, we use a variation of an oct-tree [7] to store the triangles approximating the surface. The basic idea behind an oct-tree is to surround a set of spatial data with a bounding cube. Each cube is in turn subdivided into eight disjoint subcubes whose union forms the original cube. This process continues recursively until a cube has reached a minimum size. In our scheme, every triangle is stored into the smallest cube that is capable of accommodating it. To determine which triangles are potential intersection candidates, we first determine whether the parent cube intersects the sphere. If the sphere intersects the cube, we test all triangles inside the cube for intersection and recursively test all of the subcubes. As a further enhancement, every triangle has an associated bounding sphere which can be used as a quick test for potential intersection. Finally, we can further increase the speed of the intersection procedure by noting that the surface tends to intersect the sphere in only one curve. In other words, if the surface intersects a triangle, there is a very high likelihood that it intersects one or more of the triangle's neighbors. This suggests that an effective approach would be to start by traversing the oct-tree until the first intersection with a triangle occurs. Rather than using the oct-tree to determine additional intersecting triangles, the neighbors of the first intersecting triangle are then recursively checked for intersection until no more neighboring triangles intersect the surface.

To test for intersection between a cube and a sphere we first find the point on the cube that is closest to the center of the sphere. The coordinates of this point $\mathbf{P}$ are given by

$$(9) \qquad \mathbf{P}_i = \begin{cases} \mathbf{B}_i^{\min} & \mathbf{C}_i < \mathbf{B}_i^{\min} \\ \mathbf{B}_i^{\max} & \mathbf{C}_i > \mathbf{B}_i^{\max} \\ \mathbf{C}_i & otherwise \end{cases}$$

for $i = 1..3$ and $\mathbf{C}$ is the center of the sphere and $\mathbf{B}^{\min}$ and $\mathbf{B}^{\max}$ are the corner points of the cube. Obviously, intersection or inclusion occurs if $(\mathbf{P} - \mathbf{C})^2$ is less than the square of the radius.

To compute the intersection of a triangle with a sphere, we determine the distance of the center of the sphere to the plane containing the triangle. If the distance is less than the radius of the sphere, then no further testing is required. Otherwise, we project the sphere onto the plane containing the triangle and perform the equivalent two dimensional intersection problem of a circle with a triangle on a plane. In general, the circle will lie outside the triangle or intersect in two points. It is theoretically possible for the circle to be completely contained within the triangle or for the triangle to intersect in other than two intersection points, but in practice, these cases do not occur unless the circle is small compared to the size of the triangle. On the rare occasions where these degenerate situations arise, the intersection algorithm can be repeated with a finer subdivision level.

The result of the intersection procedure is a list of piecewise circular arcs that approximate the actual intersection of the sphere with the surface traced by the elbow. Additionally, we also need to obtain curves that correspond to the shoulder and the wrist positions. Consider a circular arc segment with center $\mathbf{C}_0$ and radius R. Let **a,b,c** be the coordinates of the intersecting triangle and let $\mathbf{P}_0$ and $\mathbf{P}_1$ denote the two intersection points of the triangle with the sphere. The equation of the elbow $\mathbf{e}(t)$ can be obtained using a rational parameterization. First, perform a suitable coordinate transformation that maps C_0 onto the origin, the circle onto the xy plane, and $\frac{\overrightarrow{\mathbf{C}_0\mathbf{P}_0}}{\left\|\overrightarrow{\mathbf{C}_0\mathbf{P}_0}\right\|}$ onto the x axis. Let s be the y intercept of the line segment $(-1, 0)$ through $\mathbf{P}_1$ (figure 16).

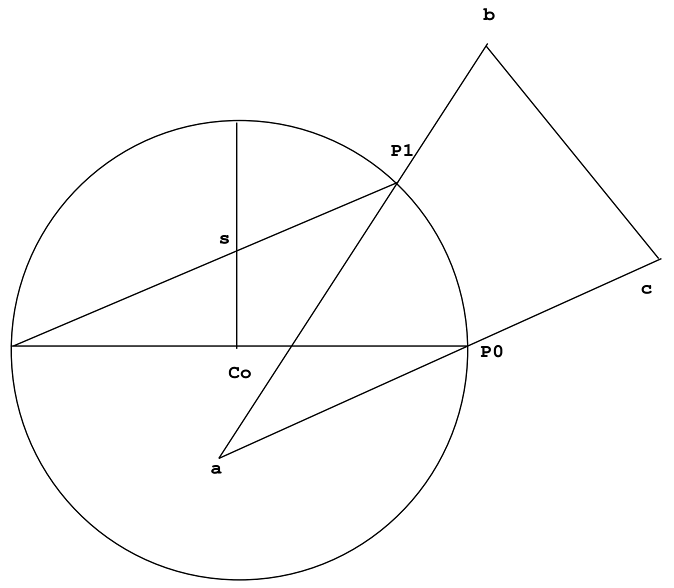

Figure 16: Obtaining a rational parameterization.

Then the equation of the elbow arc in this coordinate system is

$$\mathbf{e}(t) \quad = \quad R\left[\begin{array}{ccc} \frac{(1-t^2)}{(1+t^2)}, & \frac{2t}{(1+t^2)}, & 0 \end{array}\right]^T$$

$0 < t < s.$

To obtain $\mathbf{s}(t), \mathbf{w}_0(t), \mathbf{w}_1(t)$, we first obtain the barycentric coordinates of $\mathbf{e}(t)$ with respect to the triangle $\Delta\mathbf{abc}$. The barycentric coordinates of $\mathbf{e}(t)$, $(u(t), v(t), w(t))$ can be computed in closed form as

$$\begin{aligned} u(t) &= \frac{\Delta(\mathbf{e}(t), \mathbf{b}, \mathbf{c})}{\Delta(\mathbf{a}, \mathbf{b}, \mathbf{c})} \\ v(t) &= \frac{\Delta(\mathbf{a}, \mathbf{e}(t), \mathbf{c})}{\Delta(\mathbf{a}, \mathbf{b}, \mathbf{c})} \\ w(t) &= 1 - u(t) - v(t), \end{aligned}$$

where $\Delta(\mathbf{a}, \mathbf{b}, \mathbf{c}) = \begin{vmatrix} \mathbf{a}_x & \mathbf{b}_x & \mathbf{c}_x \\ \mathbf{a}_y & \mathbf{b}_y & \mathbf{c}_y \\ 1 & 1 & 1 \end{vmatrix}$.

The resulting expressions for $u(t), v(t)$, and $w(t)$ are quadratic rationals in t. The triangle points **a**,**b**, and **c**, correspond to control points of one of the Bézier patches. Let $\mathbf{a}_s, \mathbf{b}_s, \mathbf{c}_s$ denote the **s** components of these Bézier control points. The point $\mathbf{s}(t)$ is obtained by taking the barycentric combinations of these points:

(10) $\mathbf{s}(t) = u(t)\mathbf{a}_s + v(t)\mathbf{b}_s + (1 - u(t) - v(t))\mathbf{c}_s.$

The points $\mathbf{w}_0(t)$ and $\mathbf{w}_1(t)$ are obtained analogously.

At this point, one might ask why we bother to use Bézier surfaces at all. If the intersections are performed on triangles rather than the surface, then all that is needed is a triangularization of the data. Bézier surfaces are still preferable for a number of reasons. Obtaining a triangularization of irregularly spaced random data is a difficult problem. Additionally, all the triangles must be small to ensure accuracy. This means that the data must be carefully collected to be uniformly and finely sampled. Bézier surfaces solve both these problems for free. Moreover, Bézier surfaces are more compact than storing the entire data set and they allow the surface to be refined to whatever level of accuracy is required by the application. By contrast, a triangularization must always be stored at the finest level of accuracy that can be requested.

4.4 Computing joint angles

For a desired goal position **g**, the intersection algorithm produces a curve

$$[\mathbf{e}(t), \mathbf{s}(t), \mathbf{w}_0(t), \mathbf{w}_1(t)]^T .$$

$\mathbf{e}(t)$ represents all possible locations of the elbow such that the distance of the elbow to the goal is equal to the length of the lower arm. The variable $\mathbf{s}(t)$ encodes the location of the shoulder given by equation 10. Finally, $\mathbf{w}_0(t)$ and $\mathbf{w}_1(t)$ give the limits on the range of twist rotation about the upper arm. We assume that the elbow joint θ_6 is a revolute joint about the y-axis, and the wrist joints θ_7, θ_8, and θ_9 are spherical joints. We also assume that the minimum and maximum limits for these joints are constant. The clavicle joints θ_1 and θ_2 can be computed using equation 7, and θ_6 is determined by Euler angle extraction (for details, see Tolani [18]). For simplicity, assume that the rotations $\theta_3, \theta_4, \theta_5$ are about the x, y, and z axes respectively and that $\mathbf{R}_b = \mathbf{I}$ and $\mathbf{t}_b = [0, 0, L_2]^T$, where L_2 is the length of the upper arm. It is not necessary to infer joint limits for $\theta_1, \theta_2, \theta_3$, and θ_4, as they are automatically satisfied by the requirements that the elbow position lies on the surface and that the shoulder position is given by equation 10. Since θ_5 does not affect the position of $\mathbf{e}$, θ_3 and θ_4 can be calculated by solving

$$\mathbf{R}_x(\theta_3)\mathbf{R}_y(\theta_4)[0, 0, L_2]^T = \mathbf{R}_1^T\mathbf{R}_a\mathbf{e} - \mathbf{R}_1^T\mathbf{t}_a.$$

The variable θ_5 is not arbitrary but constrained by $\mathbf{w}_0(t)$ and $\mathbf{w}_1(t)$. To determine the joint limits $\theta_5^{\min}(t)$ and $\theta_5^{\max}(t)$, construct the matrices $\mathbf{E}_0$ and $\mathbf{E}_1$ representing the transformation matrices associated with $\mathbf{w}_0(t)$ and $\mathbf{w}_1(t)$, as described in section 4.1. Let $\mathbf{R}_{E_0}$ and $\mathbf{R}_{E_1}$ denote the rotational components of $\mathbf{E}_0$ and $\mathbf{E}_1$. To calculate the value of $\theta_5^{\min}$ associated with $\mathbf{w}_0$, we note that

$$\begin{aligned}\mathbf{R}_z(\theta_5) &= (\mathbf{R}_1\mathbf{R}_A\mathbf{R}_x(\theta_3)\mathbf{R}_y(\theta_4))^T \mathbf{R}_{E_0} \\ &= \mathbf{S},\end{aligned}$$

which implies that $\theta_5^{\min} = a\tan 2(\mathbf{S}_{01}, \mathbf{S}_{11})$. The value of $\theta_5^{\max}$ can be obtained analogously.

For the time being, assume there are no constraints on the joint limits. Let $\mathbf{C} = \begin{bmatrix} \mathbf{R}_c & \mathbf{t}_c \\ \mathbf{0} & 1 \end{bmatrix}$ denote the rigid body transformation from the distal frame of the elbow to the proximal frame of the wrist. To obtain θ_5 for a given goal $\mathbf{g}$ and value of t, we solve

$$\mathbf{R}_z(\theta_5)(\mathbf{t}_b + \mathbf{R}_y(\theta_6)\mathbf{t}_c) = (\mathbf{R}_1\mathbf{R}_A\mathbf{R}_x(\theta_3)\mathbf{R}_y(\theta_4))^T\mathbf{g} - \mathbf{R}_1\mathbf{t}_a$$

for θ_5. The wrist angles are obtained by performing Euler angle extraction on the matrix

(11) $(\mathbf{R}_1\mathbf{R}_A\mathbf{R}_x(\theta_3)\mathbf{R}_y(\theta_4)\mathbf{R}_z(\theta_5)\mathbf{R}_y(\theta_6)\mathbf{R}_c)^T\mathbf{G}.$

Of course, it is useful to automatically determine a suitable value for t that satisfies the joint limits on $\theta_5, \theta_7, \theta_8$, and θ_9. One approach is to optimize a penalty function such that the objective is positive when a joint limit is violated and zero when all joint limits are satisfied. A simple example is

$$f(t) = \sum_{i \in \{5,7,8,9\}} r_i g_i^2(\theta_i(t))$$

where r_i are positive scale factors that weight the relative importance of each joint limit and

$$g_i(\theta_i(t)) = \max(0, \theta_i - \theta_i^{\max}) + \max(0, \theta_i^{\min} - \theta_i)$$

is zero whenever θ_i obeys the joint limits and is positive otherwise. Obtaining a minimum of $f(t)$ can be solved as a root finding problem or an optimization problem in one variable.

Although the derivatives of θ_i can be computed analytically with the assistance of a symbolics mathematics package, the closed formulas for the derivatives are quite complex and expensive to compute. From a practical point of view, it is simpler and faster to approximate the derivatives using a finite difference approximation since the function itself can be evaluated efficiently. It is known that a finite difference approximation works particularly well with Newton's method in systems with a small number of variables. Additionally, since the problem only involves one variable, a number of search based techniques that do not require derivatives can be used. The most popular is quadratic fitting where the function is sampled at three points about a starting position and fitted with an interpolating quadratic approximation. The minimum of the quadratic is computed and used as the next guess iteratively.

Unfortunately, successfully finding a minimum of $f(t)$ does not necessarily guarantee that the joint limits will always be enforced. For example, if the goal lies outside the reachable workspace, then no value of t will satisfy all the joint limits. However, in general, the position constraints on an arbitrary goal can usually be solved provided that the goal does not extend beyond the length of the arm. Therefore, it seems to make sense to find a solution that satisfies the position constraint while minimizing the orientation error. One method is to treat the wrist angles as independent variables from t. This allows us to obtain solutions for the wrist angles that no longer are forced to obey orientation constraints which cannot be enforced. We first minimize $r_5 g_5^2(\theta_5(t))$ to obtain a solution for θ_5 that tends to minimize the position error while satisfying the joint limits. To solve for the wrist angles, we solve another optimization problem

$$f(\theta_7, \theta_8, \theta_9) = \|x(\theta_7, \theta_8, \theta_9) - x_g\| + \|y(\theta_7, \theta_8, \theta_9) - y_g\|,$$

subject to the inequality constraints on the wrist joints. The variables $x(\theta_7, \theta_8, \theta_9)$ and $y(\theta_7, \theta_8, \theta_9)$ are the x and y columns of the rotation matrix induced by the wrist angles and x_g and y_g are the corresponding columns of the matrix of equation 11.

4.5 Conclusion

The surface-based model can also be used to solve problems involving partial orientation and aiming (for details, see Tolani [18]). A partial orientation problem introduces an extra degree-of-freedom ψ. The goal position is independent of ψ, so θ_5 can be computed as before. However, the wrist angles and the corresponding optimization function are now functions of both t and ψ.

Aiming constraints also introduce one extra degree of freedom r, which parameterizes the distance of the goal to the tip of the elbow. Although we have not addressed this problem, it is probably not difficult to derive the required hand position and θ_5 for given values of r and t. However, it is not wise to attempt to optimize the objective function with respect to r, since the relationship between r and the elbow curve is complex and requires a surface intersection computation. Instead it seems best to require the user to prescribe a given r and to treat the problem as a one dimensional optimization problem in terms of t.

Our work has demonstrated the feasibility of computing arm trajectories (in both position and orientation) customized to a particular individual's anatomical structure from sparsely sample motions. The method is geometrically formulated as a triangular Bezier surface intersection problem and is solved by repeatable, consistent, and analytic techniques.

5 Appendix: Bézier Curves and Bézier Triangles

5.1 Bézier Curves

In this section, we provide a brief and informal review of polynomial curves and surfaces. We only cover the minimal amount of material necessary to understand the surface interpolation scheme used in section 4, and make no attempt to be rigorous or exhaustive. We refer the reader to Gallier [8] and Hoschek [11] for a comprehensive survey of this area.

5.2 The de Castlejau Algorithm

We can connect two points $\mathbf{p}_0$ and $\mathbf{p}_1$ with a line segment using linear interpolation, getting

$$F(u) = (1-u)\mathbf{p}_0 + u\mathbf{p}_1,$$

where $0 \leq u \leq 1$. Similarly, given three points $\mathbf{p}_0, \mathbf{p}_1$, and $\mathbf{p}_2$, we can construct a quadratic curve connecting $\mathbf{p}_0$ and $\mathbf{p}_2$ by successive linear interpolants as shown in figure 17.

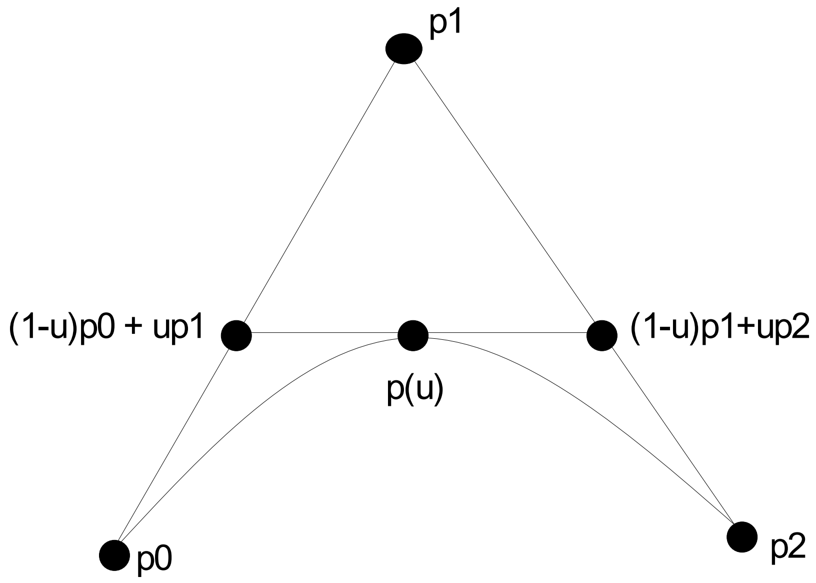

Figure 17: Evaluating a quadratic Bézier curve.

$$\mathbf{p}_0^1 = (1-u)\mathbf{p}_0 + u\mathbf{p}_1$$

$$\begin{aligned}\mathbf{p}_1^1 &= (1-u)\mathbf{p}_1 + u\mathbf{p}_0 \\ F(u) &= (1-u)\mathbf{p}_0^1 + u\mathbf{p}_1^1.\end{aligned}$$

In general, this idea can be generalized to n points using the de Castlejau algorithm:

```
deCastlejau(p[0],...,p[n], u)
for i=0 to n
    for j=0 to n-1
        p[j] = (1-u)p[j] + u*p[j+1]
    end
end
return p[0]
```

The polynomial curve of degree n formed by applying de Castlejau's algorithm is called a *Bézier curve*, and $\mathbf{p}_0, ..., \mathbf{p}_n$ are called the *control points*. A Bézier curve can also be written in closed form as

$$B(u) = \sum_{i=0}^{n} \binom{n}{i} (1-u)^{n-i} u^i \mathbf{p}_i. \tag{12}$$

5.3 Blossoms

Given any polynomial curve $F(u)$ of degree d and an integer $n \geq d$, there exists a unique n-variable polynomial function $f(u_1, ..., u_n)$ called the *blossom* or *polar form* of F that satisfies the following properties:

1. f is n-affine: $f(..., \lambda s + \mu t, ...) = \lambda f(...., s, ...) + \mu f(..., t, ...)$, if $\lambda + \mu = 1$.
2. f is symmetric: $f(u_1, ..., u_n) = f(u_{\pi(1)}, ..., u_{\pi(n)})$, for every permutation π.
3. f agrees with F along the diagonal: $f(u, ..., u) = F(u)$, for all $u \in \Re$.

As a notational convenience, we will use u^n to denote u repeated n times. Given the blossom $b(u_1, ..., u_n)$ of a Bézier curve $B(u)$, we can find the Bézier control points by taking advantage of the affine properties of b:

$$B(u) = (1-u)b(0, u^{n-1}) + ub(1, u^{n-1}).$$

If we repeat this process we obtain the equation

$$B(u) = \sum_{i=0}^{n} \binom{n}{i} (1-u)^{n-i} u^i b(0^{n-i}, 1^i),$$

which reveals that the Bézier control points $\mathbf{p}_i$ are given by

$$\mathbf{p}_i = b(0^{n-i}, 1^i).$$

Blossoms are an extremely powerful tool for analyzing polynomial curves and surfaces. They provide a unifying framework for studying curve and surface evaluation, derivatives, subdivision, reparameterization, basis conversion, degree raising, knot insertion, and interpolation. Moreover, generalizations of the de Castlejau algorithm can be used in conjunction with blossom values to provide numerically stable and straightforward procedures for solving these problems. For example, it is frequenly useful to subdivide a Bézier curve $B(u)$ into two subcurves whose union is equivalent to the original curve. To split $B(u)$ at a parametric value s, we only need to find two curves whose control points are given by $b(0^{n-i}, s^i)$ and $b\left(s^{n-i}, 1^i\right)$ respectively.

5.4 Triangular Bézier Patches

Blossoms and Bézier curves can also be extended to surfaces. The two most popular parametric surface schemes are tensor-product surfaces defined over rectangular control patches and Bézier triangles which are defined over triangular patches. We only discuss Bézier triangles which are the most natural generalization of Bézier curves to surfaces.

5.5 Barycentric Coordinates

Given a reference triangle defined by three points **a**,**b**, and **c**, we can write any point **p** on the plane containing the triangle as a *barycentric combination* of **a**,**b**,**c**,

$$\mathbf{p} = u\mathbf{a} + v\mathbf{b} + w\mathbf{c}$$

where

$$u + v + w = 1.$$

The components of the vector $\mathbf{u}$=$[u, v, w]^T$ are called the barycentric coordinates of **p** with respect to $\mathbf{a}, \mathbf{b}, \mathbf{c}$.

Using barycentric coordinates, the de Castlejau algorithm can be generalized to triangular control meshes to form surfaces. It is easiest to describe the algorithm using an example, and the extension to the general case is straightforward. Consider a quadratic control net with the following control points

$$\begin{array}{ccccc} & & \mathbf{b}_{200} & & \\ & \mathbf{b}_{101} & & \mathbf{b}_{110} & \\ \mathbf{b}_{002} & & \mathbf{b}_{011} & & \mathbf{b}_{020} \end{array}$$

To evaluate the surface at a point $[u, v, w]^T$, where $u+v+w = 1$, we first form an intermediate subtriangle consisting of points $\mathbf{b}^1_{001}, \mathbf{b}^1_{010}$, and $\mathbf{b}^1_{100}$

$$\begin{array}{ccc} & \mathbf{b}^1_{100} & \\ \mathbf{b}^1_{001} & & \mathbf{b}^1_{010} \end{array}$$

$$\begin{aligned}\mathbf{b}^1_{001} &= u\mathbf{b}_{101} + v\mathbf{b}_{011} + w\mathbf{b}_{002}\\ \mathbf{b}^1_{010} &= u\mathbf{b}_{110} + v\mathbf{b}_{020} + w\mathbf{b}_{011}\\ \mathbf{b}^1_{100} &= u\mathbf{b}_{200} + v\mathbf{b}_{110} + w\mathbf{b}_{101}.\end{aligned}$$

Finally, the point on the curve is determined by evaluating the barycentric coordinates with respect to $\mathbf{b}^1_{001}, \mathbf{b}^1_{010}$, and $\mathbf{b}^1_{100}$:

$$b^2_{000} = u\mathbf{b}^1_{100} + v\mathbf{b}^1_{010} + w\mathbf{b}^1_{001}.$$

The general de Castlejau algorithm for a triangular patch is as follows:

$$\mathbf{b}^r_i(\mathbf{u}) = u\mathbf{b}^{r-1}_{\mathbf{i+e1}}(\mathbf{u}) + v\mathbf{b}^{r-1}_{\mathbf{i+e2}}(\mathbf{u}) + w\mathbf{b}^{r-1}_{\mathbf{i+e3}}(\mathbf{u}),$$

where

$$\begin{aligned}\mathbf{e}_1 &= [1,0,0]^T, \mathbf{e}_2 = [0,1,0]^T, \mathbf{e}_3 = [0,0,1]^T\\ r &= 1...n\\ |\mathbf{i}| &= n - r,\end{aligned}$$

and $\mathbf{b}^0_i$ are the control points and $\mathbf{b}^n_{ooo}(\mathbf{u})$ is the point on the Bézier triangle evaluated at parameter value $\mathbf{u}$.

5.6 Triangular Blossoms

In the case of surfaces defined by triangular control meshes, the domain parameter is allowed to range over $\Re^2$ instead of $\Re$. For every bivariate polynomial $B(\mathbf{u})$ of degree n, where $\mathbf{u} \in \Re^2$, there exists a unique blossom $b(\mathbf{u}_1, ..., \mathbf{u}_n)$ that is symmetric, multiaffine, and agrees with B on the diagonal. As in the case of curves, it can be shown that the control point $\mathbf{b}_{ijk}$ of a Bézier triangle $B(\mathbf{u})$ are given by the blossom values $b(\mathbf{e}^i_1, \mathbf{e}^j_2, \mathbf{e}^k_3)$, where $i + j + k = n$.

5.7 Subdivision

Subdivision of Bézier triangles follows naturally from blossoming. Evaluating a Bézier patch whose domain triangle is given by $\triangle\mathbf{uvw}$ at $\mathbf{t}$ generates a set of intermediate control points that break the surface into three pieces about $\mathbf{t}$, defined by domain triangles $\triangle\mathbf{uvt}, \triangle\mathbf{tvw}$, $\triangle\mathbf{utw}$. In general, many subdivision schemes are possible, but perhaps the most natural subdivision method is to use a regular partition of a triangle patch into four subpieces. We illustrate this scheme with the following example. Consider a quadratic patch defined over the triangle $\triangle\mathbf{uvw} = ((1,0,0),(0,1,0),(0,0,1))$, with control points

$$(b(w,w), b(v,w), b(v,v), b(u,w), b(u,v), b(u,u)).$$

We can split the patch into four subtriangles $\triangle\mathbf{tsw}$, $\triangle\mathbf{str}$, $\triangle\mathbf{uvs}$, and $\triangle\mathbf{urt}$, where $\mathbf{r} = \left(\frac{1}{2}, \frac{1}{2}, 0\right)$, $\mathbf{s} = \left(0, \frac{1}{2}, \frac{1}{2}\right)$, and $\mathbf{t} = \left(\frac{1}{2}, 0, \frac{1}{2}\right)$, by evaluating the appropriate blossom points as shown in figure 18. Gallier [9] shows how to implement this subdivision scheme using only four calls to the de Castlejau algorithm.

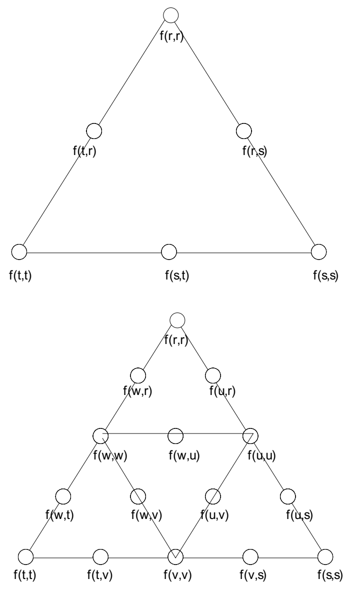

Figure 18: Dividing a Bézier Triangle.

5.8 C^1-Continuity between Triangular Patches

It is frequently necessary to construct a surface out of several patches that are stitched together. In order to ensure that the patches join smoothly, various conditions on the control points must be satisfied. In this section, we only summarize the conditions necessary for C^0 and C^1-continuity. Let $A = \triangle\mathbf{prs}$ and $B = \triangle\mathbf{qrs}$ be two triangles in parametric space sharing the edge $\mathbf{rs}$. Let B_A and B_B represent two corresponding surfaces of degree m for A and B with blossom forms b_A and b_B. The C^0-continuity conditions require that

$$\begin{aligned} b_A\left(\mathbf{r}^k, \mathbf{s}^{m-k}\right) &= b_B(\mathbf{r}^k, \mathbf{s}^{m-k}) \\ & \ 0 \le k \le m. \end{aligned}$$

This condition states that B_A and B_B share the same control points along the edge $\mathbf{rs}$.

Suppose $\mathbf{q} = u\mathbf{p} + v\mathbf{r} + w\mathbf{s}$. For C^1-continuity, we must also have

$$\begin{aligned} b_B(\mathbf{q}, \mathbf{r}^k, \mathbf{s}^{m-k-1}) &= u b_A(\mathbf{p}, r^k, \mathbf{s}^{m-k}) + \\ & \ v b_A(\mathbf{r}^k, \mathbf{s}^{m-k}) + \\ & \ w b_A(\mathbf{r}^{k+1}, \mathbf{s}^{m-k-1}). \end{aligned}$$

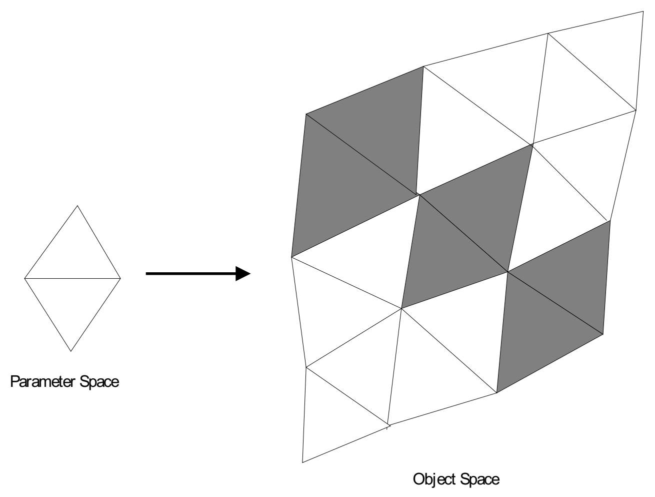

Figure 19: C^1-continuity conditions for triangular patches.

In the special case where the domain triangles are equilateral, then

$$\mathbf{p} + \mathbf{q} = \mathbf{r} + \mathbf{s},$$

and the midpoints of the line segments $(b_A(\mathbf{p}\mathbf{r}^k\mathbf{s}^{m-k-1}),\, b_A(\mathbf{r}^{k+1}, \mathbf{s}^{m-k-1}))$, and $(b_A(\mathbf{p}, \mathbf{r}^k,\, \mathbf{s}^{m-k-1}), b_A(\mathbf{r}^k, \mathbf{s}^{m-k}))$, are equal (figure 20). Higher degree continuity places further restrictions on the control points. The reader is referred to Gallier [8] for proofs and a treatment of the general case.

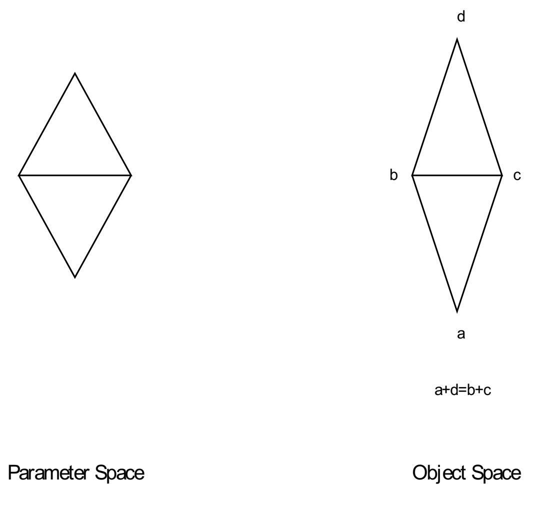

Figure 20: C^1-continuity when the parameter triangles are equilateral.

References

[1] Yahya Aydin and Masayuki Nakajima. Realistic articulated character positioning and balance control in interative environments. In *Computer Animation '99*, pages 160–168. IEEE, 1999.

[2] J. Brown. Vertex based data dependent triangulations. *Computer Aided Geometric Design*, 8(3):239–251, 1991.

[3] A. Engin and S. Chen. Statistical data base for the biomechanical properties of the human shoulder complex I: Kinematics of the shoulder complex. *Journal of Biomechanical Engineering*, 108:215–221, 1986.

[4] A. Engin and S. Chen. Statistical data base for the biomechanical properties of the human shoulder complex II: Passive resistive properties beyond the shoulder complex sinus. *Journal of Biomechanical Engineering*, 108:222–227, 1986.

[5] A. Engin and S. Tumer. Three-dimensional kinematic modeling of the human shoulder complex-part I: Physical model and determination of joint sinus cones. *Journal of Biomechanical Engineering*, 111:107–112, 1989.

[6] A. Engin and S. Tumer. Three-dimensional kinematic modeling of the human shoulder complex-part II: Mathematical modelling and solution via optimization. *Journal of Biomechanical Engineering*, 111:113–121, 1989.

[7] J. Foley, A. Van Dam, S. Feiner, and J. Hughes. *Computer Graphics*. Addison Wesley, New York, 1990.

[8] J. Gallier. *Curves and Surfaces In Geometric Modeling: Theory And Algorithms*. Morgan Kaufmann, San Fransciso, 1999.

[9] J. Gallier and D. De Carlo. On the efficiency of strategies for subdividing polynomial triangular surface patches. Technical report, University of Pennsylvania, 1997.

[10] C. Hoffman. *Geometric and Solid Modeling*. Morgan Kaufmann, San Fransciso, 1989.

[11] J. Hoschek and D. Lasser. *Computer Aided Geometric Design*. A K Peters, Boston, 1993.

[12] V. Inman, J. Saunders, and L. Abbott. Observations on the function of the shoulder joint. *Journal of Bone and Joint Surgery*, 26a:1–30, 1944.

[13] D. Lasser. Intersection of parametic surfaces in the Bernstein-Bézier representation. *Computer-Aided Design*, 18:186–192, 1986.

[14] G. Monheit and N. Badler. A kinematic model of the human spine and torso. *IEEE Computer Graphics and Applications*, 11(2):29–38, 1991.

[15] Franck Multon, Jean-Luc Nougaret, Bruno Arnaldi, Gérard Hégron, and Luc Millet. A software system to carry-out virtual experiments on human motion. In *Computer Animation '99*, pages 16–23. IEEE, 1999.

[16] E. Otani. Software tools for dynamic and kinematic modeling of human motion. Technical report, University of Pennsylvania, 1989.

[17] E. Stollnitz, T. DeRose, and D. Salesin. *Wavelets for Computer Graphics.* Morgan Kaufmann, San Fransciso, 1996.

[18] D. Tolani, A. Goswami, and N. Badler. Real-time inverse kinematics techniques for anthropomorphic limbs. *GMIP*, 1999. Submitted for publication.

[19] D. Wiley and J. Hahn. Interpolation synthesis of articulated figure motion. *IEEE Computer Graphics and Applications*, 17(6):39–45, 1997.