JOINT-DEPENDENT LOCAL DEFORMATIONS FOR HAND ANIMATION AND OBJECT GRASPING

Nadia Magnenat-Thalmann Richard Laperrière Daniel Thalmann

MIRALab, HEC/IRO
Université de Montréal
Canada

ABSTRACT

Algorithms and methods are presented for animating the hands of a synthetic actor. The algorithms allow not only to move the hand and grasp objects, but also to compute deformations of the hand as it moves, for example: rounding at joints and muscle inflations. The mapping of surfaces onto the skeleton is based on the concept of Joint-dependent Local Deformation (JLD) operators, which are specific local deformation operators depending on the nature of the joints. The major problem in the hand covering process is the calculation of the coordinate bases. The key to our method was to find a model for calculating the bases which was sophisticated enough for the simulation of complex motions. This model was then improved with additional realistic elements such as muscle inflations and joint roundings. The calculation of the coordinate bases of a vertex in 3-space is separated into two cases: single segment covering for the fingers and two-segment covering for the hand palm. For positioning the hand on an object, three processes are described: the calculation of the distance between a hand vertex and an object, the intersection calculation between the hand and the object and the semi-automatic calculation of the flexion angle, also based on the distance process. Several examples are shown and the user interface introduced into the HUMAN FACTORY system is also described.

RESUME

Des algorithmes et des méthodes d'animation des mains d'acteurs synthétiques sont présentées. Les algorithmes ne permettent pas seulement de déplacer la main et de saisir des objets, mais calcule aussi les déformations des mains lorsqu'elles bougent, par exemple les courbures aux articulations et les gonflements des muscles. L'enveloppement des surfaces sur le squelette est basé sur le concept d'opérateur de déformation locale dépendant de l'articulation (opérateur JLD). Le principal problème dans le recouvrement de la main est le calcul des bases de coordonnées. La clé de notre méthode a été de trouver un modèle de calcul des bases assez sophistiqué pour la simulation de mouvements complexes. Ce modèle a été amélioré en ajoutant des éléments réalistes tels que les gonflements de muscles et les courbures aux articulations. Le calcul des bases de coordonnées d'un sommet dans l'espace peut se faire de deux façons: recouvrement d'un segment pour les doigts et recouvrement de 2 segments pour la paume. Pour la saisie d'objets, trois approches sont décrites: le calcul de la distance entre un sommet de la main et l'objet, le calcul d'intersection entre la main et l'objet et le calcul semi-automatique de l'angle de flexion, basé aussi sur un calcul de distance. Plusieurs exemples sont présentés et l'interface usager introduit dans le système HUMAN FACTORY est décrit.

KEYWORDS: synthetic actor, hand animation, keyframe, muscle inflation, object grasping

1. INTRODUCTION

In three-dimensional character animation, the complexity of motion may be arbitrarily divided into three parts: facial animation, hand animation and body animation. A lot of attention has been devoted to facial animation[1 2 3 4 5 6 7 8] and body animation[9 10 11 12 13 14 15]. Very few scientific papers have been dedicated to hand animation: in the MOP system designed by Catmull[16], hands are decomposed into polygons, but undesired variation of finger thickness may occur. Badler[17] proposes a model based on B-spline surfaces. The surface is computed based on the skeleton of the hand; spheres are linked to the B-spline surface, which does not itself appear on the image. It is only used for the placement of the spheres.

Two problems are important in hand animation: skeleton motion and surface deformation. Moreover, a connected problem is the problem of grasping an object. To grasp an object, the hand has to be used, and the joints of the hand must move correctly. And, if an object is grasped, it has to move with the hand. When the hand is turned, the object must turn along with the hand. When an object is grasped and moved to a new position, it is the arm which essentially guides the object. However to grasp the object, flexion angles have to be determined and this problem is known in robotics as an inverse kinematics problem.

In this paper, we present algorithms and methods used to animate a hand for a synthetic actor. The algorithms allow not only to move the hand and grasp objects, but also they compute the deformations of the hands: rounding at joints and muscle inflations.

2. HAND SKELETON CONTROL

The animation of the hand is mainly based on parametric key-frame animation. The hand skeleton is defined as a connected set of segments and joints. A joint is the intersection of two segments. The angle between the two segments is called the joint angle. Fig.1 shows the left hand skeleton.

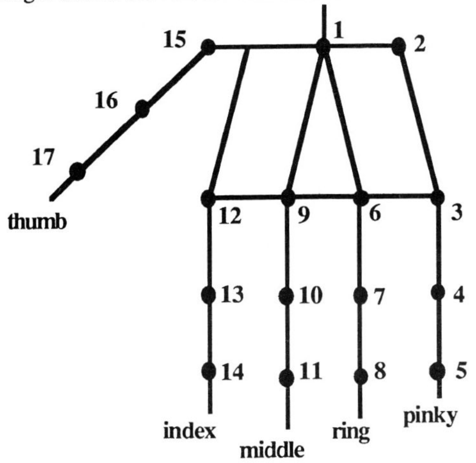

Fig.1 Left hand skeleton

Motion is specified by giving key values for each joint angle. Inbetween values are calculated using bicubic splines[18]. The animator may look at parameter values for any keyframe or interpolated frame. He/she may also obtain a wire-frame view of the hand for any frame. Fig.2 shows the hand joint angles for the left hand. We call the original skeleton (without any flexion) the **initial position** of the skeleton. Any other position is called the **final position** of the skeleton.

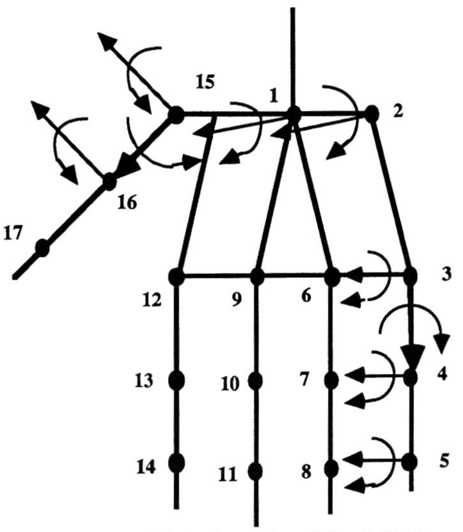

Fig.2 Joint angles for the left hand; rotations 3-4 and 15-16 are pivoting

3. JLD OPERATORS FOR HAND COVERING

Once the motion of the 3D character is designed, the hand needs to be covered with surfaces. For this, we try to completely separate the topology of the surfaces from the skeleton. This means that the hand may be constructed using any method: surfaces by sweeping, free-form surfaces or 3D reconstructed surfaces obtained from digitized projections. Our system transforms the surfaces according to the wire-frame model ensuring an automatic continuity between the different surfaces.

The mapping of surfaces onto the skeleton is based on the concept of Joint-dependent Local Deformation (JLD) operators[19], which are specific local deformation operators depending on the nature of the joints. These JLD operators control the evolution of surfaces and may be considered as operators on these surfaces. Each JLD operator will be applicable to some uniquely defined part of the surface which may be called the domain of the operator. The value of the operator itself will be determined as a function of the angular values of the specific set of joints defining the operator.

The case of the hand is especially complex, as deformations are very important when the fingers are bent, and the shape of the palm is very flexible. Segments of fingers are independent and the JLD operators are calculated using a unique segment-dependent reference system. For the palm, JLD operators use reference systems of several segments to calculate surface mapping. In order to make the fingers realistic, two effects are simulated: rounding calculations at the joints and muscle inflation. The hand mapping calculations are based on normals to each vertex at the proximal joint. These normals are computed as the cross product between the vector describing the 3D orientation of the segment associated with the vertex and the vector describing the flexion axis at this vertex. Note that this flexion axis may be modified by the animator for each joint. Other parameters are used in these mapping calculations and may be modified by the animator in order to improve the realism of muscles and joints. These parameters include:

1. a parameter to control the inflation amplitude of a joint during the flexion
2. a parameter to define the portion of segment to round during a joint flexion
3. a parameter to control the inflation amplitude of muscles inside the hand during a flexion
4. a parameter to define the location of the point where the inflation of the internal muscles is maximum during a flexion

As mapping parameters may be defined for each segment by the animator, JLD operators are only applied to one segment at once. Moreover, finger segments are independently covered, but palm segments are covered using a neighbor segment.

For each segment to be covered, the process is as follows: first, information about both joints limiting the segment are stored:

1. The 3D coordinates of the joint in the initial and the final positions of the skeleton.
2. The initial and the final flexion axes of the joint
3. The initial and final flexion angles of the joints; note that the initial angles are generally 0; but the animator may define other initial values.
4. The maximum flexion angle for the joint
5. The four user-defined parameters which characterized the realism of the surface deformation as described above: inflation amplitude of the joint, portion of segment for joint rounding, inflation amplitude of internal muscles and maximum inflation point of internal muscles
6. For palm segments, we have to determine and store the informations 1 to 5 of the joints limiting the neighbor segment, because some vertices will have their position determined by the position of the segments (and the normals at the joints) located on each side of the vertices. It means that the axis system used to define the position of these vertices in the 3-space is defined from the orientation of each segment surrounding them and from the orientation of the normals at the joints of these segments.

For the first joint of each finger and the wrist joint, pivot angles are also determined. The pivot angle is a rotation angle of the flexion axis about the segment, which allows lateral flexions of the fingers and the hand (at the wrists). From the pivot angle, the vertical component of the flexion is calculated, because only this component should be used for evaluating the inflation amplitude, without any reference to the "lateral" component. The final flexion angle used for the mapping calculation is the angle of the projection of the final segment in the plane normal to the flexion axis (pivot angle = 0), which corresponds to the original flexion axis. The angle is calculated between the segment projection (after the flexion) in the plane and the initial segment (before flexion).

4. MAPPING ALGORITHM

The mapping algorithm works as follows: first, determine the initial and final normals for both joints of the segment. Then calculate a modified normal as the average of the initial and final joint normal. This modified normal is then used as y-axis of the coordinate basis and it will allow the simulation of the external rounding of a joint during a flexion. Fig.3 shows the principle. For palm segments, normal and modified normal calculations are also required for the neighbor segments. Then, a loop is performed on all vertices associated with the segment to be covered, and for each vertex, the process is:

```
Look for the 3D coordinates of the vertex in the digitized character
Calculate the following information to localize the vertex relative to the segment:
    Determine a projection of the vertex on the segment
    Calculate the ratio
```

$$R = \frac{\text{distance between the projection and the proximal joint}}{\text{distance between the projection and the distal joint}}$$

{The proximal joint of a segment is the nearest joint to the wrist, while the distal joint is the other joint}

Calculate the "vertex thickness", which is the distance between the vertex and its projection on the segment.

Fig. 4 shows the principle.

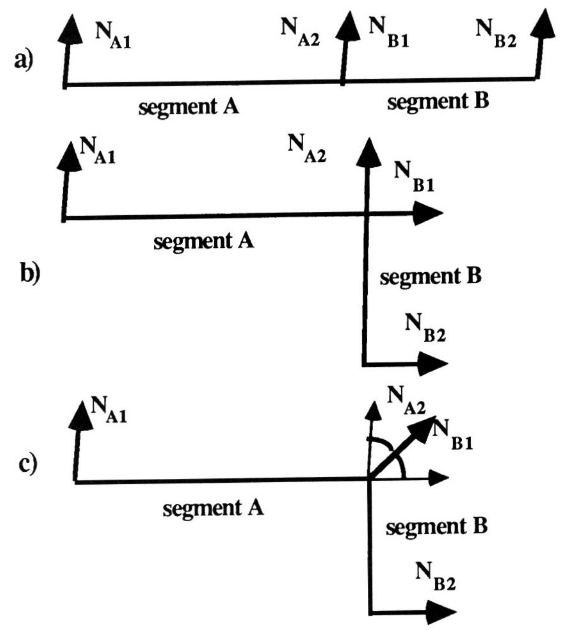

Fig.3 Normal calculations. a) Original normals, initial position b) Original normals, final position (flexion angle=0) c) Modified normals, final position

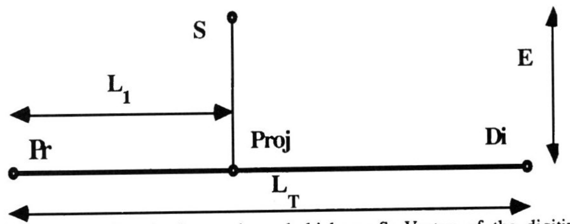

Fig.4 Concepts of projection, ratio and thickness.S: Vertex of the digitized character; Pr: Proximal segment joint; Di: Distal segment joint; Proj: Projection of S onto the segment; E: Thickness of S (distance from S to Proj); L_1 : distance from Pr to Proj; L_T: distance from Pr to Di; Ratio = $\frac{L_1}{L_T}$

A projection is also determined on the segment in its final position by calculating R (PF.D - PF.P) + PF.P where R is the above ratio, PF.P is the final position of the proximal joint and PF.D the final position of the distal joint. If the segment to be covered is a segment of the palm, same projection calculations are performed, but relatively to the neighbor segment, in order to obtain an initial projection and a final projection on this segment . These projections are used to simulate a virtual segment (see Fig.5 and Section 5) linking the vertex projection on the segment to be covered to the vertex projection on the neighbor segment. The projection of the final position is also used as reference point or position relative to the segment, in order to allow the transformation from the initial position to the final position.

This virtual segment allows the calculation of a scale factor:

$$F = \frac{\text{length of the virtual segment at the initial position}}{\text{length of the virtual segment at the final position}}$$

If the neighbor segment in the final position is further from the segment to be covered than in the initial position, the scale factor F will be greater than 1 otherwise it is in the range [0,1[. In this latter case, the distance between both segments has decreased relatively to the initial position and an inflation should be generated.

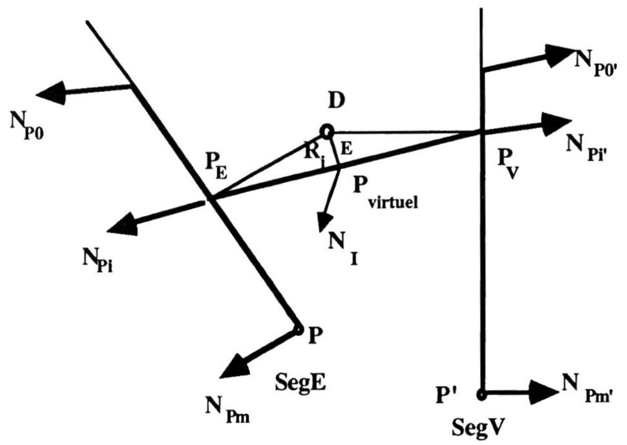

Fig.5 Normal interpolation for an inbetween area. N_{P_o} and N_{P_m}: original and modified normals for the area of the proximal joint of the segment to cover; $N_{P_o'}$ and $N_{P_m'}$: original and modified normals for the area of the proximal joint of the neighbor segment; N_{P_i} and $N_{P_i'}$: interpolated normals for each of both areas of the proximal joint; R_i: inbetwen ratio (projection onto the virtual segment); interpolated normal = $N_i = R_i \, N_{P_i'} + (1-R_i) \, N_{P_i}$

5. CALCULATION OF THE COORDINATE BASES OF A VERTEX IN 3-SPACE

The problem may be separated into two cases: single segment covering for the fingers and two-segment covering for the palm. We also separate the processing of external vertices (the upper side of the hand) and the processing of internal vertices (lower side of the hand). This separation is determined using the cosine of the angle between the direction of the vertex relative to the projection onto the segment and the direction of the normal used as y-axis.

5.1 Single segment covering

In the case of external vertices, the segment is divided into three areas, using the parameters given by the animator (see Fig.6 and Section 3). A different coordinate basis is calculated for each area, because the simulation of joint rounding may be very different for each joint, due to the type of flexion. The normals at each joint are used as Y-axes of the coordinate bases (the positive direction is towards the upper part of the hand) ; the middle area (area 3) is a buffer and uses an interpolated normal between normals of the areas 1 and 2.

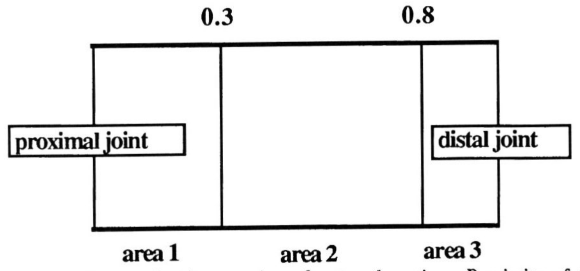

Fig.6 Segment areas for the covering of external vertices. Proximity of the proximal joint =0.3; proximity of the distal joint = 0.2

In the case of internal vertices, the muscle is inflated along the whole segment length.

5.2 Two-segment covering

In the case of external vertices, three areas are specified (see Fig.7). Areas along each of both segments (areas 1, 4, 7 and areas 6, 3, 9) are determined as in the case of single segments using animator-defined parameters and they are processed as above.

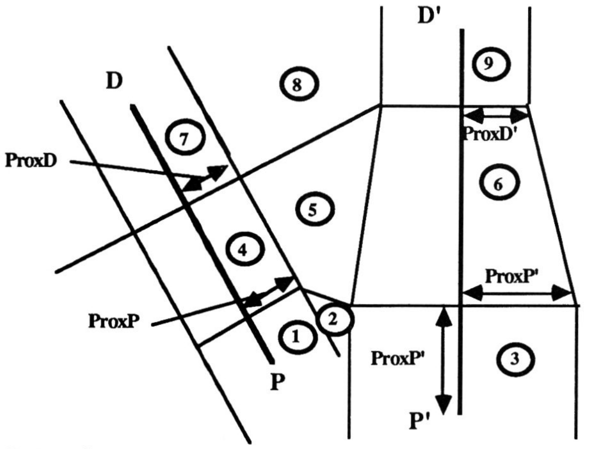

Fig.7. Areas for a two-segment mapping. P: proximal joint of the segment to cover; D: distal joint of the segment to cover; P': proximal joint of the neighbor segment; D': distal joint of the neighbor segment; ProxP: proximity parameter of P; ProxD: proximity parameter of D; ProxP': proximity parameter of P'; ProxD': proximity parameter of D'

However, these parameters are also used to determine areas between the segments, as shown in Fig.7 (areas 2, 5, 8). For these inbetween areas, the calculation of the bases is different, because there are no roundings to be calculated. In the case of internal vertices, we have only three areas: one for the segment to be covered, one for the neighbor segment and one inbetween area. No internal inflation has to be calculated.

5.3 Determination of the bases for external vertices

For areas with one joint (areas 1 and 2 of Fig.5 and areas 1,3,7,9 of Fig.7), we use as x-axis, the direction of the segment in space, and as y-axis the direction of an interpolated normal between the original normal and the modified normal; this interpolation is carried out according to the relative position of the vertex projection onto the segment inside the area (see Fig.8). The z-axis is obtained as the cross product of the x-axis and the y-axis.

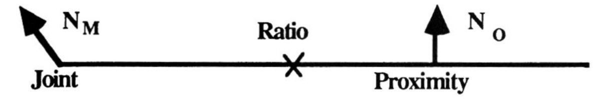

Fig.8. Normal interpolation for an area containing one segment. N_o = original normal, N_m= modified normal; the interpolated normal is computed as: $\frac{\text{Ratio}}{\text{Proximity}} N_o + (1- \frac{\text{Ratio}}{\text{Proximity}}) N_m$

For the buffer area of a segment (area 3 of Fig. 6 and areas 4 and 6 of Fig. 7), the x-axis is also the direction of the segment, but for the y-axis, the normal is interpolated between the original normal of the proximal joint and the original normal of the distal joint. The relative position is calculated according to the position of the vertex projection between the two limits of the buffer area, as shown in Fig.9.

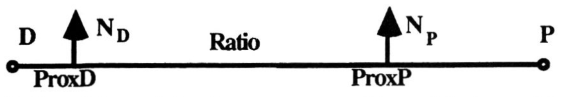

Fig.9. Normal interpolation for a tampon area.Np = original normal for the proximal joint,N_D = original normal for the distal joint; the interpolated normal is computed as: R' N_D + (1-R') Np with R' = Ratio - $\frac{\text{ProxP}}{\text{1-ProxD}}$ - ProxP

Finally, for the inbetween areas (used only for two-segment covering), the method uses the virtual segment introduced in Section 4. This virtual segment allows the calculation of inbetween values:

- a projection of the vertex to be covered onto the virtual segment
- the ratio of this projection to the virtual segment, which means the distance between this projection and the projection of the vertex onto the segment to cover divided by the length of the virtual segment
- a new thickness value obtained from the distance between the vertex and its projection onto the virtual segment.

The direction of the virtual segment is used as x-axis. For the y-axis, the normal is interpolated at each normal limiting the virtual segment. These normals are themselves interpolated from the normals calculated in the areas along the segments, as shown in Fig.5.

5.4 Determination of the bases for external vertices

For inbetween areas, the method above for calculating the bases works well. For areas of segments, it is better to directly interpolate between the modified normals of each joint in order to avoid discontinuities in the internal inflations. The y-axis is calculated as shown in Fig.10, the y-axis and z-axis are calculated as above.

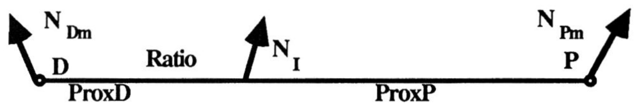

Fig.10. Normal interpolation for a tampon area. Np_m and Np_m= modified normals for the proximal and distal joints; the interpolated normal is computed as: N_I = Ratio N_{Dm} + (1-Ratio) Np_m

5.5 Vertex determination

Once the coordinate bases have been found for a vertex in the initial and the final position, the last step consists of representing the position of the vertex in the basis of the initial position in order to obtain a relative position in the space. To obtain the xyz-coordinates of the vertex in the basis, we have to represent in the initial basis, the vector obtained by the direction of the vertex in the space relative to its projection either onto the nearest segment, or onto the virtual segment , depending on the area in which it is located.

6. SIMULATION OF MUSCLE INFLATIONS AND JOINT ROUNDINGS

Coordinates may be then modified in order to simulate muscle inflations and the roundings of joints for improving the realism.

6.1 Joint roundings

The process of rounding the hand surface at the joints is only performed when the animator-defined parameter for external inflation is greater than zero. In this case, we have to modify the position in y and z of the external vertices located in the areas where there are joints. The modification is made using the following flexion ratio:

$$FR = \frac{\text{flexion angle given at the joint}}{\text{maximal angle possible for the joint}}$$

This means that the greater the flexion, the greater the inflation. The modification is also based on the relative position to the segment: the larger the ratio, the less the inflation. Moreover, when the direction of the vertex is further from the normal determining the y-axis, the z-coordinate increases. The amplitude of the inflation is directly proportional to the animator-defined parameter.

6.2 Muscle inflation

If the animator-parameter for muscle inflation is greater than zero, we have to modify the y-coordinate of the internal vertices located in the segment areas. The maximum inflation is directly proportional to the value of the internal inflation parameter and the above flexion ratio. The variation in y is defined by two exponential functions (a^b) joined together (exponential functions are easy to modify). The a value is determined according to the relative position of the vertex projection onto the segment (ratio) and according to the join point of both functions (this is an animator-defined parameter). The calculation of this value a depends on the the value of the ratio relative to the value of the join parameter. The calculation method is shown at Fig.11.

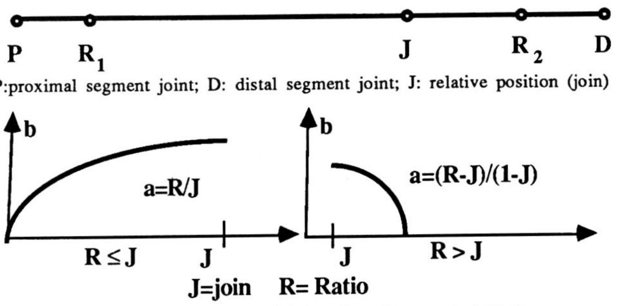
P
R_1
J
R_2
D
P:proximal segment joint; D: distal segment joint; J: relative position (join)
b
a=R/J
$R \leq J$
J
b
a=(R-J)/(1-J)
J
R > J
J=join R= Ratio

Fig.11 The use of exponential functions for muscle inflations

The b value for each exponential function has been semi-empirically defined in order to obtain curves as shown in Fig.11. It means that exponential functions should be adjusted for better results. Then the exponential functions are applied to the y-values, according to the ratio, in order to determine the new y-value after inflation. Fig. 12 and Fig. 13 show the appearance of a finger before and after application of the exponential functions.

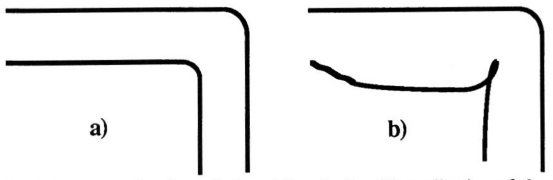

Fig.12 Aspect of a finger before (a) and after (b) application of the exponential functions

Once these modifications have been performed, the xyz-coordinates are converted into the final coordinate basis by using as origin the final projection as extrapolated at the beginning.

An example of animated sequence is shown in Fig.14 (skeleton) and in Fig.15 (shaded image).

6.3 Comments on the method

The major problem in the hand covering process is the calculation of the coordinate bases. We have tried to achieve both accuracy and flexibility. Our results are accurate, because the use of JLD operators allow the accurate simulation of joint roundings and the buffer areas avoid rough changes of coordinate bases, which could generate bad deformations. Also accuracy, because of the use of virtual segments in the inbetween areas avoids large variations in the orientations of the axes of the inbetween bases between two segments. This is specially important in the case of the thumb-index area where the segments of the palm are very mobile one relative to one other. The approach is flexible, because of the use of 9 areas for processing vertices according to their position relative to the segments, although a proximal assignment has been performed. This avoids assigning some vertices to the wrong segment. The key to our method was to find a model for calculating the bases which was sophisticated enough for the simulation of complex motions. It should be noted that discontinuities are avoided for external vertices, because of the inbetween area.

7. OBJECT GRASPING

Our approach to the problem of hand grasping may be considered as semi-automatic. Object manipulation consists of:

- handling a list of objects associated with a hand for a certain time and calculating the positions in 3D-space
- allowing the hand to be positioned on objects avoiding any intersection between the objects and the hand.

For any object of the list, we store the joint to be associated with the object and the initial and final time of association. The joint is used as a reference point in the 3-space or origin point of the basis used for the calculation of the positions of the object in the space. The initial time and final time indicate the period of association of the object with the joint. As for hand mapping, an initial basis is calculated for the starting position of the object and a final basis for the final position. As the origin of the bases is in fact the associated joint, the object will follow the motion of the joint. Axes of the bases are the flexion angle in x, a segment belonging to the joint in y, and the cross product in z. With such an approach, the object in the 3-space is oriented according to the flexions of the associated joint. For positioning the hand on an object, three processes exist:

1. The calculation of the distance between a hand vertex and an object. This consists of representing a hand vertex in a basis with an origin corresponding to a point belonging to a facet of the object (selected by the animator). The axes of the basis are a vector in the facet plane in X, the normal to the plane in y, and the cross product in z. Such a basis is used, because it gives a good idea of the position of a hand vertex with respect to the object. x- and z-coordinates indicate the lateral position and the y-coordinate indicates whether the hand and the object have a contact. To obtain this information, surface mapping has only to be calculated for the segment which contains the hand vertex for which the distance to the object has to be evaluated.
2. The intersection calculation between the hand and the object. This process uses the distance process, but for each vertex belonging to a particular segment specified by the user.
3. The semi-automatic calculation of the flexion angle is also based on the distance process. The user has to specify the contact point on the hand, the contact facet on the object and the joint for which the flexion angle has to be calculated. Instead of using an inverse kinematic process, which may be too time-consuming for an interactive editor, a dichotomous search is used for finding the angle α which corresponds to a distance DIST(α) such that $|DIST(\alpha)-DIST|<\varepsilon$ where ε is a threshold value. The hand mapping process is still used for the segment containing the contact point on the hand. It means that the animator should get the hand close enough to a correct grasp that the system can just bend one angle to make contact.

Collision between other hand parts are not taken into account. Fig.16 - 19 show frames of an animation sequence with the synthetic actress Marilyn Monroe.

8. SOFTWARE ENVIRONMENT

Hand animation and object grasping are part of the HUMAN FACTORY animation system. The main purpose of the HUMAN FACTORY system is the direction of synthetic actors in their environment. In this fourth generation system, synthetic actors are controlled by animators and designers without any programming knowledge. Not only has the user a high-level interface with menus and commands, but also he/she may add his/her own commands and menus to the system. The HUMAN FACTORY system is structured into four main modules:

- BODY-MOVING: a human body animation editor
- FACE-MOVING: a human face animation editor
- SABRINA: an object modelling and image synthesis system
- MIRANIM: an extensible director-oriented animation system

9. THE USER INTERFACE

Although object grasping may involve other joints, we shall limit our description to arm control from the shoulder. Arm joints which are concerned are: shoulders, elbows and wrists. For example, the command SHOULDER controls the three angles for the shoulders: flexion angle, pivot angle and twisting angle; its syntax is as follows:

```
SHOULDER   [LEFT/RIGHT][FLEXION/PIVOT/TWISTING]
           <key1><key2><step><value>
```

Similarly, we define:

```
ELBOW [LEFT/RIGHT][FLEXION/TWISTING]<key1><key2><step> <value>
WRIST [LEFT/RIGHT] [FLEXION/PIVOT]<key 1> <key 2> <step> <value>
```

For the hands themselves, we define metacarps joints (joints 1 and 2) for the palm animation and finger joints (joints 6 to 17). As shown in Fig.1, metacarpi are small bones linking joints 2 to 3 and 1 to 6. The flexion of metacarps may vary from 0 to 20 degrees. They can be controlled by the command:

```
METACARPUS [LEFT/RIGHT] <key 1> <key 2> <step> <value>
```

Fingers joints are separated into two groups: the five joints at the finger base (3, 6, 9, 12 and 15) and the two other joints for each finger (4,5,7,8,10,11,13,14,16 and 17). For the first category, two angles are defined for each joint: the flexion and pivot angles; the command is defined as:

```
FINGER1    [LEFT/RIGHT] [THUMB/INDEX/MIDDLE/RING/PINKY]
           [FLEXION/PIVOT] <key 1> <key 2> <step> <value>
```

For the second category, only a flexion angle is defined:

```
FINGER23  [LEFT/RIGHT] [THUMB/INDEX/MIDDLE/RING/PINKY]
          [KNUCKLES2/KNUCKLES3] <key 1> <key 2> <step> <value>
```

As shown in previous sections, there are two types of operations for hand grasping: object association and semi-automatic angle calculation. The OBJECTS command allows the association of an object with a hand joint for a given time. The command is defined as follows:

```
OBJECTS  <object identifier> [THUMB/INDEX/MIDDLE/RING/PINKY]
         [LEFT/RIGHT][KNUCKLES2/KNUCKLES3]
         <starting time><stopping time>
```

For example, "OBJECTS APPLE INDEX RIGHT KNUCKLES2 5 10" means that until time 5, the apple will be drawn at its intial position, then from time 5 to time 10, the apple will follow the motion of the second index joint. After time 10, the apple is considered as located at its final position. Three similar commands calculate, for a given key-frame, the flexion angles required to place a hand vertex in contact with the surface of an object. The animator specifies the vertex on the hand and the facet of the object. For example, the command AUTO_WRIST calculates the flexion angle for the wrist:

```
AUTO_WRIST  <keyframe number>  [LEFT/RIGHT] <hand vertex number>
            <object identifier> <object facet number>
```

Similarly the commands AUTO_F_1 and AUTO_F_2_3 calculates the flexion angles for the fingers.

10. CONCLUSION

This paper has described algorithms for animating the hand and grasping objects in the context of synthetic actors. These methods have proved efficient for producing computer-generated films. However, task-level issues need to be emphasized and robotics and A.I. techniques must be introduced to allow automatic object grasping. This implies the evaluation of the flexion angles of the joints from the current position of the hand and the current position of the object to be grasped. This problem is quite similar to a robotics problem[20], such as inverse kinematics. Moreover, the animation of the hand is dependent on the object shape— a plate is not grasped in the same way as a glass. This is A.I. type problem[21]. Future research at MIRALab will focus on these issues.

ACKNOWLEDGEMENTS

The authors would like to thank Lucie Marchand and Ross Racine who designed the images and Denis Rambaud for his collaboration. The research was supported by the Natural Sciences and Engineering Council of Canada, and the FCAR foundation.

REFERENCES

1 Hill DR, Pearce A, Wyvill B (1987) Animating Speech: an Automated Approach Using Speech Synthesised by Rules, The Visual Computer, Vol.3, No6

2 Lewis JP, Parke FI (1987) Automated Lip-synch and Speech Synthesis for Character Animation, Proc. CHI '87 and Graphics Interface '87, Toronto, pp.143-147.

3 Magnenat-Thalmann N, Primeau E, Thalmann D (1987) Abstract Muscle Action Procedures for Human Face Animation, The Visual Computer, Springer, Vol.3, No 6

4 Nahas M, Huitric H, Saintourens M (1987) Animation of a B-spline Figure, The Visual Computer, Vol.3, No4

5 Parke FI (1982) Parameterized Models for Facial Animation, IEEE Computer Graphics and Applications, Vol.2, No9, pp.61-68.

6 Pearce A, Wyvill B, Wyvill G, Hill D (1986) Speech and expression: a Computer Solution to Face Animation, Proc. Graphics Interface '86, pp.136-140.

7 Platt S, Badler N (1981) Animating Facial Expressions, Proc. SIGGRAPH '81, pp.245-252.

8 Waters K (1987) A Muscle Model for Animating Three-Dimensional Facial Expression, Proc. SIGGRAPH '87, Vol.21, No4, pp.17-24.

9 Armstrong WW, Green M (1985) The Dynamics of Articulated Rigid Bodies for Purposes of Animation, The Visual Computer, Vol.1, No4, pp.231-240.

10 Badler NI and Smoliar SW (1979) Digital Representation of Human Movement, ACM Computing Surveys, March issue, pp.19-38.

11 Badler NI; Korein JD, Korein JU, Radack GM, Brotman LS (1985) Positioning and Animating Figures in a Task-oriented Environment, The Visual Computer, Vol.1, No4, pp.212-220.

12 Kroyer B (1986) Animating with a Hierarchy, Seminar on Advanced Computer Animation SIGGRAPH '86.

13 Magnenat-Thalmann N, Thalmann D, Forest L (1986) An Integration of Keyframe and algorithmic animation, Proc. Computer Graphics Tokyo '86.

14 Magnenat-Thalmann N, Thalmann D, Forest L, Rambaud D (1986) Keyframe subactors, Proc. Graphics Interface '86.

15 Wilhelms JP, Barsky BA (1985) Using Dynamic Analysis to Animate Articulated Bodies such as Humans and Robots, Computer-generated Images, Springer Tokyo , pp.209-229.

16 Catmull E (1972) A System for Computed-generated movies, Proc. ACM Annual Conference, pp.422-431.

17 Badler NI Morris MA (1982) Modelling Flexible Articulated Objects, Proc. Computer Graphics '82, Online Conf., pp.305-314.

18 Kochanek D, Bartels R (1984) Interpolating Splines with Local Tension, Continuity and Bias Tension, Proc. SIGGRAPH '84, pp.33-41.

19 Magnenat-Thalmann N, Thalmann D (1987) The Direction of Synthetic Actors in the film Rendez-vous à Montréal, IEEE Computer Graphics and Applications, Vol. 7, No 12.

20 Korein JU (1985) A Geometric Investigation of Reach, MIT Press

21 Magnenat-Thalmann N, Thalmann D, Modeling and Animation of Synthetic Actors Using Artificial Intelligence, Prentice Hall, Englewood Cliffs (to appear in 1988)

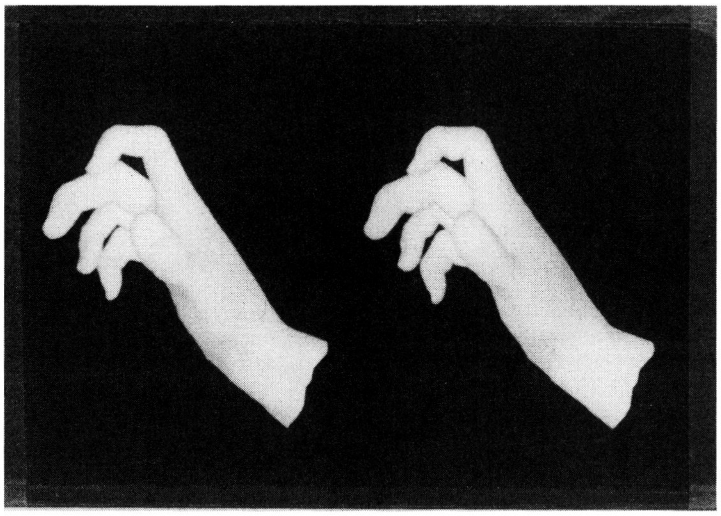

Fig.13 Appearance of a finger before and after application of exponential functions

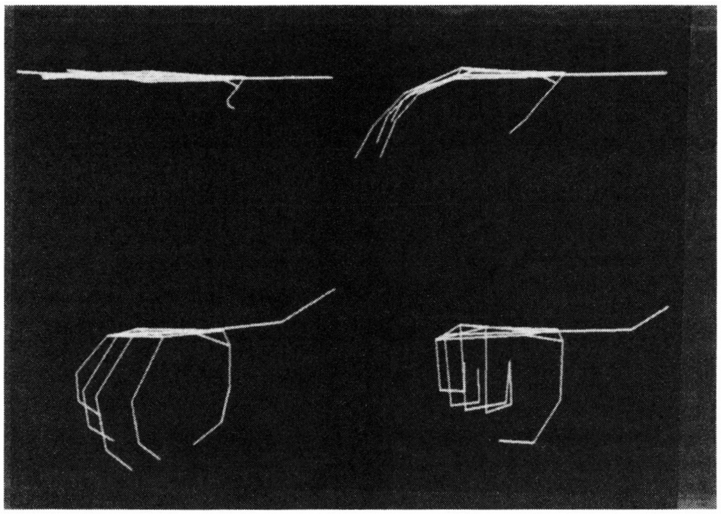

Fig.14 Animated sequence (skeleton)

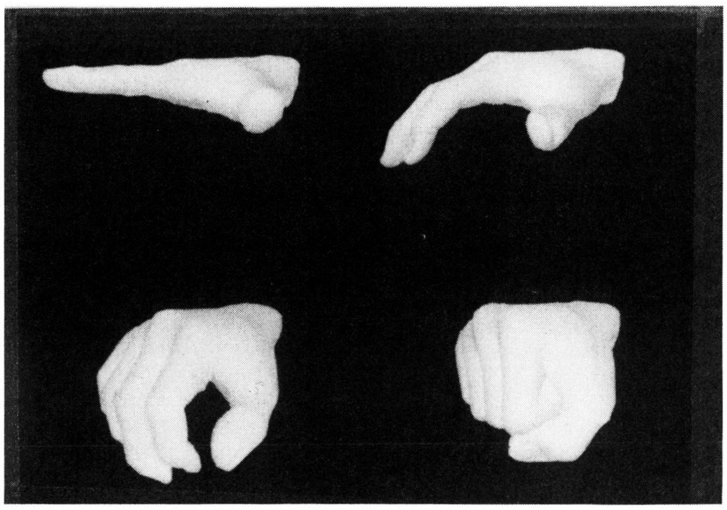

Fig.15 Animated sequence (shaded images)

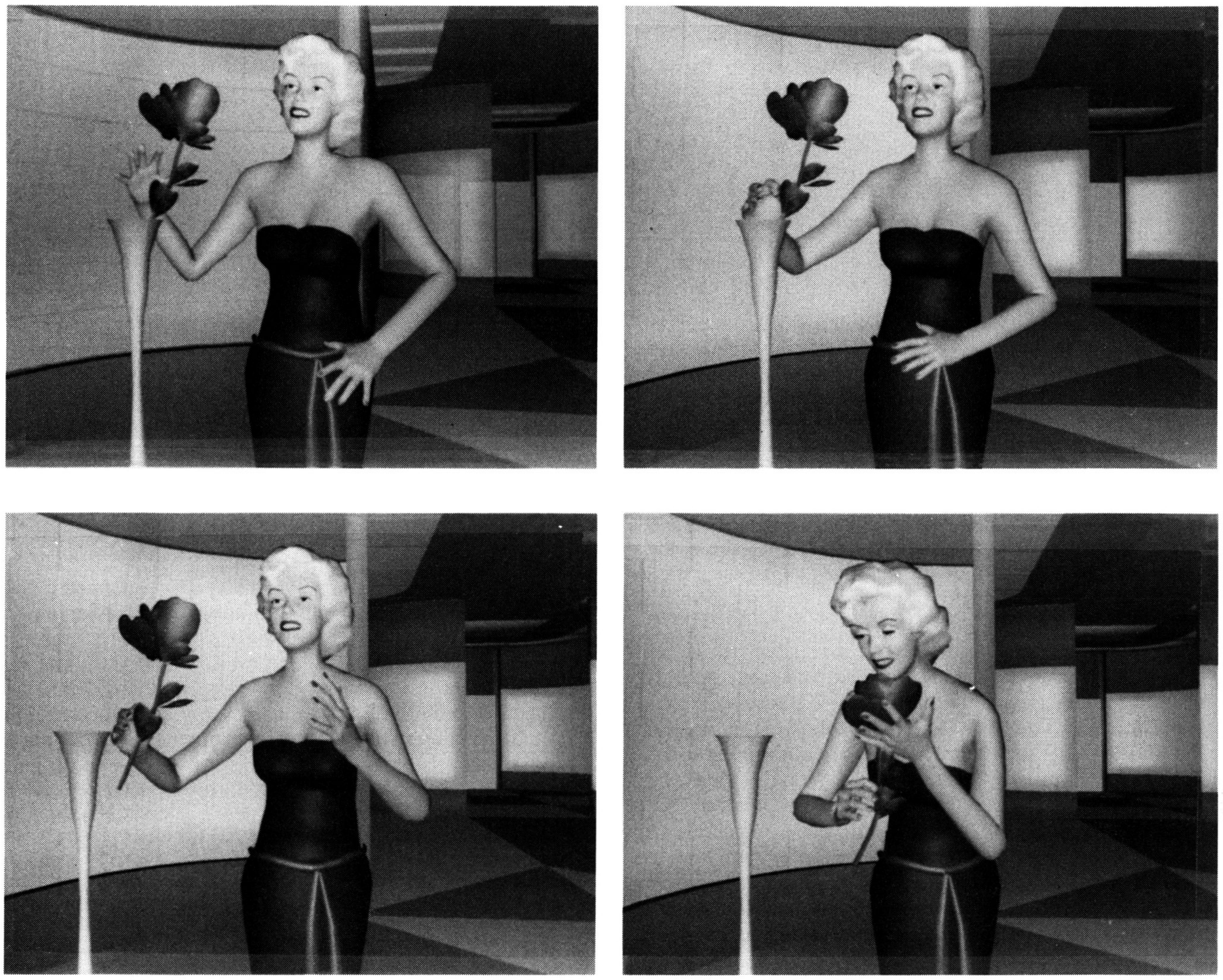

Fig.16-19 Animation sequence with object grasping