〔非公開〕

ＴＲ－Ｃ－００８９

Evaluation of the Homotopy Sweep Technique: Representation and Animation of three-dimensional images for human face generation and mouth animation

ムバラキ　ハッサン　Lhassan MOUBARAKI
田中　弘美　Hiromi TANAKA
北村　泰一　Yasuich KITAMURA

１９９３．４．３０

ＡＴＲ通信システム研究所

Evaluation of the Homotopy Sweep Technique Representation and Animation of three-dimensional images for human face generation and mouth animation.

Lhassan Moubaraki Hiromi T.Tanaka Yasuichi Kitamura

Contents

List of Figures

Acknowledgements

I would like to thank Mr R. Guedj , I.N.T Director for Scientific and Intenational Affairs who gave me the opportunity to accomplish my internship in Japan, Mr. N. Terashima - President of ATR Communication Systems Research Laboratories, and Mr. K. Habara - Executive Vice President of ATR International, Chairman of the Board of ATR Communication Systems Research Laboratories, for having invited me to come to ATR for five months.

I would like to express my sincere gratitude to Mr. F. Kishino - Head of the Artificial Intelligence Department, who welcomed me in his Department and enabled me to have a broad view of Japanese research on telecommunications thanks to the visit of NTT,NEC and KDD laboratories in Tokyo; and especially Mrs. H. Tanaka, my supervisor in ATR , who advised me and encouraged me for all my work.

I would like to thank Mr. M. Benjelloun, my scientific advisor in I.N.T for his encouragements, Mr. Y. Kitamura for his great help in the experiments, Mr. N. Ahuja for his encouragements and his advices, Mr. J. Ohya for his constant support in my research and Mr. Ochi who accepted to take part to my experiments.

Finally, I would like to thank Mr. K. Fujii, Senior Staff of the Planning Division, and all the members of the Planning Division and the Planning Section of ATR Communication Systems Research Laboratories for having organised so perfectly my stay in Japan.

1 Introduction

1.1 The subject of my internship.

The purpose of my internship was to evaluate an interpolation method based on the homotopy between two arcs and to apply it to the representation and description of three dimensional data taken from images in both a static and a dynamic approach. My work is related to the field of human representation in the ATR Artificial Intelligence Department and fits into the global project of the new teleconferencing system with realistic sensations in a real time process

.

This real time process requires a special processing because real time computer data representation and animation are limited by computers' and networks' capabilities. Eventhough many objects may be constructed by assembling geometric shapes like cubes, spheres, or pyramids, most objects have a free-form shape like human faces. One can use the coordinates of the very high number of points given by a 3-d digitizer but this is inconvenient in the case of a computer finite storage at the prospect of a data transmission for a real time animation. We need methods for developping free-form surfaces easily from a limited set of data and a model to allow people to visualize and understand the structure of the modeled entity . Furthermore, this model should provide a convenient vehicle for experimentation with.

The Homotopy Sweep Method was proposed in 1991 for surface generation using a set of two-dimensional contours with the interesting ability to control the transition from one contour to the other. During the five months of my internship in ATR, I was given the task to evaluate the Homotopy Sweep Technique for human face representation and to implement it on Silicon Graphics Workstations using three-dimensional data given by a digitizer. At the same time, we were thinking about a generalization of this method and , in order to take advantage of the convenient control of the deformation, I developped another interpolation formulation based on a dynamic approach. With some assumptions concerning the muscles and the human morphology, I applied this dynamic approach to mouth animation using a Lip Tracking Process and data given by a three-dimensional digitizer. With this method, we can control mouth continuous deformation just by specifying a few parameters. The animation results (recorded on VHS tape) were very realistic and the simple interpolation computations can be considered as a real time process.

In this report, I introduced the Homotopy Sweep Technique and the differents problems I had to solve for its application in the two approaches. Section 2 contains the basic principles , notions and definitions related to the Homotopy Sweep Technique. The static approach and the assumptions of our implementation are developped in Section 3. Section 4 and the VHS tape show the differents aspects of the dynamic approach for the human mouth animation and the powerful ability to control the deformation.

1.2 ATR presentation

ATR is a newly founded group (1986) whose main purpose is basic research in the telecommunications field. Its mainframe is five independent laboratories working in different areas.The total research and development funds given to the five laboratories each year amount to approximately 9 billion yens with 70 % coming from the Key technology center and 30 % coming from a set of private companies (NTT, KDD, NHK, NEC, Matsushita, etc). The total numbers of researchers amounts 300 in 1992.

The five laboratories are :

ATR Communication Systems Research Laboratories (full sensory communications,non linguistic communications, high security networks, automated development of communications software) working on a human-oriented intelligent communication system,

ATR Interpreting Telephony Research Laboratories (speech recognition, machine translation, speech synthesis) developing an interpreting telephone,

ATR Auditory and Visual Perception Research Laboratories (mechanisms of perception and cognition in the human senses of sight and hearing) seeking the ideal human-machine interface,

ATR Optical and Radio Communications Research Laboratories (communication devices based on artificially modulated material structure) working on a network covering from space to the individuals

ATR Human Information Processing Research Laboratories enhancing human-machine communication technologies.

2 The Integrated Homotopy Sweep Technique

2.1 The Generalized Cylinders

The generation of three-dmensional solid objects and more generally solid geometric modeling is very useful in computer-aided design (CAD) and computer-aided engineering (CAE) to create and communicate shape information.

Usually, in the field of solid representations, the most suitable model depends heavily on the domain. In CAD applications, general sweeps are a quite popular representation in computer vision , where they go by the name generalized cylinders and sweeping is frequently used for object modelling : the volume of the object can be described as a "swept volume" of a two-dimensional set moved along some three-dimensional space curve called the trajectory.

The simplest sweeping is the translational sweep: a solid can be represented by a two-dimensional set translated along a space vector. A rotational sweep is similarly defined by rotating the two-dimensional set around an axis ((Figure 1). A general sweep is therefore a combination of a translational and a rotational sweep of a two-dimensional set (or volume in the general case) which may vary parametrically along the axis space curve (as on Figure 1): a sweep surface is generated.

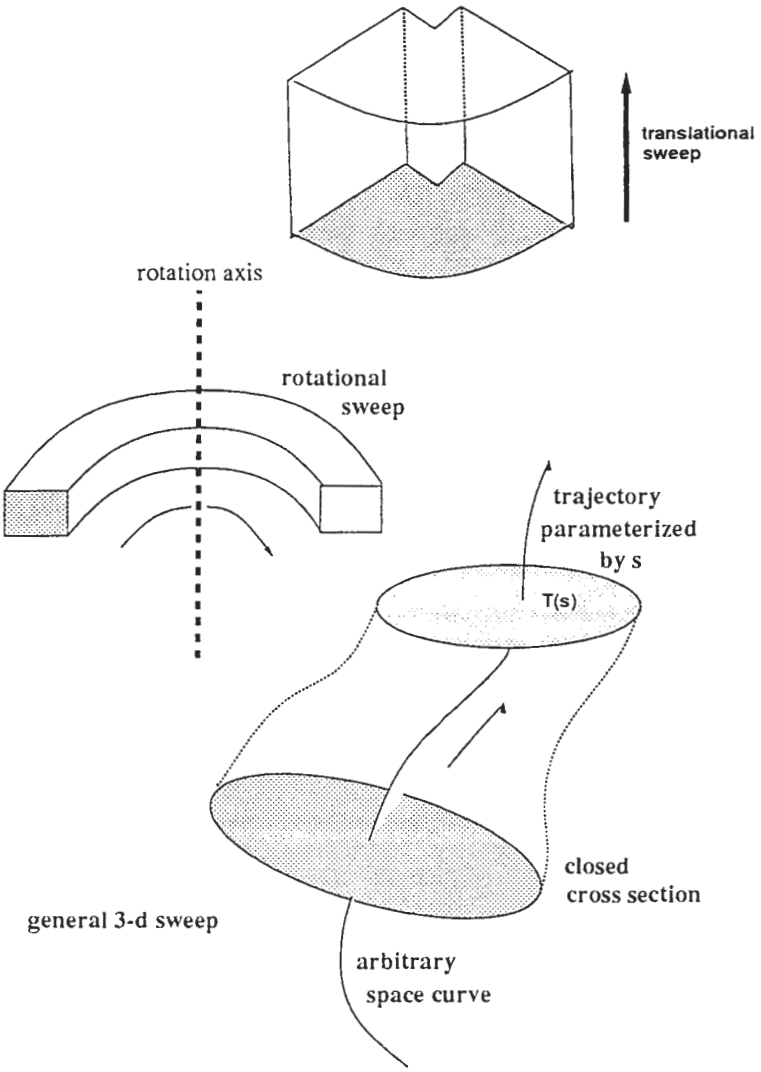


Figure 1: A translational, a rotational and a general sweep.

Then, a generalized cylinder is a solid whose axis is a space curve. At any point on the axis, a closed cross section is defined and called a contour. Usually, it is easy to think of an axis space curve and a cross section point set function , both parametrized by arc length along the axis curve. The usual restriction is that the axis be normal to the cross section. A generalized cylinder may have a varying cross section along the axis space curve .

Let T be a space curve parameterized by a parameter v, $v_i \leq v \leq v_f$. The coordinates of $T(v)$ in a (O,X,Y,X) coordinate system are given by:

$T(v) = [T_x(v), T_y(v), T_z(v)]$.

Two mathematical functions representing axis T and cross section for each point $T(v)$ define a unique solid. At each point T(v) of the axis, we have to choose a local coordinate system whose origin is attached to T(v): the cross section is then defined in this coordinate system.

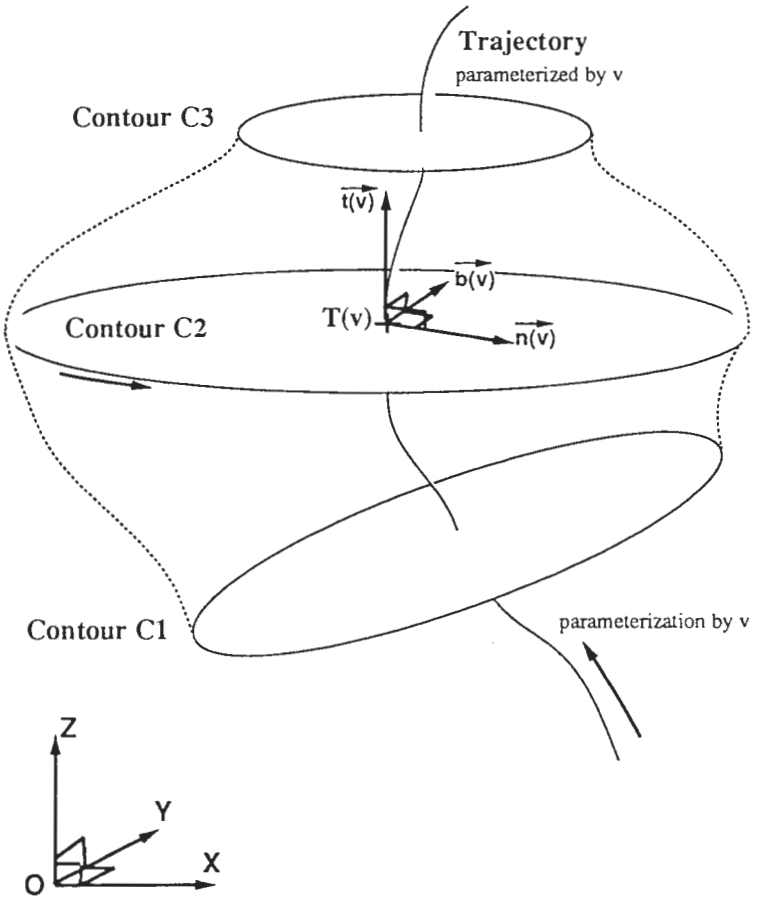


Figure 2: Generalized Cylinders and the Frenet Frame.

At each point , the Frenet frame gives good information about the axis of the generalized cylinder by using the following unit vectors:

$\vec{t}$: unit tangent vector at $T(v)$,

$\vec{t}(v) = \frac{\vec{T}'(v)}{\|\vec{T}'(v)\|}$

$\vec{b}$: unit binormal vector $\vec{b}(v) = \frac{\vec{T}'(v) \times \vec{T}''(v)}{\|\vec{T}'(v) \times \vec{T}''(v)\|}$

n: unit normal vector $n(v) = b(v) \times t(v)$

where $\vec{T}'(v)$ and $\vec{T}''(v)$ are respectively the first and second derivatives of $T(v)$.

However, the Frenet frame is not well defined when the curvature of the axis is zero (in that case, an adapted Frenet frame was defined by Bronsvoort and Klok 1985 : $\vec{t}(v)$ is unchanged; $\vec{b}(v)$ is a chosen fixed unit normal to the plane of the trajectory and $\vec{n}$ is the vector product of b and t). Anyway, it gives us the ability to define cross sectional contours at any point of the trajectory $T(v)$.

Finally, it can be practical to use generalized cylinder representation for a solid. The axis curve presents no difficulties but a usable representation for the cross section set is often not so simple. In our case, the representation is explicitly given by the Cyberware three-dimensional digitizer.

2.2 The mathematical definition of Homotopy

Here is reviewed the mathematical concept of homotopy, beginning by a couple of definitions.

2.2.1 Definitions

Definition 1

Let $B \subset R^3$ *and let* $\alpha_0 : [0, L] \longmapsto B$ *be two arcs of* B,
$\alpha_1 : [0, L] \longmapsto B$
joining the points $p = \alpha_0(0) = \alpha_1(0)$
and $q = \alpha_0(L) = \alpha_1(L)$.

We say that α_0 *and* α_1 *are homotopic if there exists a continuous map* $H : [0, L] * [0, 1] \longmapsto B$
such that:1. $H(s, 0) = \alpha_0(s)$
$H(s, 1) = \alpha_1(s)$, $\forall s \in [0, L]$
2. $H(0, t) = p$
$H(L, t) = q$, $\forall t \in [0, 1]$.
The map H *is called a homotopy between* α_0 *and* α_1.

The homotopy is a family of arcs α_t, $t \in [0, 1]$, which constitutes a continuous deformation of α_0 to α_1 in such a way that the extremities p and q of the arcs α_t remain fixed during the deformation.

The following definition shows a closer analogy with our study .

Definition 2

Let f,g: $X \longmapsto Y$ *be maps*
where X *and* Y *are topological spaces.*
If there exists a map $\mathcal{F} : X * I \longmapsto Y$
such that: $\mathcal{F}(x,0)=f(x)$
$\mathcal{F}(x,1) = g(x)\ \forall x \in X$
(where I=[0,1])
then the map $\mathcal{F}$ *is called a homotopy from f to g.*

For instance,a straight line homotopy is given by : $\mathcal{F}(x,t) = (1-t)f(x) + tg(x)$ $\forall t \in [0,1]$ and $\forall x \in [xi, xf]$.

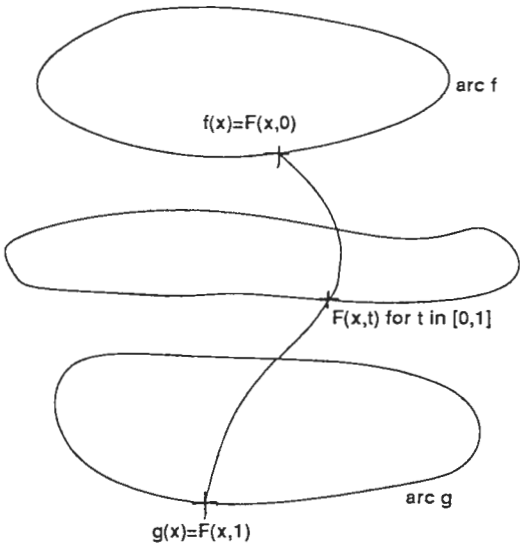


Figure 3: For a given x, a continuous deformation from f(x) to g(x)

As we can see in figure 3,if $f(x)$ and $g(x)$ denote two cross sectional contours (or, more generally two space curves) parameterized by the same parameter x , then the homotopy F defines a continuous deformation of contour f into contour g as t varies from 0 to 1 .

Therefore, F represents a bivariate parametric surface connecting the contours (which are space curves) .

2.3 The Homotopy Sweep Technique

2.3.1 A new approach

The necessity of representing complex surfaces needs appropriate means . A good representation of truly generalized cylinders is interesting in all respects because many complex shapes can be considered as a combination of generalized cylinders. Of course, such a description is not enough but it covers many cases of the real world and could be a practical way of three-dimensional data representation. Truly generalized cylinders can be obtain by defining a set of contours at some locations along a space curve (Shanna and Ballard 1984). Woodward (1986) proposed an approach based on a blending of the different contours by using the B-splines and orthogonal contours to the trajectory. But in this case, the user can not really control the resulting shape of the generated shape. This is an important point to take into account since we try to avoid using many contours for the data representation . Other methods have been proposed but, generally, the rate of transition of one contour to the other is not controlled. In 1991, Shinagawa an Kunii proposed the Homotopy Sweep Model for surface construction from a set of planar contours.

2.3.2 Integrated Homotopy Sweep Technique

Introduction. The Homotopy Sweep Technique which was proposed by Shinagawa and Kunii is an integration of the two notions defined above. The method was introduced as a model for surface construction by Chiew-Lan Tai, Kia-Fock Loe and Tosiyasu L. Kunii.

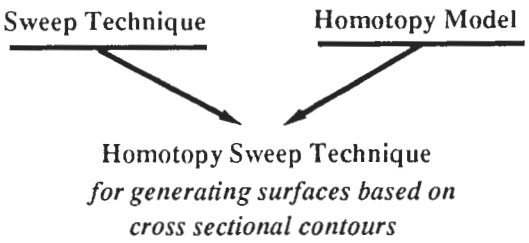


Figure 4: Integration of the two techniques.

A formulation of the Homotopy Sweep Technique. This technique is a method to reconstruct surfaces from cross-sectional data of objects using a homotopy to generate surfaces connecting consecutive contours. It provides a control of the shape of the cross section between two defined contours.

Here we introduce a formulation which takes into account the shape control between two defined contours.

Assumptions. Let T be a space curve parameterized by a parameter v, $v_i \leq v \leq v_f$. The coordinates of $T(v)$ in a (O,X,Y,X) cartesian coordinate system are given by:

$$T(v) = [T_x(v), T_y(v), T_z(v)]$$

Let C1 and C2 be two cross sectional contours parameterized by the same parameter u : $C1(u)$ and $C2(u)$,where $u_i \leq u \leq u_f$ (cf Figure 5.) (We will see how to make correspondance between contours which have different parameterizations.)

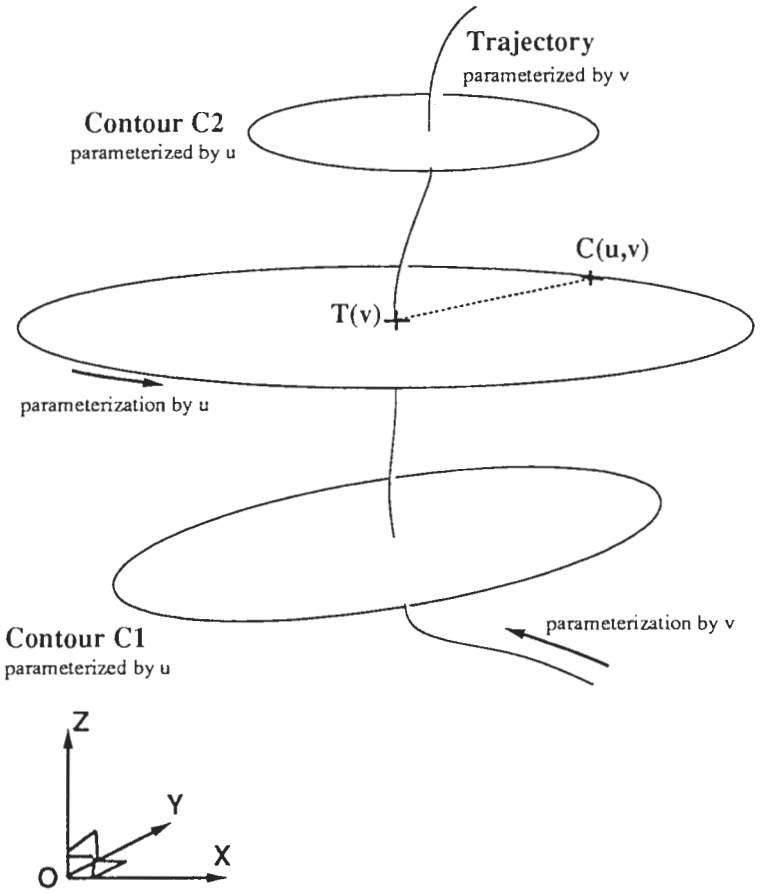


Figure 5: Homotopy Sweep Technique Parameterization.

For a given value v, a point $T(v)$ is defined on the trajectory; in the cross sectional plane (defined orthogonally to the tangent vector $t(v)$) , the planar

curve C representing the cross sectional contour is also supposed parameterized by u, $C(u)$, where $u_i \leq u \leq u_f$.

A point of the lower contour C1 is represented by : $C1(u)$.

A point of the upper contour C2 is represented by : $C2(u)$.

A point of the cross sectional contour corresponding to the point T(v) on the trajectory is represented by : $C(u, v)$.

Let us choose v in [0,1]: $C(u, 0)$ defines the point $C1(u)$ and $C(u, 1)$ defines the point $C2(u)$.

Objective and interest of the technique. The purpose is to choose a suitable homotopy from C1 to C2 which gives us a convenient control of the shape of the inbetween reconstructed contours.

The interest of this technique is that we can choose freely the contours to use and the deformation is controlled by two functions :

a blending function which controls the smooth transition from one contour to the other,

a scaling function which shapes the outline of the sweep object.

Only one parameter controls the sweeping as well as the homotopic deformation of the cross-sectional shape.

An original formulation. As introduced above, $C(u_0, v_0)$ represents the point on the cross- sectional contour at $T(v_0)$ whose parameter on the curve is u_0. By using the Frenet frame (or the adapted Frenet frame) at each cross-sectional contour along the trajectory, each point $C(u, v)$ has two coordinates in the cross-sectional plane (which is also the plane passing through the point $T(v)$ and which is orthogonal to the vector $\vec{t}(v)$):

$C_n(u, v)$ and $C_b(u, v)$

such that :

$$T(v)\vec{C}(u, v) = C_n(u, v)\vec{n(v)} + C_b(u, v)\vec{b(v)}$$

or, more simply:

$$C(u, v) = T(v) + C_n(u, v)\vec{n(v)} + C_b(u, v)\vec{b(v)}$$

In the work of [1], the following form of homotopy has been proposed by analogy with the straight line homotopy:

for given values u and v ,

$C(u, v) = [(1 - R_n(v))C1(u) + R_n(v)C2(u)][I_2 + S(v)] \quad 0 \leq v \leq 1$

where $C1(u) = [C_{1n}(u), C_{1b}(u)]$

$C2(u) = [C_{2n}(u), C_{2b}(u)]$

$$S(v) = \begin{pmatrix} S_x(v) & 0 \\ 0 & S_y(v) \end{pmatrix}$$

and $C(u,v)$ represents the coordinates of the point $C(u,v)$ in the local Frenet frame.

The blending function $R_n(v)$ is a scalar-valued function which controls the transition from C1 to C2 thanks to a parameter n. $S(v)$ is the scaling function where S_x and S_y are two scalar-valued functions that scale a blended inbetween cross section in two perpendicular directions.

The boundary conditions of the homotopy definition, i.e.

$C(u,0) = C1(u)$ and $C(u,1) = C2(u)$ $\forall u \in [u_i, u_f]$

implies the following conditions on the blending function R_n and the scaling function S:

$S(0) = S(1) = 0$

$R_n(0) = 0$

$R_n(1) = 1$

Since R_n blends the two contours $C1$ and $C2$ as v varies (by attributing a weight to each of the given contours in the inbetween contour), it may satisfy the condition $0 \leq R_n(v) \leq 1$ at any v values.

The reparametrization and linear trajectory hypotheses In order to simplify the technique , knowing that the data we have to manipulate are parallel contours given by the Cyberware digitizer, we assume a reparameterization is performed such that the correspondance between the parameters of C1 and C2 is simply chosen as follows:

$T(1)\vec{C}2(u)$ and $T(0)\vec{C}1(u)$ are, for all u , collinear vectors (cf. Figure 6).

Since we have chosen a linear trajectory, we will use indifferently the notation (x,y) for the axes or (n,b) for the same axes; moreover, we can choose at any moment a couple of (x,y) axes for the representation.

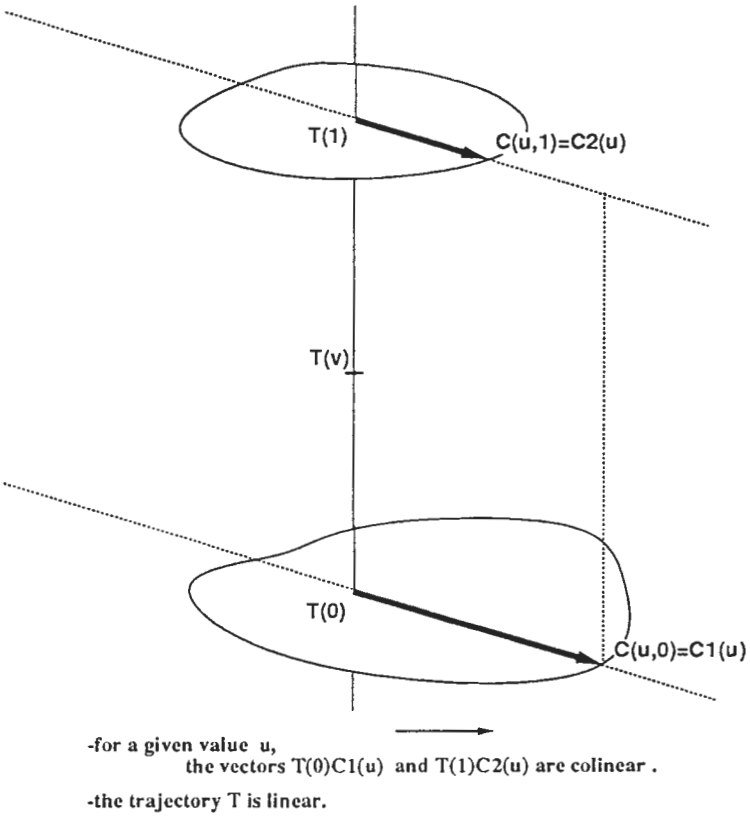


Figure 6: Assumptions on the trajectory and the correspondances between parameters of planar contours.

The choice of the blending function This is a suitable rational polynomial function proposed by (Tai):

$$R_n(v) = \frac{(1+n)v}{1+nv} \quad 0 \leq v \leq 1$$

and

$$-1 \leq n$$

Figure 7 shows the graph of R_n for different values of n. The lowest curve corresponds to $n = -0.9$.

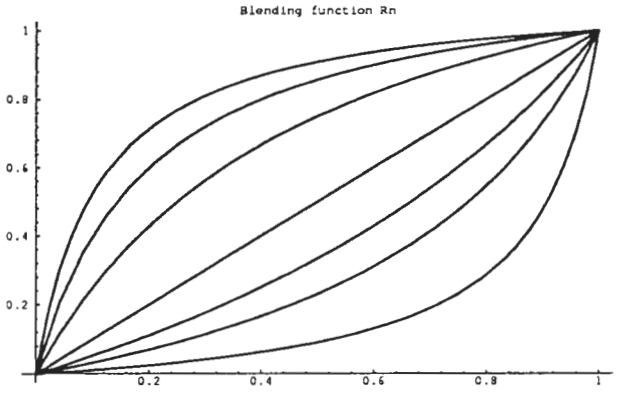


Figure 7: First example for R_n at n=-0.9, -0.7, -0.5, 0, 2, 5 and 9.

As we will see further, another form for the blending function is the following which has null derivatives at v=0 and v=1:

$$\begin{aligned} R_n(v) &= \frac{(1+n)v^2}{(1+n)v^2+(1-v)^2} \\ & 0 \leq v \leq 1 \\ & -1 \leq n \end{aligned} \tag{1}$$

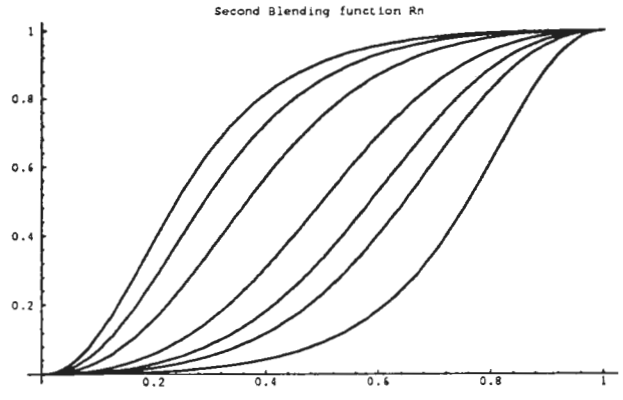


Figure 8: Second example for R_n at n=-0.9, -0.7, -0.5, 0, 2, 5 and 9.

The lowest curve corresponds to $n = -0.9$.

By using this second blending function and the formulation for homotopy described above, we can obtain objects like in two following figures with differents values of n:

Figure (9) : n=0;

Figure (10) : n=9;

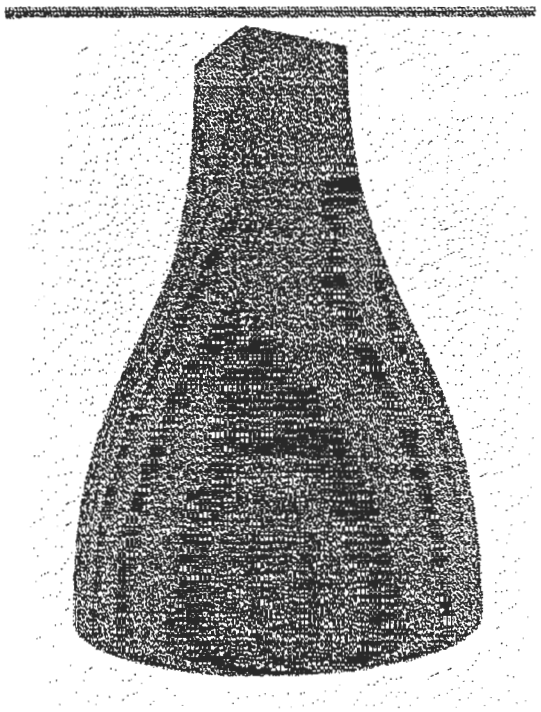


Figure 9: Object created by the homotopy technique with n=0 and Sx=Sy=0 (end contours: a square and a circle).

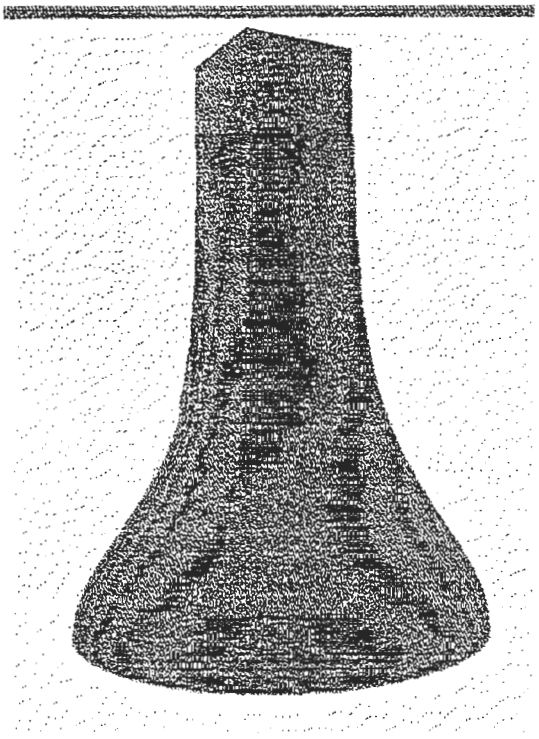


Figure 10: object created from the same end contours (square and circle) with n=9 and Sx=Sy=0.

The scaling functions have been affected the value 0. The end contours are circles : the smaller one is contour C2 .

We can interpret the two axes as components of the end contours. For instance, when we interpolate from the contour C1 at v=0 to the contour C2 at v=1 (C2 is the upper and smallest contour) , the value $R_n(v)$ at a given v is a blending between contour C1 and contour C2: the greatest the value $R_n(v)$, the greatest the component of contour C1, the smallest the component of contour C2 in the generated inbetween contour , namely the blended contour is closer than the contour C1 than contour C2.

Obviously, more complex contours could be used as inputs.

2.4 Continuity control at joint contours

In order to allow users to model arbitrary cross-sectional shape objects, it is necessary to envisage using more than two contours to reconstruct an object.

First,we begin by exposing the definition of the geometric continuity.

2.4.1 Geometric continuity

If two curve segments join together, the curve has G^0 geometric continuity. If the directions but not necessarily the magnitudes of the two segments' tangent vectors are equal at a join point, the curve has G^1 geometric continuity. In computer-aided design of objects , G^1 continuity between curves is often required. G^1 continuity means that the geometric slopes (tangent lines) are equal at the join point : there is a continuity in the tangent , but not necessarily neither in normal curvature as we can see in Figure 11 nor in the magnitude of the tangent vectors (Figure 12).

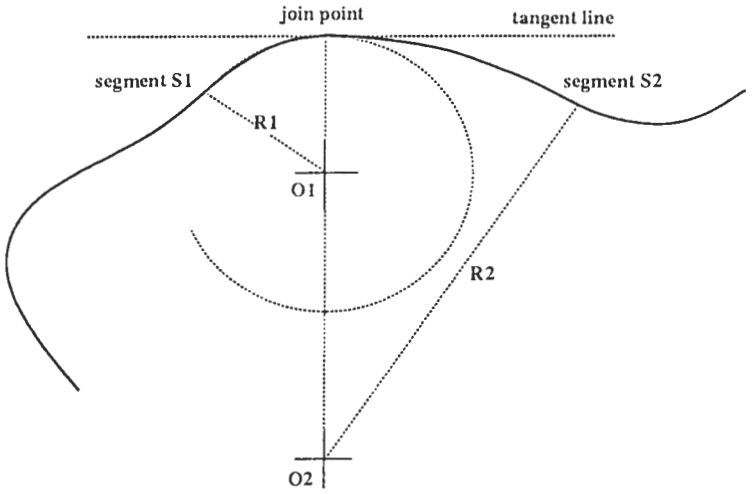


Figure 11: G_1 geometric continuity : the osculating circles.

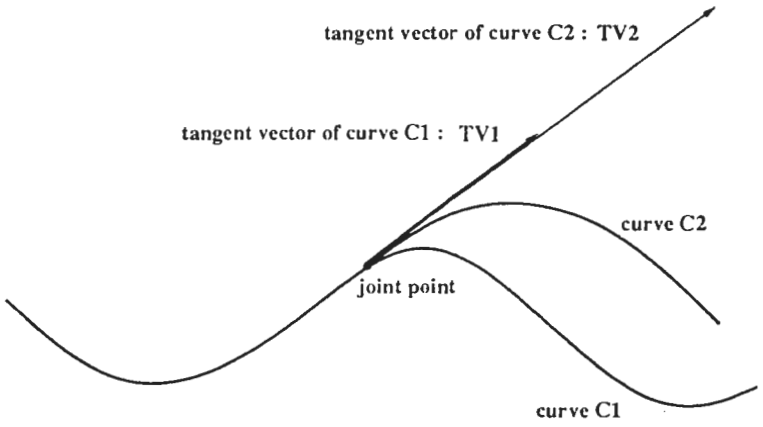


Figure 12: G_1 geometric continuity : the tangent vectors at the joint point.

The motivation for the definition of geometric continuity. Two curves that have identical plots can have differents tangents vectors (it depends on the parametrization of the curve) but still have the same tangent line at each point. We therefore restrict ourselves to the geometric continuity to have smoothness at a joint point between two curves.

2.4.2 Formulation of homotopy with G_1 geometric continuity.

Two contours allow us to construct by homotopy a surface. Let us think about having more given contours. Conditions of continuity appear therefore at the joint contour.

Let D_A be the generated surface between two contours $C1$ and $C2$, and D_B between the two other contours $C2$ and $C3$. We assume that the linear trajectory lies on the z-axis and the three contours are at the locations z_1, z_2 and z_3, respectively. We choose two other axes x and y such that (x,y,z) is an orthogonal coordinate system.

The constructed surfaces are given by:

$$D_A(u,v) = (C_x^A(u,v), C_y^A(u,v), (1-v) * z_1 + v * z_2)$$

$$D_B(u,v) = (C_x^B(u,v), C_y^B(u,v), (1-v) * z_2 + v * z_3)$$

$$0 \leq v \leq 1, \quad u_i \leq u \leq u_f$$

where:

$$\begin{pmatrix} C_x^A(u,v) \\ C_y^A(u,v) \end{pmatrix} = \begin{pmatrix} (1+S_x^A(v))(1-R_n(v))C_{1x}(u) \\ (1+S_y^A(v))(1-R_n(v))C_{1y}(u) \end{pmatrix} + \begin{pmatrix} (1+S_x^A(v))R_n(v)C_{2x}(u) \\ (1+S_y^A(v))R_n(v)C_{2y}(u) \end{pmatrix}$$

$$\begin{pmatrix} C_x^B(u,v) \\ C_y^B(u,v) \end{pmatrix} = \begin{pmatrix} (1+S_x^B(v))(1-R_m(v))C_{1x}(u) \\ (1+S_y^B(v))(1-R_m(v))C_{1y}(u) \end{pmatrix} + \begin{pmatrix} (1+S_x^B(v))R_m(v)C_{2x}(u) \\ (1+S_y^B(v))R_m(v)C_{2y}(u) \end{pmatrix}$$

For the values $v = 0$ and $v = 1$, the derivative vector relative to u is clearly in the plane of the contour going through $T(v)$. With respect to the definition of geometric continuity, having the same direction for the cross boundary derivative vectors (derivative vectors relative to v for a given u_0) is the necessary condition for achieving G^1 geometric continuity at the boundary between the two generated surfaces D^A and D^B.It is also a condition of tangent plane continuity at the boundary.

Figure 13 shows (in the plane P_u^{AB} which is defined by the linear trajectory and the parameter u) a case on non-G_1 continuity at the boundary of the two generated surfaces.

G^1 geometric continuity is achieved if we have colinear cross-boundary vectors at every point of the boundary; that is:

$$\frac{\partial D^A(u,v)}{\partial v}|_{v=1} = k\frac{\partial D^B(u,v)}{\partial v}|_{v=0} \qquad \forall u \in [u_i, u_f]. \tag{2}$$

where k is a shape parameter called the bias at the joint.

This condition has a very simple formulation.

As written in (Tai), the sufficient conditions for achieving G_1 continuity are:

$$\frac{\partial R_n^A(v)}{\partial v}|_{v=1} = 0 \tag{3}$$

$$\frac{\partial R_n^A(v)}{\partial v}|_{v=0} = 0 \tag{4}$$

$$\frac{\partial S_x^A(v)}{\partial v}|_{v=1} = k * \frac{\partial S_x^B(v)}{\partial v}|_{v=0} \tag{5}$$

$$\frac{\partial S_y^A(v)}{\partial v}|_{v=1} = k * \frac{\partial S_y^B(v)}{\partial v}|_{v=0} \tag{6}$$

$$z_2 - z_1 = k(z_3 - z_2) \forall u \in [u_i, u_f] \tag{7}$$

If we do not care of the continuity at the joint contour between two generated surfaces, we may create an object like in Figure (13) where the surfaces are generated separately with two different scaling functions and the same parameter n=0 (using the second form blending function). The lowest contour is a square, the middle one is a circle and the third one is a wavy contour.

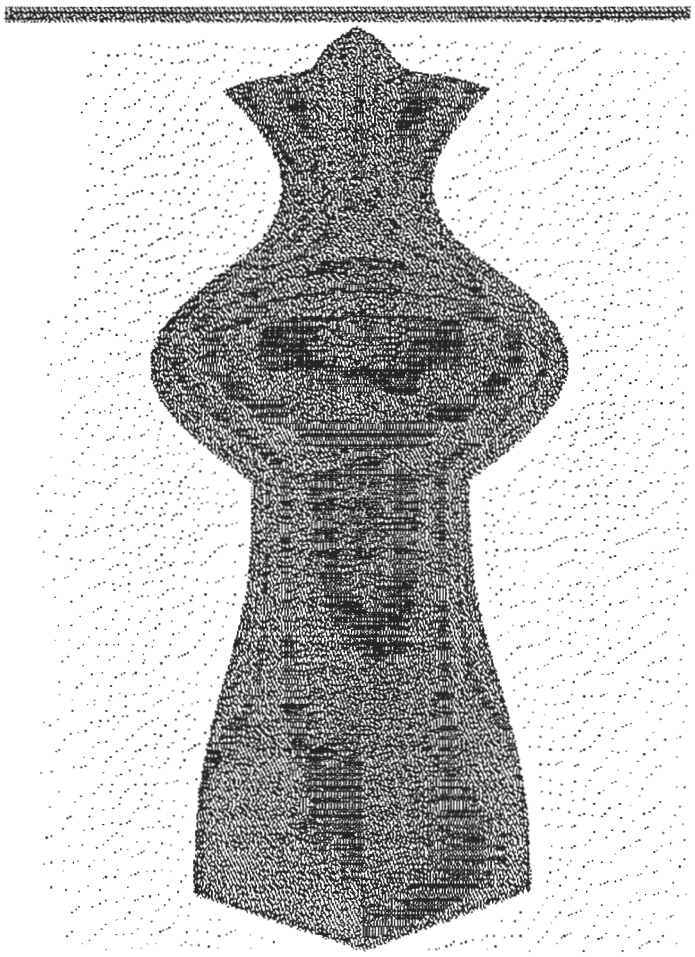


Figure 13: A generated object without G_1-continuity at the joint contour from a wavy contour, a circle and a square (with $S_x = S_y = 0$) .

These conditions at the limits can be satisfied easily if we choose the following blending function which verifies (3) and (4):

$$R_n(v) = \frac{(1+n)v^2}{(1+n)v^2 + (1-v)^2} \qquad (8)$$
$$0 \leq v \leq 1$$
$$-1 \leq n$$

This function still allows us to control the transition between the two con-

tours thanks to parameter n.

By calculating the first derivatives at the boundaries, we have the following tangent vectors:

$$\frac{\partial D^A}{\partial v}|_{v=1} = (C_2x\frac{dS_x^A}{dv}|_{v=1}, C_2y\frac{dS_y^A}{dv}|_{v=1}, z_2 - z_1)$$

$$\frac{\partial D^B}{\partial v}|_{v=0} = (C_2x\frac{dS_x^B}{dv}|_{v=0}, C_2y\frac{dS_y^B}{dv}|_{v=0}, z_3 - z_2)$$

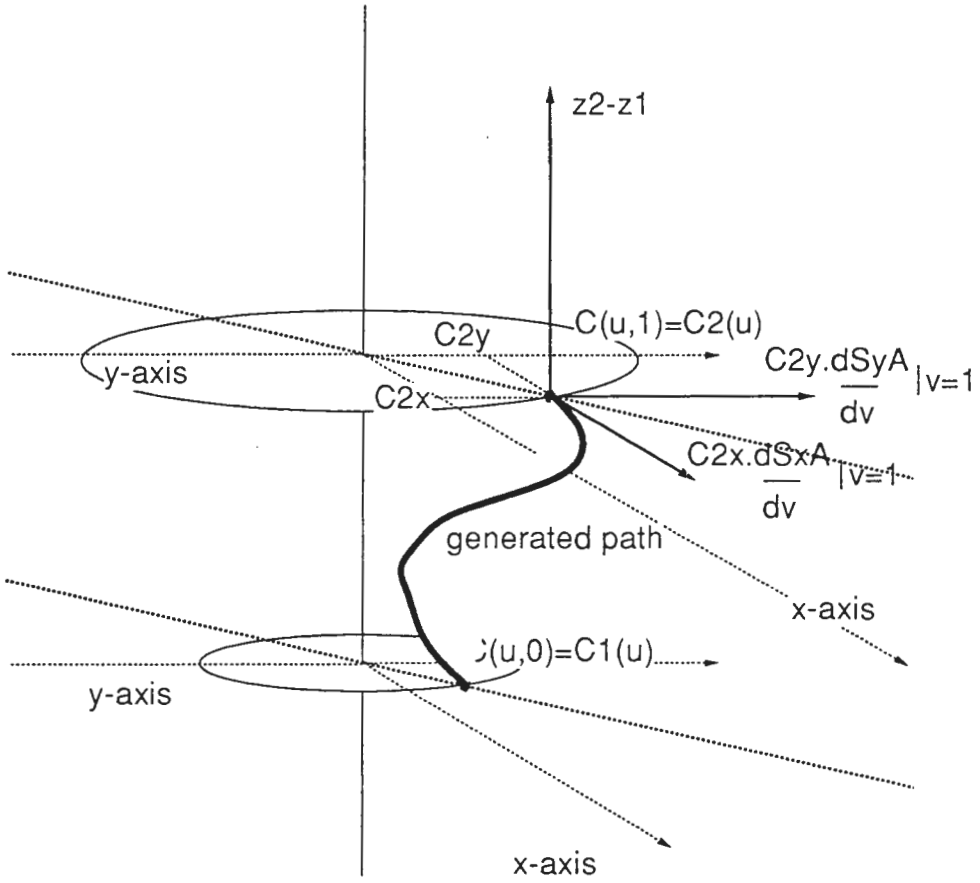


Figure 14: Generated Path from Homotopy Sweep Technique.

3 The Static Approach

3.1 The assumptions in our implementation

3.1.1 Caracteristics of the scaling and blending functions

The two scaling functions Sx and Sy allow us to choose a path from M1 to M2, thanks to the transitions which are different along x-axis and y-axis. At the same time , the blending function Rn gives an intuitive way to control the transition : at a given value v0, we can choose a transition which remains close to the contour C1 rather than to the contour C2 (and vice versa).

We can see (Figure (9) and (10)) that the curve of the blending function Rn shows the component of contour C1 or contour C2 when the trajectory parameter v changes . By specifying a parameter value n, we can obtain a curve for Rn which remains close to one of the two contours.

The method. For an implementation of the homotopy method, we must select a path from M1 to M2 on the real object to be reconstructed. By using the scaling function S and the blending function, we obtain an infinite set of possible paths from M1 to M2. The problem is therefore to give the path selection procedure from M1 to M2 and the criterions to take into account to this end.

We could choose to follow a path which goes through interesting points of the human face (like some features points). But to select feature points is itself a complex and difficult problem and needs appropriate methods. Generally , the feature points are taken manually by some human operator to obtain a wireframe of human face. In image synthesis, the ability to follow a chosen path by using a scaling funct ion S is powerful to create some unreal object or to deform a real one.

However, in the representation of a real object, we are faced to the problem of the extraction of significant points from the original object and their addition to the model we use. This means that in our case, the expression of the scaling function must be specified for each pair of contours according to interesting feature points i.e a path must be chosen for each pair of contours. A method for path search is needed, and some spline technique could be used to join the known values of the scaling function at the key points which corresponds to particular feature points or to specific paths.

Figure (15) shows an example of object obtained by a homotopy sweep interpolation in which Sy=0 and Sx=1: the effect of the scaling function is to deform the contours independently in the x-direction and the y-direction.

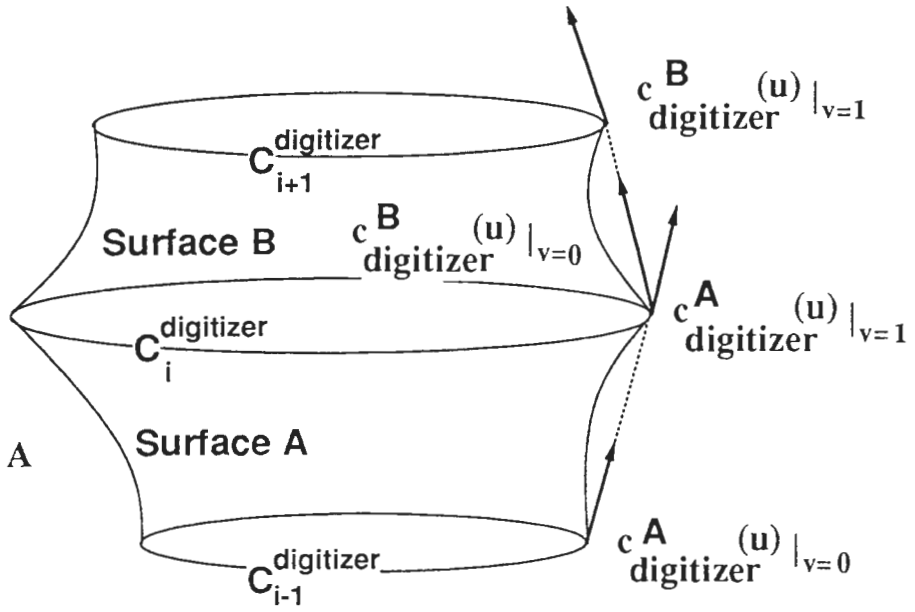


Figure 15: Effect of the Scaling Function.

For a human face representation (i.e an existing object to represent) , the problem is more arduous seeing that there is no particular axis for the interpolation (one could argue that there is a symetry axis in human faces and that we could use the homotopic deformation for a special interpolation in the nose direction; but not all the points can be generated by using such a deformation in the direction of the nose: for instance , the cheeks do not have this same orientation ! We must bear in mind that the deformation is the same for all values of u between two given contours)

As we do not make use of any feature point search method, or any particular axis for the interpolation, we have implemented a particular case of homotopy sweep interpolation technique: it is the case when the path joining the points on the two contours is located in the plane defined below.

Let us simplify our representation by assuming that the path from M1 to M2 is in the plane Pu=(O,M1,z)=(O,M2,z) In other words, there is no particular direction: contrary to figure (figure de l objet bizarre) the deformation is the same in all directions.

$$S(v) = Sx(v) = Sy(v)$$

$$\forall v$$

That is:

$$C_x(u,v) = (1+S)[(1-R_n)C_{1x} + R_n C_{2x}$$

$$C_y(u,v) = (1+S)[(1-R_n)C_{1y} + R_n C_{2y}$$

Since these vectors are in the plane P_u^{AB}, it is more suitable to use a cylindrical coordinate system (u,r,z) in which the z-axis remains the same as before (For more details , see Figure 16.

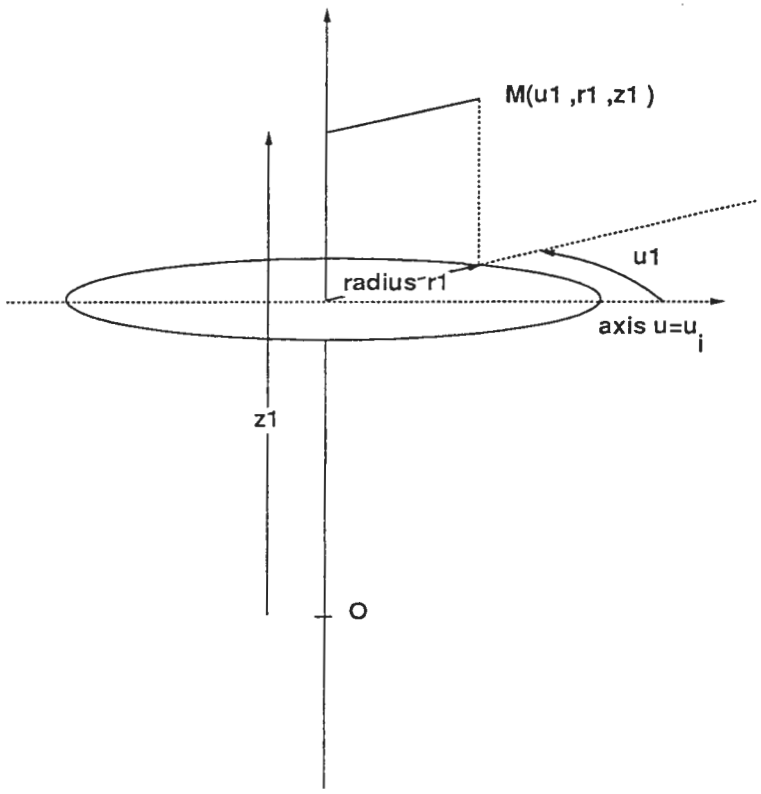


Figure 16: The (u,r,z) cylindrical coordinate system.

In this new system, the generated surfaces have the following expression:
$D_A(u,v) = (u, r_A(u,v), (1-v)*z_1 + v*z_2)$
$D_B(u,v) = (u, r_B(u,v), (1-v)*z_2 + v*z_3)$
$0 \leq v \leq 1$, $u_i \leq u \leq u_f$
where :
$r_A(u,v) = (1+S(v))*((1-R_n(v))*r_1(u) + R_n(v)*r_2(u))$
$r_B(u,v) = (1+S(v))*((1-R_m(v)*r_2(u) + R_m(v)*r_3(u))$

$r_i(u)$ for i=1,2,3 represents the radius of the contour i for the parameter u;

$z_i(v)$ for i=1,2,3 represents the z-coordinate of the point corresponding to the value v on the trajectory.

Similarly to the Cartesian coordinates, we have the following values for the derivative at the boundaries:

in cylindrical coordinates,

$$\frac{dr}{dv}|v=1 \quad = \quad r_2\frac{dS}{dv}|v=1 \tag{9}$$

$$\frac{dr}{dv}|v=0 \quad = \quad r_1\frac{dS}{dv}|v=0 \tag{10}$$

and the tangent line (relatively to the variable v) at the point $C(u, v_0)$ has the direction of vector: $(\frac{dr}{dv}|v=v_0, z_2 - z_1)$

Therefore, at the boundaries , the tangent lines (relatively to variable v) correspond to vectors:

$$(\quad r_2(u)\tfrac{dS}{dv}|v=1, \quad z_2 - z_1) \tag{11}$$

$$(\quad r_1(u)\tfrac{dS}{dv}|v=0, \quad z_2 - z_1) \tag{12}$$

3.2 The implementation

The three dimensional data acquired by the digitizer are used as points of the contours.

3.2.1 Digitizer operation

A helium-neon laser is used by the digitizer to detect the shape of the object being scanned. The laser shines out of the digitizer as a vertical plane of light.The digitizer moves in a circle that is 2.2 meters in diameter. The digitizer operates on the principle that light in a straight line, reflected off an object, can be viewed at a different angle to reveal a profile on the surface.

At a given point in the digitizer's travel, 256 data samples are taken vertically along the reflection of the line, each stored as the radius of that point from the center of rotation. As the digitizer makes a complete circle , 512 sets of these data samples are made and sent to the host computer to be stored as a grid of radius values.

If at a given point the digitizer receives no reflection (or too weak to detect) or the surface is out of the allowable range (center of rotation to a point 20 cm closer to the digitizer) , a special radius value is stored and called void. Void values can be replaced in the Echo software by using some kind of interpolation using non void values. During the digitizing operation, all operations are controlled remotely by the host computer (which is an Iris graphic workstation).

After the digitizing is complete, we use the Echo commands to display , modify and save the image.The Cyberware Echo software supports the acquisition ,

processing and output of 3D images obtained by the Cyberware 3D Digitizers. The software is intended to handle the images of 3D surfaces acquired by the Cyberware digitizers.The digitizer acquires 3D surface information as a series of profiles.

These profiles are defined as a series of coordinate values at regular intervals on the z-axis (used above; it is the axis which goes through all the contours). Typically , each profile (which lies on the z-axis) is divided into 512 intervals (512 intervals along z-axis) , each assigned a radius value . Portions of this profile may contain no value at all: these are places described as void points. Consequently, each profile is a two-dimensional object.

To define a three-dimensional surface, these profiles are digitized side-by-side across the subject surface with different angles : at each angle θ (512 angles regularly spaced between 0 and 2π), a profile is acquired as a set of 512 radius values along the z-axis . This forms a perfectly regular grid of radius values called a range map. For a given value z_0 on the z-axis, the set of 512 radius values taken at the 512 different angles (from 0 to 2π) is called a contour as in the definitions concerning generalized cylinders and homotopy sweep technique.

3.2.2 Polygon meshes for the output image

A polygon mesh is a set of connected polygonnally bounded planar surfaces. The representation is only approximate . The obvious errors in the representation can be made arbitrarily small by using more and more polygons. But this increases space requirements and the execution time of algorithms processing the representation . In our case, we used triangular patches for the displayed image on the screen of Silicon Graphics Iris workstations. t the same time, we used the Wavefront format of images and the Silicon Graphics Library for output images.

3.2.3 Planar Paths

As announced before, the implementations we have performed deal with homotopy sweep interpolation using planar paths between the corresponding points on two contours: the scaling functions S_x and S_y are therefore equal ($S_x = S_y = S$), that is, the deformation is the same in all directions (isotropic deformation).

The implementations are made on digitized human faces and use different values for the scaling functions. Here are the different cases we have tackled.

3.2.4 Null scaling function

The first implementation uses the value 0 for the scaling function:

$$S_x(v) = S_y(v) = S(v) = 0$$

$$\forall v \in [0, 1]$$

The blending function has the form defined in (8) . Finally, the G^1 continuity conditions are verified.

Result: Figure 17 (1 contour is taken among 10 digitized contours).

At each original contour of the generated face, the tangent plane is vertical i.e. oriented along the z-axis.

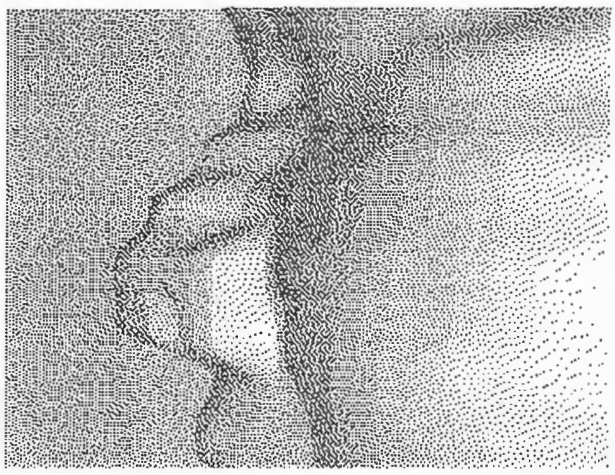


Figure 17: Generated human face from a set of contours: waves. Input: 1 contour among 10 digitized contours.

The result is a face on which waves show the effect of the blending function R_n on the transition between two adjacent contours. This is understandable according to the expressions (11) and (12) of the tangent vectors at the boundaries: the vectors for $v = 0$ and $v = 1$ are oriented along z-axis. It also means that we have to take into account the direction of tangent planes at the points of the original contours (which are given as data). Even if we can use different blending functions, or different parameters for the blending function defined in expression (8), we also need to calculate the tangents at original contours by using additionnal data and more points.

512 contours are acquired along z-axis. We have chosen one contour among ten equally spaced along z-axis.

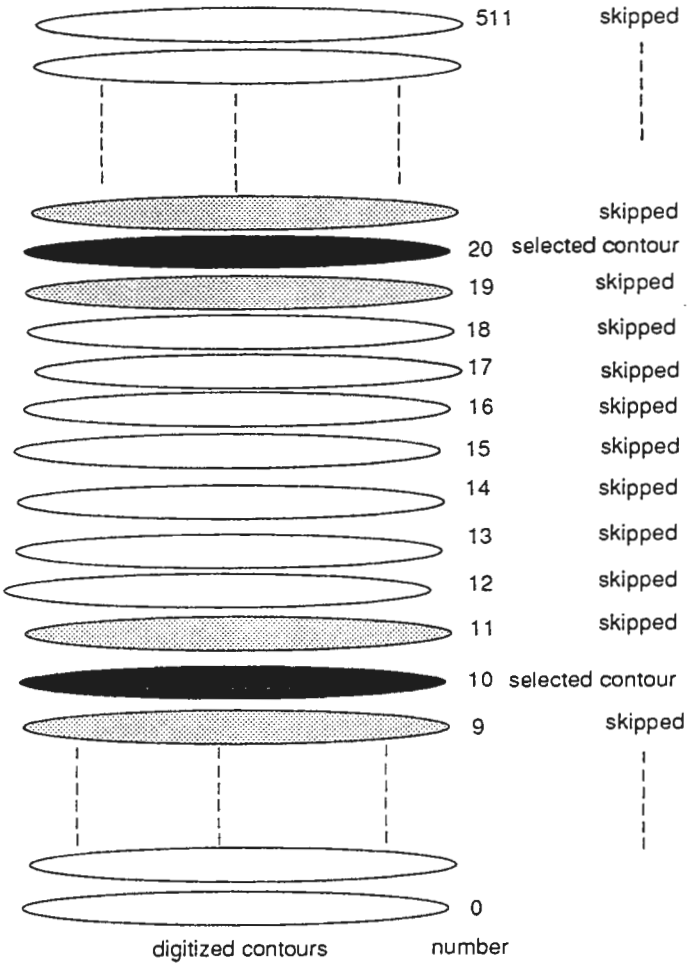


Figure 18: The selection of input contours among 512 digitized contours.

Scaling functions integrating the computed derivatives In expressions (11) and (12), the tangent lines on the given contours have the following direction:

at $M(v = 1)$

$$(r_2(u)\frac{dS}{dv}|_{v=1}, z_2 - z_1)$$

at $M(v=0)$

$$(r_1(u)\frac{dS}{dv}|_{v=0}, z_2 - z_1)$$

Let us normalise the value on the z-direction.Therefore, tangent line directions become:

at $M(v=1)$

$$(\frac{r_2(u)}{z_2 - z_1}\frac{dS}{dv}|_{v=1}, 1)$$

at $M(v=0)$

$$(\frac{r_1(u)}{z_2 - z_1}\frac{dS}{dv}|_{v=0}, 1)$$

Let us define the direction coefficient $c^A(u0)|_{v=v0}$ corresponding to surface A as the r-coordinate value of the tangent vector (relatively to v) at the point $C(u_0, v_0)$ of the generated surface when its z-coordinate is normalised to 1. In the case above,

$$c^A(u)|_{v=1} = \frac{r_2(u)}{z_2 - z_1}\frac{dS}{dv}|_{v=1}$$

$$c^A(u)|_{v=0} = \frac{r_1(u)}{z_2 - z_1}\frac{dS}{dv}|_{v=0}$$

The G^1 continuity conditions at a join point M between two generated surfaces A and B , can be expressed in the cylindrical coordinate system by the equality of the two direction coefficients corresponding to each surface , namely

$$\forall u, c^A(u)|_{v=1} = c^B(u)|_{v=0} \tag{13}$$

i.e.

$$\frac{r_2(u)}{z_2 - z_1}\frac{dS}{dv}|_{v=1} = \frac{r_2(u)}{z_3 - z_2}\frac{dS}{dv}|_{v=0}$$

which is finally the same condition as in equation (7):

$$\frac{dS}{dv}|_{v=1} = k\frac{dS}{dv}|_{v=0}$$

where $k = \frac{z_2 - z_1}{z_3 - z_2}$ and $S_x = S_y = S$.

In our implementation, we have calculated separately the two direction coefficients $c^A_{digitizer}(u)|_{v=1}$ and $c^B_{digitizer}(u)|_{v=0}$ using the data given by Cyberware digitizer.

Along the z-axis, 512 contours are represented. One contour is chosen among ten contours from the range map given by the digitizer: the contour 0 is chosen, 9 other contours are skipped, contour 10 is chosen, etc. To obtain the direction coefficients from the digitized face data , for instance for a chosen contour

$C_i^{digitizer}$ ($0 \leq i \leq 511$) , we compute the tangent values from the adjacent digitized contours $C_{i-1}^{digitizer}$ and $C_{i+1}^{digitizer}$. The two direction coefficients are computed between the following digitized contours:

for the surface A between $C_{i-1}^{digitizer}$ and $C_i^{digitizer}$,

$$c^A_{digitizer}(u)|_{v=1}$$

computed from the points $C_{i-1}^{digitizer}(u)$ and $C_i^{digitizer}(u)$;

for the surface B between $C_i^{digitizer}$ and $C_{i+1}^{digitizer}$,

$$c^B_{digitizer}(u)|_{v=0}$$

computed from the points $C_i^{digitizer}(u)$ and $C_{i+1}^{digitizer}(u)$;

The tangent vector coefficient is computed by using the direction of the straight line joining on one hand, the points $C_{i-1}^{digitizer}(u)$ and $C_i^{digitizer}(u)$, and , on the other hand, $C_{i-1}^{digitizer}(u)$ and $C_i^{digitizer}(u)$.

$$c^A_{digitizer}(u)|_{v=1} = \frac{radius(C_i^{digitizer}(u)) - radius(C_{i-1}^{digitizer})}{z_i - z_{i-1}}$$

$$c^B_{digitizer}(u)|_{v=0} = \frac{radius(C_{i+1}^{digitizer}(u)) - radius(C_i^{digitizer})}{z_{i+1} - zi}$$

After the computation of $c^A_{digitizer}(u)|_{v=1}$ and $c^B_{digitizer}(u)|_{v=0}$ these values are affected to $c^A(u)|_{v=1}$ and $c^B(u)|_{v=0}$:

$$c^A(u)|_{v=1} = c^A_{digitizer}(u)|_{v=1}$$

$$c^B(u)|_{v=0} = c^B_{digitizer}(u)|_{v=0}$$

and the direction coefficients are defined by:

$$c^A(u)|_{v=1} = \frac{r_i(u)}{z_i - z_{i-1}} \frac{dS^A}{dv}|_{v=1}$$

$$c^B(u)|_{v=0} = \frac{r_i(u)}{z_{i+1} - z_i} \frac{dS^B}{dv}|_{v=0}$$

so the conditions become:

$$c^A_{digitizer}(u)|_{v=1} = \frac{r_i(u)}{z_i - z_{i-1}} \frac{dS^A}{dv}|_{v=1}$$

and

$$c^B_{digitizer}(u)|_{v=0} \frac{r_i(u)}{z_{i+1} - z_i} \frac{dS^B}{dv}|_{v=0}$$

where $r_i(u)$ and z_i are respectively the radius and the z-coordinate of the point $C_i^{digitizer}(u)$.

For each surface A generated between two adjacent digitized contours, the scaling function S (which is equal to $S_x = S_y$) must satisfy conditions at $v = 0$ and $v = 1$:

$$S(0) = S(1) = 0$$

$$\frac{dS^A}{dv}|_{v=1} = \frac{z_i - z_{i-1}}{r_i(u)} c^A_{digitizer}(u)|_{v=1}$$

$$\frac{dS^A}{dv}|_{v=0} = c^A_{digitizer}(u)|_{v=0} \frac{z_i - z_{i-1}}{r_{i-1}(u)}$$

Result:

We generated the surfaces using the same number of contours as before (one selected contour among ten digitized ones) . The scaling functions we choose depend now on the parameter u (and not only v) because we use direction coefficients which are related to the angle (namely the variable u): they are simple polynomial functions which satisfies the constraints on the first derivatives at v=0 and v=1 .

$$S(v) = Sx(v) = Sy(v) = P(v)$$

where

$$P(v) = v.(v-1).((p_0 + p_1).x - p_0)$$

,

p_0 and p_1 are the first derivatives in 0 and 1 respectively.

In Figure (18), the dotted contours are adjacent to the selected ones: they are used for the computation of the direction coefficients.

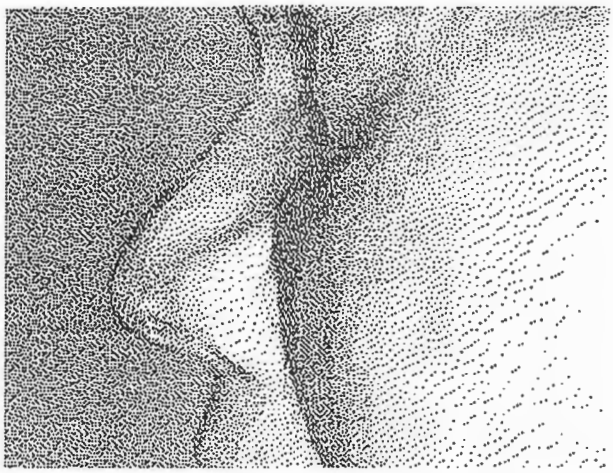


Figure 19: Generated human face from a set of contours: the cheeks are not perfectly smooth. Input: 1 contour among 10 digitized contours and corresponding adjacent contours for tangent planes computation.

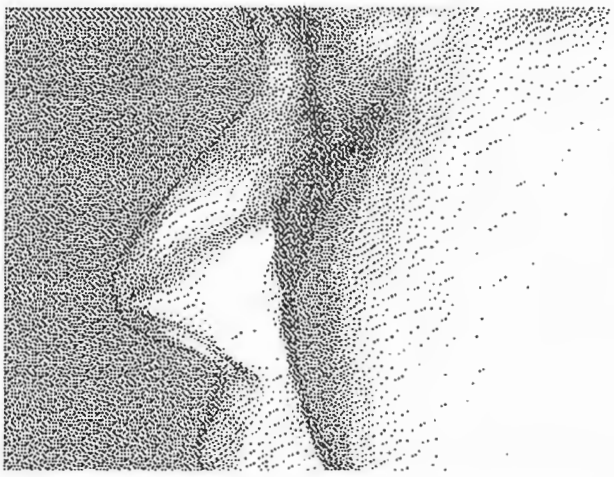


Figure 20: Generated human face from a set of contours: the cheeks are smooth. Input: 1 contour among 10 digitized contours and corresponding adjacent contours for tangent planes computation.

Eventhough the data given by Cyberware are very accurate, it is easy to notice that the direction coefficients computed on both sides of a selected digitized contour, are slight different. Another computation has been performed

by calculating the average of the two direction coefficients on both sides of a contour.

Figure (19) shows that tangent planes on both sides of a selected contour are different: the triangular patches on which is based the generated surface in our implementation are visible but the surfaces created on both sides of the input contours are much more distinguishable. Conversely,Figure (20) uses the average value of the computed direction coefficients on both sides of input contours.

The difference value in the case of a smooth cheek is certainly due to approximations made by the machine during the computations.

3.3 Conclusion

3.3.1 The results

The scaling functions are very important to perform an accurate homotopy sweep interpolation from a set of input contours. This means it is absolutely necessary to provide the model the data concerning the tangents on the object being represented.

In the above example, we used one contour among ten input contours (that is 10 % of data) and for each of the selected contours, we needed two additional contours to extract the direction coefficients : consequently, it was necessary to have 30 % of the digitized data to be able to reconstruct main features of the digitized human face or at least a generalized cylinder which is an approximation of the human face.

The scaling and blending function The scaling functions we have taken are polynomial functions with constraints on the first derivatives corresponding to the tangents at the boundaries. As we have an infinite set of possible scaling functions, we could use more sophiticated functions to fit closely the real object. But, obviously, it is easier to give explicitely the inbetween points than to try to give a scaling function joining these points. In that sense, it is more judicious to choose a simple scaling function which satisfies some first derivative conditions and to try by some method (for instance least square method) to adjust the parameter n of the blending function to the real inbetween points of real object contours.

The difficulty is due to the blending function which depends exclusively on the variable v and not on the variable u. This means that we have very similarly generated profiles at all values of v between two given contours : although the scaling functions are changing around the object, the transition from one contour to the other (which is represented by the blending function) is the same for all the angles (which correspond to the variable u). Actually, there is no reason to have such a symetry in the object to be represented.

If we use different parameters for each profile (i.e for each angle around the object), we make independent interpolations which are , in fact, two-dimensional interpolations at each angle (i.e. each profile) between two points with given tangents at these points. In that case, it is necessary to compare the homotopy sweep interpolation with other classical methods such as the spline technique.

Comparison with the spline technique Two kinds of splines are very popular: interpolating splines with C^1 continuity at knots and approximating splines with C^2 continuity at knots. The interpolating splines are divided into two distinct groups: the global cubic-spline interpolation for which any change made locally will be interpolated over the entire curve ; and the local cubic interpolation splines for which the curve at anyone point is dependent on at most five points, two ahead and two behind.

The homotopy sweep technique is typically a local interpolation method : the curve is dependent on one point ahead , one point behind and the two given tangents at these points. We can control the transition by changing the parameter n of the blending function and we can choose the suitable scaling function from one point to the other.

Some splines can be controlled along the trajectory by using tension parameters which allow the user to tighten the spline curve like in the case of ν-splines, cardinal splines or Catmull-Rom splines.

Therefore , there are many similarities .Nevertheless, the homotopy sweep interpolation needs a reference point (or axis) for each given point: that is, the homotopy sweep in our implementation allows us to interpolate real values such as radii but not geometric positions in a two or three dimensionnal space ; as we applied the particular case of homotopy, we need to provide the axis along which the radii are measured and interpolated. However, in the local spline interpolation techniques , only points in a two dimensional space are given: the interpolation does not depend on any reference point (or axis).

The discontinuities The free-form curves like human faces , clothes, moutains contain a certain number of discontinuities. Obviously, we can create these discontinuies if we try not to satisfy G^1 continuity conditions but the generated surfaces are absolutely continuous by definition of homotopy.This can also be considered as a disadvantage when discontinuities are wanted.However , spline methods also can not create discontinuities between control points . Anyway, it would be interesting to handle these discontinuities and to reconstruct them.

A too simplified model We adapted the general homotopy sweep technique to the example of human face representation but because of the particular case of a linear trajectory , and the planar path between two end points due to the lack of data concerning the location of important feature points, the model is very simplified . It could be improved by adapting the parameter n of the blending

function according to some optimization method. The tangents at these given points are necessary .

Other approaches During the implementation of the homotopy sweep technique , we were thinking about the generalization of this technique ,for example in the deformation of a surface to another surface with known correspondances between the points of the contours assuming that the surfaces are bounded by identical contours. This example looks like the animation from a surface to another by interpolating with time. We tried to add some physical constraints to this homotopy but we had to solve some equations which were general differential equations and for which it was more adapted to apply numerical analysis. In order to take advantage of the intuitive and convenient control of the deformation by using the homotopy sweep technique , we applied this model to an example of dynamic approach : the animation of the human mouth.

4 The Dynamic Approach

4.1 Introduction

In a real time process, only relatively simple calculations can be made because the image must be displayed in less than 1/15 seconds : otherwise, the illusion of continuous movement breaks down at slower speeds.

The formulation of the homotopy sweep technique is very simple and provides a control of the transition from one point to another with some conditions on the first derivatives at the end points. This is very interesting in the case of animation because the formulation we will see is inspired by the one we applied to the surface generation: the conditions on velocities are analogous to the G^1 continuity conditions and the generated trajectories for a point are similar to the generated surfaces in the case of three-dimensionnal representation.

Our purpose is to apply the homotopy sweep technique in a real time process to the movement of the lips for the global project of the new teleconferencing system with realistic sensations.

It would be very interesting to control the continuous mouth deformation just by specifying the coordinates of a few points and their velocities and to deduce the motion of all the other points.

We have tested this dynamic approach by applying a homotopy technique to the human mouth. The input was a set of two frames for which the two-dimensional locations and velocities of the markers were acquired and the output was the generation of inbetween frames which appeared like an animation sequence between the two selected frames.

The principal interest of this technique is that we can control the deformation between the two keyframes.

4.1.1 The General Scheme

Figure (21) shows the different processes of our dynamic approach. The input is given by the Lip Tracking Process , the Cyberware Digitizing Process and the manually entered wireframe.

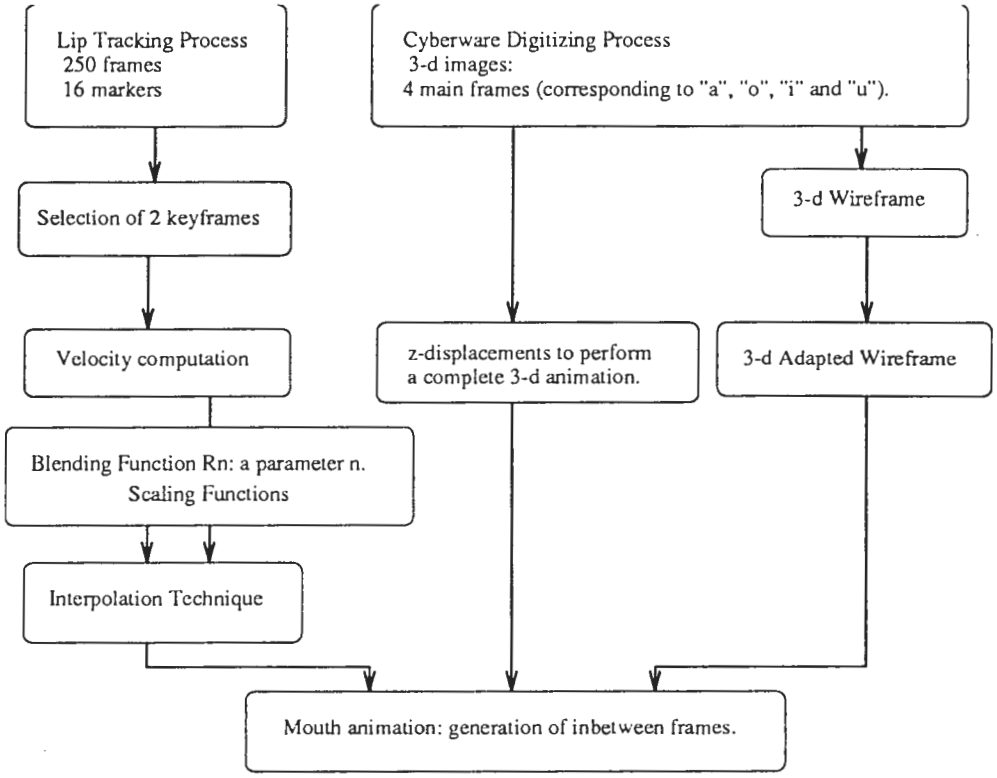


Figure 21: The Mouth Animation General Scheme.

A set of 250 frames was obtained thanks to the Lip Tracking Device with 16 markers on the mouth of a subject. 2 frames are taken from this set and , for each marker, velocity is computed by using adjacent frames (like in the above Static Approach).

A pre-existing three-dimensional wireframe is manually adapted to the location of the markers on the mouth. The three-dimensional animation is applied to the human mouth by only moving the points of the wireframe : the texture is also deformed in this motion. Therefore, a computed displacement is applied to each point on the wireframe to perform the animation.

The three-dimensional animation is completed by adding the displacement in the z-coordinate taken thanks to the Cyberware Digitizing Process.

Finally, we computed the wireframe points displacement by a Homotopic Interpolation for each marker with Blending Functions controlled by the parameters n, m and p, two scaling function : inbetween frames were generated from the two end key-frames.

The animation is therefore realized with some morphological assumptions

concerning the mouth muscles. The animation was recorded on a VHS tape with different configurations and parameters.

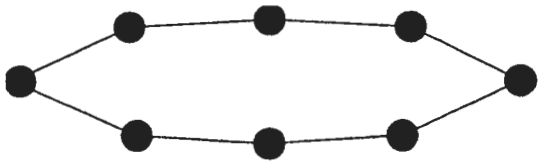
4.2 The Lip Tracking Device

The Lip Tracking Process has been performed thanks to Mr. Yoshifumi Kitamura, Researcher in the Articial Intelligence Department.

In order to obtain data relative to the lip movement , a lip traking system is available : a micro-camera and a helmet are interdependent and a system recognizing markers on the lips gives the locations of these points after each period of approximately one second. When tracking the motion of eight markers , the period is 0.8 second , and 1.2 seconds when we track 16 points.

The Lip Tracking System provides the coordinates of the points which are tracked in a two-dimensional space. The markers are set around the mouth, on the lips.

In the first experiments, we used 8 markers on the lips of an experimentator who moved his mouth according to the articulation of some letters like "a", "o", "i" and "u".



8 points

Figure 22: Lip Tracking: 8 markers.

After realizing 8 markers are not sufficient for mouth motion description, we added 8 more markers on another contour of the mouth which will be called the outter contour.

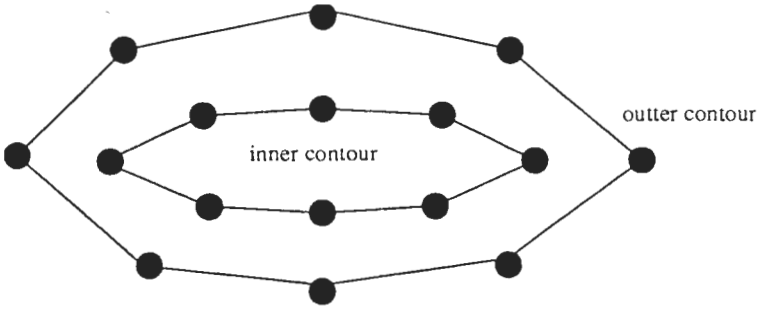


Figure 23: Lip Tracking: 16 markers.

We recorded the movement of 16 markers (for the same articulation of letters) which is sampled each 1.2 seconds: 250 samples from the Lip Tracking measurements were available for our interpolation. Moreover,thanks to the Cyberware three-dimensional digitizer , we scanned the experimentator's face to keep the exact locations of the markers on his face (in fact, four images were taken for the four main positions of his mouth among the 250 frames).

4.3 The interpolation

We assume having two frames with given point coordinates and velocities. The interpolation is made between these two end positions: therefore , we choose two samples among the ones taken by the Lip Tracking Device .

Let O be a point in the two-dimensional space , A a marker on the lip and B the corresponding point in the second frame.

A local cylindrical coordinate system (O, $\vec{U_r}(\theta)$, $\vec{U_\theta}(\theta)$) is attached to the initial point A and depends on the angle θ between (OA) and (Ox) (cf Figure 24).

The correspondances between the points are supposed to be given. The time for the two keyframes is normalised to $t = 0$ and $t = 1$. At an instant t and for the angle θ , a point $M(\theta, t)$ is generated between A and B.

In (O, $\vec{U_r}(\theta)$, $\vec{U_\theta}(\theta)$), $M(\theta, t)$ is defined by :

$$\vec{OM}(\theta, t) = r(\theta, t)\vec{U_r}(\theta) + w(\theta, t)\vec{U_\theta}(\theta).$$

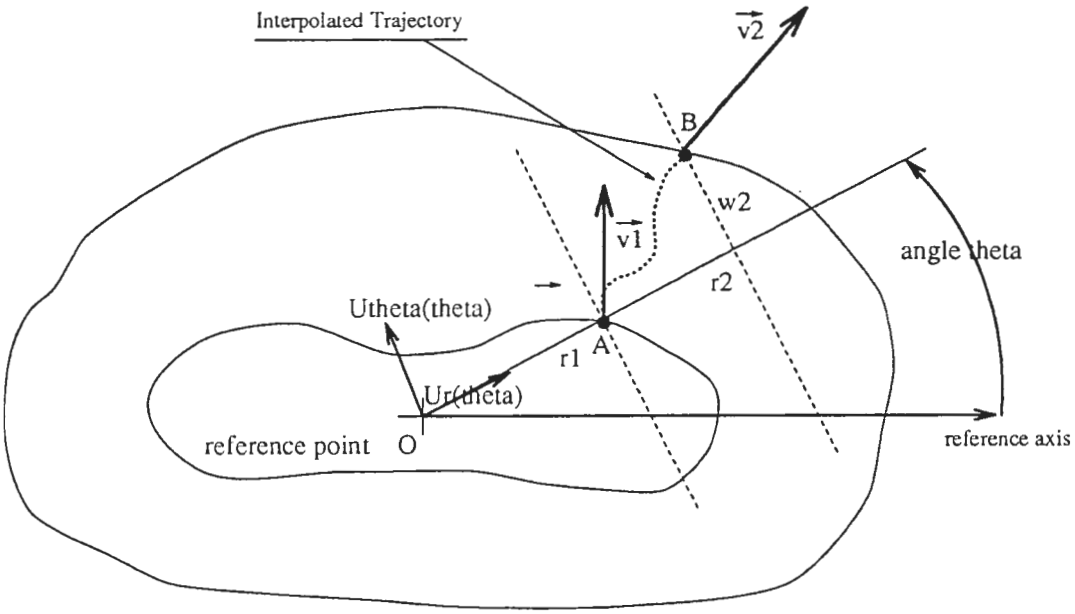


Figure 24: Coordinate System and Interpolation.

The conditions at the limits are imposed as following:

$$M(\theta, 0) = A$$

$$M(\theta, 1) = B$$

$$\frac{d\vec{M}}{dt}|_{t=0} = \vec{v_1}$$

$$\frac{d\vec{M}}{dt}|_{t=1} = \vec{v_2}$$

The velocities are also expressed in this coordinate system:

$$\vec{v_1} = v_{1r}\vec{U_r}(\theta) + v_{1\theta}\vec{U_\theta}(\theta)$$

$$\vec{v_2} = v_{2r}\vec{U_r}(\theta) + v_{2\theta}\vec{U_\theta}(\theta)$$

$$\vec{OA} = r_1\vec{U_r}(\theta)$$

$$\vec{OB} = r_2\vec{U_r}(\theta) + w_2\vec{U_\theta}(\theta)$$

The constraints on velocities are given by:

$$\frac{dr}{dt}|_{t=0} = v_{1r}$$

$$\frac{dr}{dt}|_{t=1} = v_{2r}$$

$$\frac{dw}{dt}|_{t=0} = v_{1\theta}$$

$$\frac{dw}{dt}|_{t=1} = v_{2\theta}$$

By analogy with the application of the Homotopy Sweep Technique in the static approach, we propose the following interpolations for each of the r and w components.

4.3.1 The r-component

The interpolation for r is done between the two values r_1 and r_2:

$$r(\theta, t) = (1 - R_n(t))(1 + S_1(t))r_1 + R_n(t)(1 + S_2(t))r_2$$

Therefore, the first derivative is :

$$\frac{dr}{dt} = \frac{dR_n}{dt}(-(1 + S_x)r_1 + (1 + Sy)r_2) + (1 - R_n)r_1\frac{dS_x}{dt} + R_n(t)r_2\frac{dS_y}{dt}$$

Let us choose the second form for the blending function R_n. The velocity conditions are expressed by:

$$\frac{dS_1}{dt}|_{t=0} = \frac{v_{1r}}{r_1}$$

$$\frac{dS_2}{dt}|_{t=1} = \frac{v_{2r}}{r_2}$$

These velocity conditions can be satisfied easily by using some scaling functions S_1 and S_2 with velocity conditions at $t = 0$ or $t = 1$.

Finally, we can choose any parameter n for the blending function R_n (to control the transition from r_1 to r_2) and the scaling functions S_1 and S_2.

4.3.2 The w-component

The interpolation is performed as above for the w-component from 0 to w_2. In our implementation, we made the following interpolation :

$$w(\theta, t) = (1 - R_m(t))(1 + S_1(t))A + R_m(t)(1 + S_2(t))(w_2 + A) - A$$

where m is the parameter for the second form of the blending function and A is an offset value which translates the interpolation from the interval $[0, w_2]$ to $[A, w_2 + A]$. The velocity constraints are given by:

$$\frac{dw}{dt}|_{t=0} = v_{1\theta}$$

$$\frac{dw}{dt}|_{t=1} = v_{2\theta}$$

$$w(\theta, 0) = 0$$

$$w(\theta, 1) = w_2$$

As for the interpolation for r, the conditions at $t = 0$ and $t = 1$ are given by :

$$\frac{dS_1}{dt}|_{t=0} = \frac{v_{1\theta}}{A}$$

$$\frac{dS_2}{dt}|_{t=1} = \frac{v_{2\theta}}{w_2 + A}$$

The examples of scaling functions are in the Appendix I.

4.3.3 The reference point

In the homotopy technique, it is necessary to choose a reference point because the interpolation is achieved between two values. Since mouth has a vertical symetrical axis, we choosed the reference point on that axis. It is clear that we can choose a reference point for each point to be moved but we made the implementation by using the same reference point for all the markers.

4.3.4 The z-coordinate for animation

The animation was first made by using the displacements in a two-dimensional space . Actually, we first implemented the motion using this two dimensional displacement to check the validity of our experiment and our implementation. Then we added the z-component for mouth animation thanks to the 3-d images we saved during the Cyberware Digitizing Process. For four main frames corresponding to the articulation of four letters, we had the 3-d coordinate of each marker. The interpolation we developped considers the velocity constraints as limit conditions, but actually, we only had 2-dimensional velocities because we used only one camera for Lip Tracking. This is the reason why we chose the very simple following formulation:

$$z(t) = (1 - R_p(t))z_1 + R_p(t)z_2$$

where z_1 and z_2 are the z-coordinates of two face expressions for the same marker. We chose the closest main frame (which correspond to one of the articulations "a", "o", "i" or "u") to the selected key-frame (among the 250 possible frames).

4.4 The velocity computation

Since 250 samples were available, we simply computed the velocity for each marker thanks to the adjacent frames, for example by computing the average displacement between this frame and k frames before (or after). This is acceptable because only 2 choosed key-frames (among the 250 available samples) were considered as the end frames in our interpolation.

4.5 The wireframe

4.5.1 The initial wireframe

In order to perform a human face animation, it is common to handle a wireframe which contains the coordinates of distinguishable three-dimensional points joined by straight lines: the information concerning the shape and the texture are therefore represented by the polygons defined in the wireframe.

Usually, a wireframe of the human face which was marked in the Lip Tracking Process already exists (in our case, the subject's wireframe had been taken manually from the three-dimensional images using some software to handle this kind of images) .

In the first attempt to apply this interpolation to the mouth animation, the wireframe was fixed a priori and did not correspond to the locations of the markers during the Lip Tracking measurements . The results of the interpolation were not very satisfying.

However, it was possible to adapt manually the wireframe to the Lip Tracking markers: we made some of the wireframe points correspond to the markers by using the Cyberware three-dimensional images taken immediately after the Lip Tracking process .

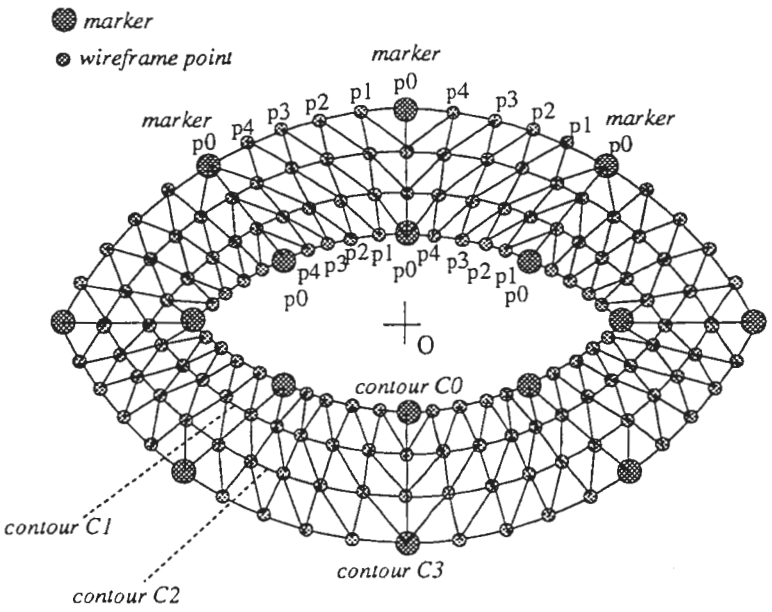


Figure 25: The wireframe is adapted to the 16 markers.

4.5.2 The adapted wireframe

Since we had 3-d images of the subject, we adapted manually some points of the wireframe to our markers and we made the wireframe contours correspond to the circular muscle around the mouth.

Figure (25) shows 4 contours in the wireframe . The extreme contours C_0 and C_3 correspond to our inner and outter contours and we know the displacement of only 8 points on each contour: the markers are the biggest circles.

For a given pair of adjacent markers on the same contour, let us call $p0$ the location of the markers on that contour and $p1$, $p2$, $p3$ and $p4$ the four inbetween points on the wireframe; the node sequence on the wireframe is therefore $p0$ (=marker), $p1$, $p2$, $p3$, $p4$, $p0$ (=marker) (Figure 25).

4.5.3 The inbetween contours C_1 and C_2

Hypotheses. We assume that we know the velocities and locations at $t = 0$ and $t = 1$ of the markers: this enables us to calculate the displacement for that markers at any instant t ($t \in [0, 1]$) thanks to the previous interpolation.

Our problem is to deduce the displacement of the points on the inbetween contours C_1 and C_2.

Dilatation and compression of the circular muscle. A muscle is compressed and dilated according to its fiber constitution . Our assumption is to consider the same deformation for all the fibers of a circular muscle (cf Figure 26).

Let A_0 and A_3 be the markers on the outter and inner contours and A_1 , A_2 the points on the inbetween contours C_1 and C_2.

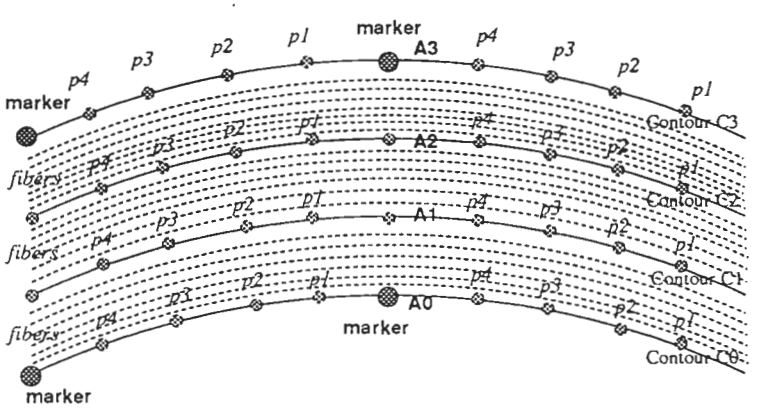


Figure 26: Mouth muscle fibers.

The proportionality assumption for dilatation and compression can be expressed as follows:

$\frac{A_0A_1}{A_0A_3}$ and $\frac{A_0A_2}{A_0A_3}$ are constant $\forall t \in [0,1]$.

In our implementation, the first expression is taken equal to $\frac{1}{3}$ and the second one to $\frac{2}{3}$.

Let us call the wireframe points A_1 and A_2 the first stage points.

4.5.4 The inbetween points on the same contour ($p1$ to $p4$)

Hypotheses. The location of all points $p0$ (=marker) and their velocities are known at $t = 0$ and $t = 1$. The interpolation technique let us control the displacements from $t = 0$ to $t = 1$ of the markers.

The displacement of the first stage points A_1 and A_2 on inbetween contours C_1 and C_2 is also supposed to be known.

The problem is to deduce the displacement of the inbetween points from the markers displacement (for the points on the outter and inner contours) and from the first stage points displacement (for the inbetween contours).

Linear homotopy. Since we do not have a priori any constraint at the limits (for velocity or displacement) and we do not know the shape of the curve joining the markers (or the first stage points) , we first just apply a linear homotopy to

find the displacement of the inbetween points. It is clear that this interpolation can be improved.

4.6 The Texture Mapping

I first made animations on Silicon Graphics workstations and, when I obtained the promising first results with animation examples using simple geometrical objects, we decided to realize a complete demonstration with real texture mapping of my animation model. I was proposed to work in collaboration with a programmer who would adapt his graphics software (called "live") to the data I would give him in real time (via a unix pipeline).

4.7 Conclusion

I introduced a formulation for the mouth animation in analogy with the Static Approach and I showed how we can easily control the deformation during the interpolation by using a few parameters. Eventhough I made a few assumptions relative to circular fibers, the animation was very realistic without using any physical constraints.

After the implementation of the dynamic approach, I realized it was possible to make a realistic animation without any complex physical constraints on the model and I understood the texture mapping deformation is more important than the volume deformation in human face animation.

The animation does not need time consuming computation and can be used in real time. For a future work, the parameters of the scaling functions can be improved easily and the method can be applied to wrinckles animation, for instance.

The VHS animation tape I realized in collaboration with Mr. Yoshifumi Kitamura is one of the demonstrations showed to ATR's visitors.

5 Conclusion

5.0.1 The technical experience in ATR

During the five months of my stay in ATR, I learned very much about the techniques used in computer vision and computer graphics thanks to the documentation I read and to the tools I used . I got acquainted with the Silicon Graphics workstations and with the three-dimensional Cyberware Digitizer ; I dealt with many image format problems ; I used many differents tools and I had to make all these tools communicate to achieve my goal. I think this is a good engineer experience.

I also had very theoretical discussions with Prof. Narendra Ahuja and Dr. Tanaka to try to generalize the Homotopy Sweep Model to a global model for three-dimensional surface deformation but five months were very short to obtain such a general formulation. However, I had a very good contact with research fields and researchers in general which makes me intend to work later in these fields.

5.0.2 The technical results of my internship

I was first proposed the Homotopy Sweep technique as a possible data representation model for human face (or other parts of the human body like arms). I studied this method and the particular case of planar paths, and tried to generalize this interpolation to three-dimensional surfaces in very interesting discussions with my supervisor, Dr. Hiromi Tanaka and the Professor Narendra Ahuja.

After the implementation of this Static Approach, I evaluated this method in comparison with other methods and showed the problems I encountered such as the discontinuities on the human face: when using Generalized Cylinders, all the discontinuities of the human face become continuous because the masterword of the Homotopy Sweep Technique is continuity everywhere. Obviously, we can just avoid applying $G^1 - continuity$ for surface generation between contours to obtain the desired discontinuities but this can not be used for all the existing discontinuities in the human face.

This leaded me to think about using this homotopy in human face animation because in this case, we can easily fix the velocity constraints as we fixed the $G^1 - continuity$ constraints for the static approach. I therefore wrote an interpolation formulation in analogy with the Static Approach formulation.I first obtained some results with geometrical objects which are deformed into other objects smoothly. After realizing these deformations can be very efficient in animation techniques, my supervisor proposed me that a programmer in my laboratory could write a special program for the output image: I therefore computed the animation parameters and the deformation of each point of the wireframe and I used his special program to animate the image.

The result is impressive and the deformation looks realistic as we can see on the videotape. I believe this Dynamic Approach can be improved in many ways in order to be applied to other parts of the human face, such as the wrinkles, for instance. After my internship, I am still in contact with ATR and with Dr. Hiromi Tanaka and I intend to write a paper for an international conference on image processing to be hold in the end of 1993.

5.0.3 My personal experience

During my internship in ATR, I acquired experiences and skills from a methological and technical point of view as well as from a human point of view.

First of all, the ATR research laboratories are a priviledged environment for performing researches with important facilities at the disposition of the researchers.

From a technical point of view, I could perform a lot of experimentations in the field of image processing and data representation and I got acquainted to many techniques and problems encountered in the field of computer vision. The excellent computing tools in my laboratory and the kindness of all the researchers helped me constantly. I was lead to attend to talks of many famous researchers from other international research centers and to read a lot of documentation on image processing, animation techniques and representation models.

From a methodological point of view, the subject of my research was limited in goal but not in the ways to reach this goal, because I had to define the main steps of my work by myself, under the supervision of Dr. Hiromi Tanaka and not just to implement given algorithms . I therefore had an experience of real research in the computer vision and graphics field, including experimentations , implementation and problem solving. After the first two weeks of my internship , I introduced the general principle of the Homotopy Sweep Method and how I could use it in the case of human face representation. I am very grateful to my superiors (Dr. Hiromi Tanaka and Dr. Fumio Kishino) who were confident enough in me to give me such a responsability from the start of my internship.

I was very much interested in observing japanese methods of work. In my laboratory , the researchers are working in many different fields (artificial intelligence , man-machine interfaces, differents areas of image processing, networks) and it is customary to hold a regular meeting every week where several researchers explain the progress in their research. Thanks to these discussions , everyone is aware of problems or difficulties of the other researchers and can give advice. Unfortunately, these meetings were held in japanese and of course, were not as profitable as if they were in english eventhough people tried to switch to english as soon as possible. At least once a week , visitors (Japanese and foreigners) were demonstrated research results: this kind of policy is fruitful in the long run because it is a source of exchange of knowledge and can lead to further cooperation, or more prosaically to financial supports.

From the human relationship point of view, I feel that being employed in a japanese company is certainly a good way to discover Japan and its people. Moreover, the Kansai area is a splendid region interesting at the touristical and cultural point of view.

Japanese companies are often spoken of as big families. In fact, as far as I could see, the relationships in my laboratory were very much according to fact. When I arrived, I was greeted very kindly and everybody tried to make me feel comfortable and at ease. In everyday life, the masterword seems to be harmony and everyone avoids any direct clash or remark. "Laboratory parties" are held one or twice a month and everyone has a good time and can get to know the others better. This creates a very good ambiance of work into which the newcomer is integrated veyr quickly.

In fact, after having spent eight months in a japanese environment, I feel I have some general ideas about japanese behavior although I would not dare saying I understood all of it. Anyhow, I could go through the cliches in use in Europe to get to know better the japanese people and appreciate them a lot.

6 Bibliography

References

[1] Chiew-Lan Tai, Kia-Fock Loe, Tosiyasu L. Kunii. "Integrated Homotopy Sweep Technique for Computer-Aided Geometric Design", Computer Vision p.583 Springer-Verlag (1991)

[2] Nadia Magnenat-Thalmann, Daniel Thalmann. "Image Synthesis", Springer-Verlag (1992)

[3] "State-of-the-Art in Computer Animation", Springer-Verlag (1992)

[4] C. Braccini, S. Curinga, A. A. Grattarola, F. Lavagetto. "Muscle Modeling for Facial Animation in Videophone Coding", IEEE International Workshop on Robot and Human Communication (1992)

[5] F. Lavagetto, S. Curinga. "Videophone Coding Based on 3D Modeling of Facial Muscles", SPIE Vol.1818 Visual Communications and Image Processing (1992)

A Appendix I: Scaling functions.

In our implementation, we used the following second form for the Blending Function:

```
float Rn(n,v)
        float v;
        float n;
{
        float value;
        value=( ((1+n)*v*v)/ ( (2+n)*v*v-2*v+1 ) );
        return value;
}
```

In file "hpy.c", we first used the following formulation for the r and w-component:

```
float Sx(v,tA)
        float v;
        float tA;
{
float value;
float Cte;
        Cte= tA;
        value= (0.25-( (v-0.5)*(v-0.5) ) )*Cte ;
        return value ;
}
float Sy(v,tB)
        float v;
        float tB;
{
float value;
float Cte;
        Cte= -tB;
        value= (0.25-( (v-0.5)*(v-0.5) ) )*Cte ;
        return value ;
}

float r(n,rA,rB,VrA,VrB,t)
float n,rA,rB,VrA,VrB,t;
{
float value,tan_in_0,tan_in_1;
tan_in_0=VrA/rA;
tan_in_1=VrB/rB;
```

```
value=(1.-Rn(n,t))*(1.+Sx(t,tan_in_0))*rA +  Rn(n,t)*(1.+Sy(t,tan_in_1))*rB  ;
return value;
}

w(m,wB,VthetaA,VthetaB,t)
float m,wB,VthetaA,VthetaB,t;
{
float value,tan_in_0,tan_in_1;
tan_in_0=VthetaA/Trans_value;
tan_in_1=VthetaB/(wB+Trans_value);

value=(1.-Rn(m,t))*(1.+Sx(t,tan_in_0))*(Trans_value)+Rn(m,t)*(1.+Sy(t,tan_in_1))
*(wB+Trans_value)-Trans_value;
return value;
}
```

We changed this formulation by adding parameters alpha, beta, gamma and delta for each component to control the scaling functions defined as following:

```
float Sx(x,tA,alpha,beta)
float x,tA,alpha,beta;
{
float value,a,b;
        a=((beta/(alpha*(alpha-1)))+tA)/alpha;
        b=-tA;
        value=x*(x-1)*(a*x+b);
        return value;
/*polynome passant par (0,0) (alpha,beta) (1,0) ayant pour pente tA en 0*/
}

float Sy(v,tB,gamma,delta)
/* polynome passant par (0,0) (gamma,delta) (1,0) ayant pour pente tB en 1 */
float v,tB,gamma,delta;
{
float value,a,b,alpha,beta,tA;
        alpha=1-gamma;  beta=delta;     tA=-tB;
        value= Sx( (1-v),tA,alpha,beta);
        return value;
}


/*la hauteur de la scaling function correspondante est:
```

```
beta_mult_pente*(la hauteur precedente)
        avec la hauteur precedente= pente en 0 ou en 1 divisee par 4

mm chose pour delta_mult_pente.
*/

float r(n,rA,rB,VrA,VrB,t,
        alpha,  beta_mult_pente,
        gamma,  delta_mult_pente)
float n,rA,rB,VrA,VrB,t,alpha,beta_mult_pente,gamma,delta_mult_pente;
{
float value,tan_in_0,tan_in_1,scalingX,scalingY;
tan_in_0=VrA/rA;
tan_in_1=VrB/rB;
scalingX=Sx(t,tan_in_0,alpha,beta_mult_pente*tan_in_0/4.);
scalingY=Sy(t,tan_in_1,gamma,delta_mult_pente*tan_in_1/4.);

value=(1.-Rn(n,t))*(1.+scalingX)*rA +  Rn(n,t)*(1.+scalingY)*rB  ;
return value;
}

w(m,wB,VthetaA,VthetaB,t,
        alpha,  beta_mult_pente,
        gamma,  delta_mult_pente)
float m,wB,VthetaA,VthetaB,t,alpha,beta_mult_pente,gamma,delta_mult_pente;
{
float value,tan_in_0,tan_in_1,scalingX,scalingY;
tan_in_0=VthetaA/Trans_value;
tan_in_1=VthetaB/(wB+Trans_value);
scalingX=Sx(t,tan_in_0,alpha,beta_mult_pente*tan_in_0/4.);
scalingY=Sy(t,tan_in_1,gamma,delta_mult_pente*tan_in_1/4.);

value=(1.-Rn(m,t))*(1.+scalingX)*(Trans_value)+Rn(m,t)*(1.+scalingY)*(wB+Trans_v
alue)-Trans_value;
return value;
}
```

Here is the main function of the animation:

```
void hpy(A,B,Xorigin,Yorigin,Interpol,n,m,R_alpha,R_beta_mult_pente,R_gamma,R_de
lta_mult_pente,W_alpha,W_beta_mult_pente,W_gamma,W_delta_mult_pente)
float A[N][4], B[N][4];
float Xorigin,Yorigin;
```

```
float Interpol[N][frequency][3];
float n,m,R_alpha,R_beta_mult_pente,R_gamma,R_delta_mult_pente,W_alpha,W_beta_mu
lt_pente,W_gamma,W_delta_mult_pente;
{
float XA,XB,YA,YB,VXA,VXB,VYA,VYB,rA,rB,Urx,Ury,Uthetax,Uthetay,wB,rt,wt;
int i,k;

/* INITIALISATION OF Interpol with given points A & B */
for (i=0;i<N;i++) {
        Interpol[i][0][0]=A[i][0];
        Interpol[i][0][1]=A[i][1];
        Interpol[i][0][2]=0;
        Interpol[i][frequency-1][0]=B[i][0];
        Interpol[i][frequency-1][1]=B[i][1];
        Interpol[i][frequency-1][2]=0;
}
/* COMPUTATION OF remaining values for curves between A-curve & B-curve */
for (i=0;i<N;i++) {
float VrA ,VrB,VthetaA,VthetaB;
        /* initialisation of variables */
        XA=A[i][0]-Xorigin; YA=A[i][1]-Yorigin;
        XB=B[i][0]-Xorigin; YB=B[i][1]-Yorigin;
        VXA=A[i][2]; VYA=A[i][3];
        VXB=B[i][2]; VYB=B[i][3];
        rA=fsqrt( XA*XA+YA*YA );
        rB=fsqrt( XB*XB+YB*YB );
        Urx=XA/rA;      Ury=YA/rA;
        Uthetax=-Ury ;  Uthetay=Urx ;
        wB= XB*Uthetax + YB*Uthetay;
VrA=VXA*Urx + VYA*Ury; /*produit scal.*/
VthetaA=VXA*Uthetax + VYA*Uthetay;

VrB=VXB*Urx +VYB*Ury;
VthetaB=VXB*Uthetax + VYB*Uthetay;

        /* Computation for k=0 (point A) until k=frequency-1 (pointB) */
        for (k=0; k <= frequency-1; k++) {
float t;
t=k/( (float) frequency);
/*using homotopy***********************************************/
rt=r(n,rA,rB,VrA,VrB,t,R_alpha,R_beta_mult_pente,R_gamma,R_delta_mult_pente);
wt=w(m,wB,VthetaA,VthetaB,t,W_alpha,W_beta_mult_pente,W_gamma,W_delta_mult_pente
);
```

```
Interpol[i][k][0]=rt*Urx + wt*Uthetax +Xorigin;
Interpol[i][k][1]=rt*Ury + wt*Uthetay +Yorigin;
Interpol[i][k][2]=0;
        }
}

}
```