Arm-Posture Creation Based on Iso-Comfort Regions

(等姿勢領域に基づいた現実感のある腕の姿勢決定)

Yahya Aydin†, Hiroki Takahashi† and Masayuki Nakajima† (Member)

Abstract Finding realistic arm postures for synthetic actors is a complex and important problem in the field of human motion research. But an achievable method for creating them has not yet been proposed. The problem of creating realistic arm posture can be solved in two main steps using the presented algorithm. First, hand posture is created based on a database which is formed using an octree data structure. Classifying the reachable space and forming a database seriously simplifies the creation of hand posture. Second, the posture of the rest of the arm is determined by using iso-comfort regions; a concept introduced by the authors. Obtained results prove the effectiveness of iso-comfort regions from the standpoint of realism.

1. Introduction

Realistic arm motion has been the focus of an increasing number of studies in the last decade. Although the motion of the arm has been deeply investigated for many years, especially in the robotics literature, the need for its realistic motion has become urgent due to the emergence of synthetic-actors. In a virtual environment, synthetic-actors have to interact with many types of virtual-objects around them. The objects can have constant or varying shapes and sizes. Moreover, they can be stable or mobile. For such environments, it is necessary to have an algorithm for grasping realistically.

The problem of creating a realistic arm-posture can be divided into two main sub-problems. The first one is placing the hand over the grasped object, and the second one is finding the posture of the rest of the arm. Although the solutions of the problems are interdependent, the proposed algorithm solves the first problem by using a database of which data is obtained from the observations of real postures, and thus the posture of the rest of the arm does not need to be known while solving the first problem.

1.1 Finding the hand-posture

In order to create a realistic arm-motion, the problem of obtaining a realistic hand-posture over the grasped object must first be solved. Some methods have been proposed for natural looking human grasping. But these researches usually focused on the shape of the grasped object in determining the orientation and position of the hand, and they did not present a tangible method considering the relationship between the posture of the hand and the posture of the grasped object with respect to the body. Observations show that this relationship and the shape of the object are the most important determinants of the hand-posture over the grasped object, though there are many other factors affecting the posture of the hand.

The method proposed by the authors solves the problem of realistic hand-posture generation in an
efficient way\textsuperscript{1}. A database is formed based on the observations, and the orientation of the hand for a given object is obtained after applying necessary rotations to the stored data. The position of the hand is calculated based on the size and shape of the object. Any object to be grasped is first approximated with a simple primitive; this idea was first introduced in the robotics literature\textsuperscript{4}. The use of octree data structure provides both classification of the reachable space and efficiency in data-storage. In this paper, the term "posture" refers to both position and orientation together.

1.2 Finding the arm-posture

Numerical and analytical methods have been proposed to solve the problem of reach to grasp\textsuperscript{7}. In each approach, different mechanisms are utilized to reduce the number of degrees-of-freedom, that is the number of variables that can be changed independently. In numerical approaches, an objective function is used to define a unique solution. An important deficiency of these methods is the difficulty of finding such an objective function that leads to a realistic posture when minimized. Numerical methods have been successfully applied to robot-arms, but they usually do not give realistic results when used for human arm. Analytical approaches have the problem of choosing appropriate angles for joints, because only a small subset of the solution set gives realistic-looking images.

The solution proposed in this paper is mainly based on an important observation. For any arm-posture, comfort levels of the joints are very close to each other unless it is necessary to strain one joint more than the others. In other words, the burden of the arm is distributed as equally as possible among its joints. For that reason, a method minimizing only the total comfort level of the joints does not result in a realistic solution. Based on this important observation, the proposed algorithm has two main functions; the first one is related with total comfort level, and the other one is associated with the closeness of these levels to each other. For any object given in the reachable space, after fixing the position of the wrist on it, the arm-posture resulting in the maximum weighted sum of two functions is chosen as the solution.

In section 2, the method used to find the wrist position is explained. In section 3, definition of iso-comfort regions is illustrated. In section 4, manipulation of these regions is explained in detail. Experimental results are given in section 5.

2. Finding the position of the wrist

Finding the posture of the hand, and consequently the position of the wrist, is the first step of the algorithm. Classification of the reachable space is explained in section 2.1, and how to form and use the database is illustrated in section 2.2.

2.1 Classification of the reachable space

An object in the reachable space can be at infinitely many positions, and for each position, it can have infinitely many orientations. The purpose of classification is to reduce the number of these positions and orientations to a reasonable number so that it becomes possible to form a database in terms of the datasize and manual work. To justify the idea of classification, as an example, consider an object in front of the body and at the level of the chest. If the object is moved in its near vicinity, keeping the orientation constant, the change in the orientation of the hand is negligible. The object can have any orientation, but it is an important point to keep that orientation constant during the motion. Generally speaking, in the reachable space, while grasping an object with constant orientation, there are certain subspaces in which the change in the orientation of the hand is negligibly small. The crucial point in classification is the determination of these subspaces. Initially, we start with an octree of only one volume. The size of the octree must be selected so as to surround the reachable space. The volume occupied by the octree is divided into subvolumes based on the following test: When an object with constant orientation is randomly moved inside a subvolume, if the change in the orientation of the hand is negligibly small, the size of that subvolume is left unchanged, otherwise it is divided into eight smaller subvolumes and the same process is repeated for new subvolumes. There is a trade-off between the size of the subvolumes and the level of the realism. As an extreme case, we can divide the reachable space into very small subvolumes
such that a different data is stored for almost every point in the reachable space. But in that case the amount of the stored data becomes extremely huge, though the level of the attained realism also increases. A subvolume can be defined as a sufficiently small block such that an object with constant orientation, regardless of its position in the subvolume, can be grasped realistically with the same hand orientation.

The size of the subvolumes varies through the reachable space. Because at some points in the reachable space, a small change in the position of an object with constant orientation causes drastic changes in the orientation of the hand grasping that object. An example to this case is when an object which is slightly over the shoulder is moved just to the bottom of the shoulder. But at some points such as in front of the body, the change in the orientation of the hand is considerably small compared to that of the previous example. Due to this variant characteristic of the space, octree that enables the division of different parts of the space at different levels is an appropriate data structure, as shown in Figure 1. From the above discussion, a subvolume of the octree can be larger than others if there is no need to divide it any further.

Division of the reachable space by using an octree data structure is the first level classification. After determining the subvolumes of the octree, in each subvolume, the object can be at infinitely many orientations. These orientations are also classified by using a finite number of axes, which can be thought as the second level classification. The data is stored only along those axes. While applying the algorithm for a given object, if the object does not lie along the axes used for classification, the orientation of the hand is found using the data of the closest axis as explained in the next section. Similar to the first level classification, the number of axes can be increased according the level of the realism we need.

The octree is defined with respect to a coordinate system which is fixed to the body at an appropriate point between the shoulders.

2.2 Forming and using the database

The database is composed of two main parts. The first part consists of the vectors representing the axes which are used for the second level classification. The number of these axes need not be the same for all subvolumes, because some subvolumes may require more axes for efficient classification of the object-orientations. The second part of the database contains the vectors representing the hand orientations, and the vectors used to store the the orientation of the object at the moment of grasping. For each primitive type and for each axis, the orientation of the hand is stored for its most comfortable situation. Two vectors are sufficient to represent a hand orientation, as shown in Figure 2.

As an example, for a randomly selected subvolume, assume that ten axes have been used to classify the object-orientations. These axes are stored in the database by using ten vectors. Assuming that five primitive object types are defined (block, cylinder, cone etc.), for each axis and for each of the five primitive types, the most realistic hand orientation is stored in the database together with the orientation of the grasped object. For primitive types such as cone, the direction of the object should also be considered.

In our implementation, we have defined three primitive object types: cylinder, block and sphere.
For each primitive type, forming and using the database is explained below in detail.

To determine the data of which subvolume is to be used to find the hand-posture for a given object, provided that it is in the reachable space, the subvolume or subvolumes containing the object are found. Assuming that the object has almost homogeneous mass distribution and a reasonable size to be able to grasp with one hand, the data of the subvolume including the mass center of the object is used to find the hand-posture. While the mass center of the object is not in the reachable space, as a singularity, if some part of it is in the reachable space, this situation is handled specially.

A cylindrical object lies along an axis if it is parallel to the axis. In other words, if a line which is perpendicular to the circular faces of the cylindrical object is parallel to an axis, then the object lies along that axis. In Figure 4, the object lies along the $\ell$-axis. For a block, if an axis is perpendicular to one of its surfaces, then the object lies along that axis. A spherical object lies along an axis for all orientations of the object.

How to apply the method for a cylindrical object is explained with an example in Figure 4. In that example, it is assumed that data is stored along three main axes, as shown in Figure 3. For a cylindrical object, there is no need to use an additional vector to store the orientation of the cylinder at the moment of grasping, because the object lies along the axis.

For a given object which is approximated with a cylindrical primitive, the angles between the line passing through the object and main axes are calculated, and the data of the axis resulting in the minimum angle is used for later steps.

In Figure 4, it is assumed that the minimum angle is $\alpha$, and the data of $y$-axis is used. Namely,

$$\alpha = \angle DOB = \min\{\angle DOA, \angle DOB, \angle DOC\}$$

For equality cases, the angle causing the weighted-minimum-change in the orientation and position of the hand with respect to its previous posture is chosen. In Figure 4, the desired hand orientation is obtained by rotating the stored orientation around the line coinciding with $\overrightarrow{r}$, where

$$\overrightarrow{r} = \overrightarrow{e} \times \overrightarrow{g}$$

as shown in the figure.

Some hand-postures obtained for randomly selected object orientations are displayed in Figure 5. The results belong to the same subvolume of the octree, and the hand orientations shown in Figure 3 have been used as the stored data of that subvolume.

For a block, hand orientations are stored along the axes which are used for the second level classification. To store the orientation of the object at
the moment of grasping, as different from that of the cylindrical object, an additional vector is needed to store the grasped faces. In order to simplify the algorithm, it has been assumed that the block has a size such that it is possible to grasp it by using any pair of its opposite faces. If it is not, the problem can be solved by defining priorities and storing data for each pair of the opposite faces.

For a spherical object, only one orientation is stored for each subvolume of the octree. This is because the orientation of a hand grasping a spherical object is independent of the orientation of the object if its mass distribution is almost homogeneous. Contrary to the methods proposed so far, the problem of finding a hand-posture over a spherical object is easily solved by using this method. Since the obtained orientations are same with the stored orientation for each subvolume of the octree, implementation results have not been presented.

3. Defining iso-comfort regions

Observations show that for the motion of each joint it is possible to define certain ranges where the difference in the comfort level of the joint is negligible. Such a classification significantly simplifies the solution. All joints of the arm have their own idiosyncrasies. Moreover each joint shows a serious irregularity in terms of the strain experienced on it during its motion at different directions. Therefore observations are very important to determine the characteristics of the joints. In our algorithm, the range of the motion of each joint has been divided into some sub-ranges according to the degree of the comfort felt on the joint during its motion. In the paper, the ranges of almost the same comfort level have been called iso-comfort regions.

For spherical joints, iso-comfort regions are defined on spheres and approximated with spherical polygons. The sphere of the shoulder is fixed to the body, and the sphere of the wrist is fixed to the hand. The first step in defining the iso-comfort regions for spherical joints is the determination of the prohibited regions. These regions have been represented with −1 in our implementation. It is physically impossible to cross the prohibited regions for both upper arm and forearm. After setting the borders of the prohibited region, the remaining part of the sphere-surface is roughly divided into sub-regions beginning from the most comfortable point. During this process, each joint is handled independently. The lowest and highest comfort values can be any value, but the higher the difference is, the better it is for flexibility. The important point is to assign the same lowest and highest values for all joints. The comfort values of intermediate regions are determined relative to the lowest and highest values. Since the regions on the sphere-surface are delineated by using spherical polygons, the errors of the rough definition can be decreased to a reasonable level by moving the vertices of the polygons on the sphere. Both rough definition and fine-tuning of the regions are performed based on observations and experiments.

Figure 6 shows an example for the spheres and iso-comfort regions defined on them assuming that there are five different comfort levels. Number of the levels has been chosen randomly and the regions have been approximated accordingly just to explain the idea. If the number of comfort levels is increased, better results are obtained.

After fixing the positions of the shoulder and the wrist, the elbow can be anywhere on the circle depicted in Figure 6. If the elbow is rotated around the axis connecting the wrist and the shoulder, the upper arm and the forearm also twist around their own axes, which is called "twist-motion" in the paper. Therefore, in addition to the spherical iso-comfort regions defined for the shoulder and the wrist, we must also define iso-comfort regions for each of these twist-motions. Definition of these
regions are different than those of spherical ones. Twist-motion is expressed with the angle of rotation with respect to a reference plane. Therefore, iso-comfort regions for twist-motions are defined as angle-intervals. In our implementation the reference plane has been defined as XY plane.

To set the angle-intervals, first, the arm is placed onto the coordinate system in a proper manner so that the twist angles can be expressed with respect to the reference plane. After the determination of the prohibited-region, the remaining angle-interval is divided into sub-intervals starting from the most comfortable point. Although the iso-comfort regions for twist-motions are defined as angle-intervals, the main idea is the same as that of the spherical motions.

4. Finding the realistic interval

Iso-comfort regions for spherical-motion and twist-motion are defined as explained in the previous section. In this section, first, how to find the comfort values of the joints for a given arm posture is explained in detail. In that step, the comfort values resulting from the twist-motion of the forearm and upper arm, and the comfort values resulting from the spherical motion of the shoulder and wrist are calculated separately. In the second step, the comfort values obtained in the first step are used to determine the realistic interval.

After fixing the positions of the shoulder and the wrist, the elbow is rotated around the axis shown in Figure 6. During this rotation, two circles are drawn on the spheres by the forearm and the upper arm, as illustrated in the figure. Meanwhile, the twist-motion occurs on both parts of the arm. At any moment, total number of the comfort values is four: two from spheres and two from the iso-comfort regions defined for twist-motions.

An important point to be noted is the calculation of the comfort value for twist-motions. If the elbow is transformed from one position to the other by rotating from the shoulder, it also twists around the upper arm. The degree of the twist changes with respect to the rotation axis. In our implementation the elbow has been first transformed so that it coincides with the x-axis and the elbow points to the positive direction as depicted in Figure 7. The shoulder is located to the center of the coordinate system. After this transformation which is executed by the rotation from the shoulder, the twist angle is calculated with respect to the reference plane and this angle is used to find the corresponding iso-comfort region, consequently the comfort level. But if the rotation axis is not a specially calculated one, such a transformation of the elbow causes an additional twist on the upper arm. To calculate such a special rotation axis, the following procedure is followed.

Let \( e \) be the vector pointing to the elbow. If the length of the upper arm is \( l_u \), then the vector \( v = (l_u, 0, 0) \) is the one onto which we want to transform the elbow as shown in Figure 8(a). A rotation around an axis passing through the origin and lying on the YZ plane does not change the value of the twist during the transformation explained above. But after rotation, the elbow vector \( e \) should coincide with the vector \( v \). To be able satisfy this constraint, the rotation axis must lie on a plane \( P \) which is equidistant from \( e \) and \( v \). Consequently, the rotation axis is the intersection of YZ and \( P \) planes. That is, \( P \) is the set of all points \( p \) which satisfy

\[
(v - e) \cdot p = 0
\]

Substituting \((x, y, z)\) for \( p \) in the equation, it expands to

\[
(v_x - e_x)x + (v_y - e_y)y + (v_z - e_z)z = 0
\]

Since the rotation axis lies on the YZ plane,
We can choose any vector satisfying this equation. Let \( \mathbf{a} \) be one of these vectors
\[
\mathbf{a} = (0, v_y - e_y, v_z - e_z)
\]

After finding the rotation axis, the angle of rotation must be calculated. This is the angle between the vectors \( (\mathbf{a} \times \mathbf{e} \times \mathbf{a}) \) and \( (\mathbf{a} \times \mathbf{v} \times \mathbf{a}) \) as seen from the tip of the vector \( \mathbf{a} \) and counterclockwise, as explained in Figure 8 (b).

Similar operations are applied to find the twist angle of the forearm and the corresponding iso-comfort region. The other two comfort values are directly calculated from the spheres. The problem to be solved is to find the intersection points between a circle on the sphere and the boundaries of iso-comfort regions.

After obtaining four comfort values, the interval that will give the most realistic results must be determined. For intervals in which all comfort values are greater than \(-1\), a value which is the weighted sum of two functions is calculated. Note that the regions with the comfort value of \(-1\) are prohibited regions. In an interval, if any of the comfort values is \(-1\), this interval is immediately excluded and not considered in further steps.

The first function calculates the sum of all comfort values for the current interval and normalizes it to 1.0. Let \( C_i \) is the \( i \)th comfort value, where \( 1 \leq i \leq 4 \) and \( MC \) is the maximum comfort level. The first function is
\[
F_1(C_1, C_2, C_3, C_4, MC) = \frac{(C_1 + C_2 + C_3 + C_4)}{MC}
\]

The result of the second function is related with the closeness of comfort values to each other and normalized to one. The second function is
\[
F_2(C_1, C_2, C_3, C_4, MaxC) = \left( \frac{\sum_{i=1}^{4} \sum_{j=i+1}^{4} \text{MaxC} - |C_i - C_j|}{\text{MaxC}} \right) / 6 \leq 1.0
\]

If all comfort values are equal, the function \( F_2 \) has its highest value (1.0). The final selection is done on the weighted sum of two functions.
\[
\text{WeightedSum} = \frac{3F_1 + F_2}{4} \leq 1.0
\]

In our implementation, the first function had three times more weight on the final result. This ratio has been determined based on trials. If it is between two and four, results are quite satisfactory.

5. Implementation

Experimental results are shown in Figure 9. Iso-comfort regions have been defined quite roughly to simplify the implementation.

The images in the figure have been obtained for an arm grasping a block placed on the same position, but with different orientations.

The program has been implemented in the C language and by using the OpenGL graphics system on a Silicon Graphics Indy workstation.

6. Conclusion

A method for finding realistic arm-postures has been proposed. The posture of the hand is found by using the data obtained from the observations of real postures. Octree data structure has been used...
for both classification and efficiency in data storage. As for the posture of the rest of the arm, a new concept, iso-comfort regions, and two functions manipulating these regions have been presented. The results are realistic, although they have been obtained by using roughly defined iso-comfort regions.

Much work needs to be done on the creation of realistic human-motion. In the next step of our research, we will focus on the generation of realistic arm-motion by using iso-comfort regions.

References

2) K.M.B. Bennett, U. Castiello: Insights into the Reach to Grasp Movement (1994)

Yahya Aydin is a master-course student in Tokyo Institute of Technology, Japan. His research interests include realistic human-motion and realistic image synthesis. Aydin received his BSc degree in Computer Engineering and Information Sciences from Bilkent University, Turkey, in 1993.

Hiroki Takahashi received the B.E. degree and the M.E. degree from the Tokyo Institute of Technology, Tokyo, Japan, in 1990 and 1992, respectively. He suspended the doctor course in 1994. Since 1994 he has been with the Department of Computer Science, Graduate School of Information Science & Engineering, Tokyo Institute of Technology, Tokyo, Japan, where he is now a Research Associate. His fields of interest are pattern recognition, neural networks and genetic algorithms.

Masayuki Nakajima received the B.E.E. degree and Dr. Eng. degree from the Tokyo Institute of Technology, Tokyo, Japan, in 1969 and 1975, respectively. Since 1975 he had been with Imaging Science and Engineering Laboratory, Tokyo Institute of Technology, Yokohama, Japan. He is now a professor at the Department of Computer Science, Graduate School of Information Science & Engineering, Tokyo Institute of Technology, Tokyo, Japan. His fields of interest are computer graphics, pattern recognition, image processing and virtual reality.