〔非公開〕

ＴＲ－Ｃ－００９０

表情表現を考慮した顔特徴点抽出に関する検討

－表情変化による顔皮膚表面の時間的変位の計測箇所と表情再現品質の関係の基礎検討－

坂口竜己 Tatsumi SAKAGUCHI
大谷淳 Jun OHYA
岸野文郎 Fumio KISHINO

１９９３　９．２

ＡＴＲ通信システム研究所

表情表現を考慮した顔特徴点抽出に関する検討

- 表情変化による顔皮膚表面の時間的変位の計測箇所と表情再現品質の関係の基礎検討 -

坂口　竜己　TATSUMI SAKAGUCHI
大谷　淳　JUN OHYA
岸野　文郎　FUMIO KISHINO

1993. 9. 2

ATR通信システム研究所

目　　次

第１章

はじめに

本報告は、１９９３年７月から同８月までの２ヶ月間、（株）ATR通信システム研究所知能処理研究室において、表情表現を考慮した顔特徴点抽出に関する検討結果をまとめたものである。

以前より当研究室が実験を行なってきた臨場感通信において、人間の顔表情の分析合成技術は、欠くことの出来ない重要な要素である。

現在送信側では、顔に数箇所の測定点を決定し、マーカーを貼付、そのマーカーを時間毎に位置計測し移動量を送り出す。また受信側においては受信された移動量によって３次元顔モデルを変形し動画像を作成している。

ここで設定されている測定点数（マーカー数）は、通信の容量・端末能力の制限より決定したものであり、人間の多彩な顔表情をすべて忠実に再現できるものではない。

そこで本研究では、顔表情を忠実に再現するために必要な測定点数、および３次元モデル変形のためのアルゴリズムを検討した。

また、人間の顔表情表出時の、皮膚表面各部位の運動の３次元的・時系列的な検討も行なった。

本文では、以上のテーマについて、４章に分けて検討・実験結果を報告する。

まず、第２章では、現在用いられている顔表情の分析合成手法を再考し、問題点等を導く。

次に第３・４章では、人間の顔表情表出時の皮膚表面各部位の運動を計測するための手法について述べる。また、計測結果から表情を再合成する手法についても述べ、その実験結果を示す。

次に第５章では、測定に用いたマーカー数と再合成された画像の品質について検討し、適切な測定点数および位置を考察、その実験結果を示す。

最後に第６章では全体のまとめと今後の課題について述べる。

第２章

臨場感通信に於ける顔画像の分析合成手法

本章では、現在臨場感通信会議システムで用いられている、顔表情の分析・合成技術について述べてゆく。またその抱える問題点を挙げ、本研究の位置づけを行なう。

２.１　分析・認識系[1]

システム内に於て顔表情の変化は、ヘルメットに付けられたカメラからの画像を元に測定される。

被験者は顔各部位に、検出を容易にするためのマーカーを貼付ける。撮影用カメラ位置は、被験者の頭部に被せられたヘルメットに固定されている。これは、外界に固定されたカメラで対象を撮影した場合、検出対象のマーカーの動きが、本来の表情による変化に加えて、頭部の移動・回転等の動きが加わったものとなり、表情のみに起因する移動量を導く計算が困難になるためである。

撮影環境と、被験者の顔に貼付するマーカーの位置を、図２－１に示す。

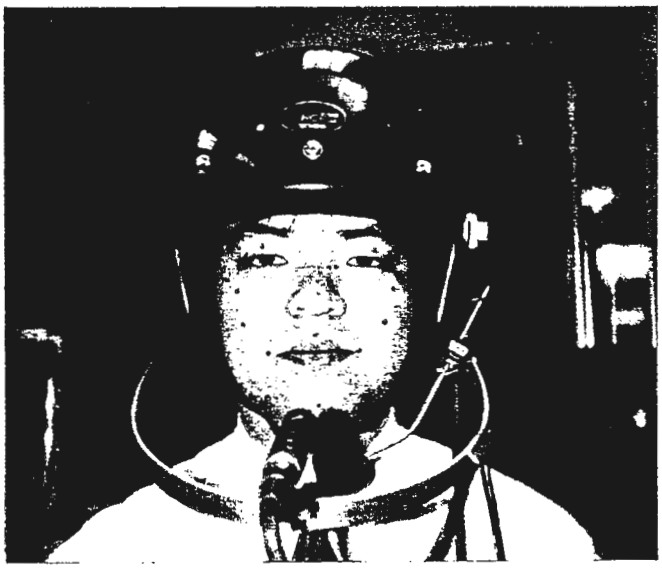


図２－１　臨場感通信会議システムに於ける、対象の撮影環境とマーカー位置

頭部固定カメラから得られた画像は、マーカーの色のみを抽出する２値化処理が施され、認識系に与えられる。

認識系に於ては、初期画像からマーカー各部位が含まれる領域決定を行ない、以後入力される連続画像中で、この領域を追跡、その移動量を算出して合成系に送信する。

ここで得られるマーカーの移動ベクトルは２次元のものである。

2.2　合成系

顔画像の再合成には、モデルベースの技術を用いる。

サイバーウェア社の３次元ディジタイザーによって得られるモデルとテクスチャを利用して、対象人物顔画像の再合成を行なう。

表情の合成を行なう際には、この取得されたモデルの各格子点を変化に応じて移動させ、テクスチャを貼付する。

本システムに於てこれらモデル変形に携わるパラメータは、認識結果から導かれる１０箇所のマーカー位置の移動量のみである。実際のモデルには１０００点程度の格子点数が存在するため、マーカー位置以外の格子点については何らかの規則を設定し、周囲の測定点と協調移動させる必要がある。

この移動規則は、測定点位置と各格子点との距離などに応じて設定している。

画像合成に用いているモデルを図２－２に示す。

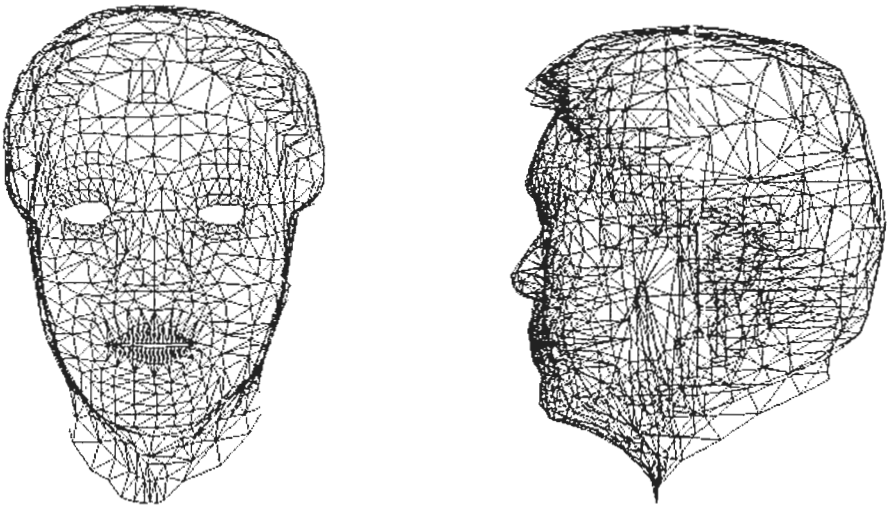


図２－２　顔画像合成に用いる頭部モデル

2.3　システムの問題点[2][3]

本節では、現在行なわれている臨場感通信会議システムの顔表情分析・合成部に関する問題点を挙げ、本研究の担う役割を述べる。

測定点数に関する問題：

現在設定されている測定点数（マーカー貼付数）は、１０箇所である（図２－１参照）。この測定点数は通信時のリアルタイム性を重視するために、通信レートや端末処理能力によって決定されたものであり、他の表情分析合成の研究と比べて著しく少ない数字である。

顔表情を表現するためにいくつの測定点が必要かという定量的な研究はなされていないが、崔らの研究では２４箇所、小林らの研究では３０箇所の測定点を利用して顔表情の分析合成が行なわれており、これらに比べると少ないと言えるであろう。

特に顔表情で最も感情の良く現われる、目・眉の周辺に測定点の少ないことには問題があると言える。

測定点数の少ない分析合成系では、その不足を補うために、合成時に用いられるモデルの変形規則を複雑にせざるを得ない。この結果、移動規則は使用するモデルや対象人物に特化したものとなりがちである。

これらは幅広いシステムの応用を考えた際に問題となってくるであろう。

次元数の問題：

測定系に於て用いている画像は、ヘルメット固定の単眼カメラから得られる、２次元画像である。このため得られる測定点の移動量も２次元のものとなる。

これに対して合成で用いるモデルは３次元のモデルであり、ここに次元数の差という問題が生じる。

現在は、得られた２次元の移動ベクトルを元に、各測定点ごとに３次元の動きベクトルを推定する規則を決定しておき、疑似的に３次元化する手法が取られている。しかしこの推定規則も実測した結果から導かれたものではなく、また実際の動作よりも誇張するよう処理が施されているために表情に忠実であるとは言い難い。

測定系での抽出移動ベクトルの３次元化は、撮影環境や処理速度の問題などから非常に困難である。つまり、２次元で得られた結果を３次元のモデルに結び付ける、このような何らかの規則が必要となってくるであろう。

本研究では、先に述べた測定点数を、実際に増減させることにより、その表情の再現品質を検討し、適切な点数及び位置を考察する。

また、測定をすべて３次元で行ない、実際の表情表出時の顔表面の動きを、３次元的に捕えることにより、今後の合成系へのフィードバックを狙う。

第３章

マーカー位置の３次元計測

第２章で述べたとおり、本研究では各表情表出時の皮膚表面の動きを、３次元的に捕える必要がある。

本章では、この皮膚表面の動きの３次元計測手法を示す。

３.１　正側面画像を元にした３次元構造の計測[4][5]

本節では、本研究に適用した３次元構造計測の手法を解説する。

３.１.１　原理

本研究では、一般的に対象の３次元形状を得るために用いられる正側面画観測による手法を採用している。その概念図を図３－１に示す。

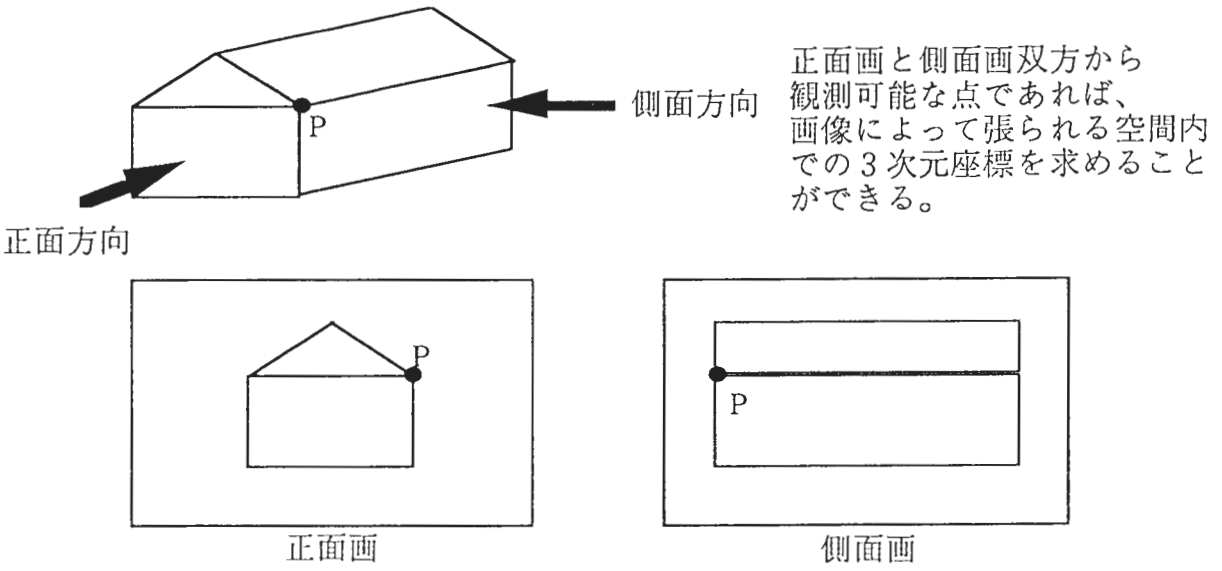


図３－１　正側面画観測による物体の３次元構造認識

カメラと対象の距離を無限遠と仮定し、２台のカメラを直角に配置した場合を考える。捕えられた対象の大きさが、２枚の画像中で等しくなるようにし、その画像中の形状点間の対応が取れているとすれば、その点P(px,py,pz)の空間座標は次式で与えられる。

$$P(px,py,pz) = (pfx , pfy , psx) \qquad (３－１)$$

ただし、pfx及びpfyは正面画に於ける２次元座標系での測定値、psxは側面画に於ける２次元座標系での測定値をそれぞれ表す。（次ページ図３－２参照）

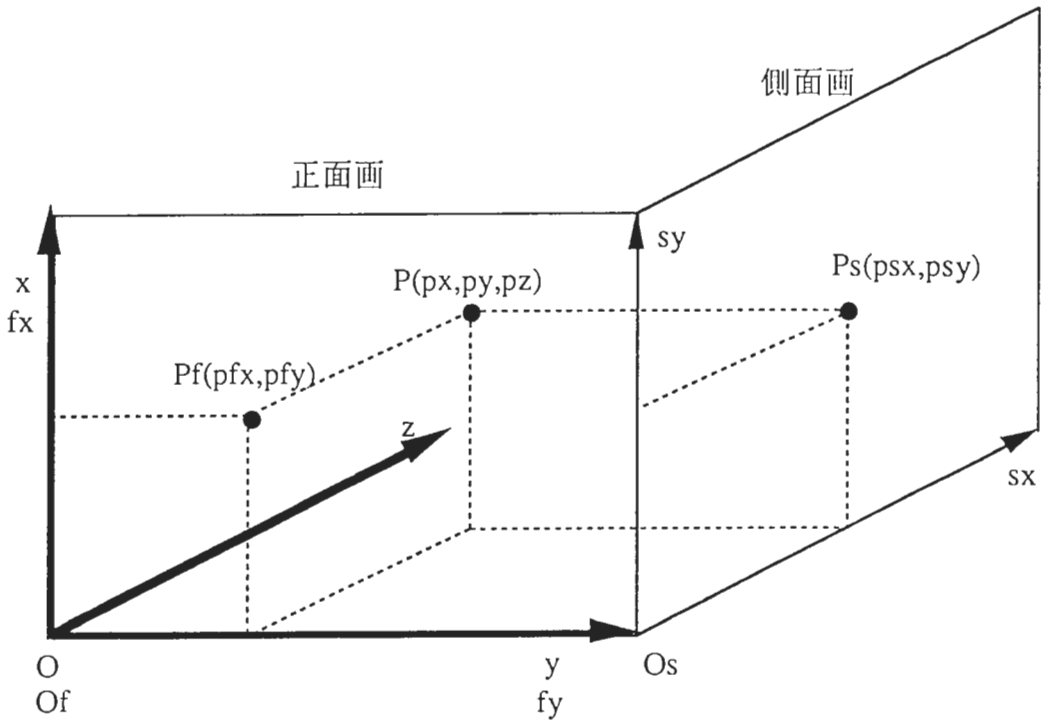


図３－２　２枚の画像によって張られる３次元空間

この時の空間 (x,y,z) は、２枚の画像によって張られる直交座標空間である。

３.１.２　カメラの画角を考慮した測定

実際の撮像装置で対象を撮影する際に、得られる画像は対象に対してカメラの視点を原点とした透視変換が施されたものである。これは、撮像装置には必ず画角というものが存在するからである。３.１.１節では、カメラと対象の距離を無限遠と仮定しているためにこの角度は無視できるが、実際の撮影系では、必ず距離がある有限値を取るため影響が無視できない。

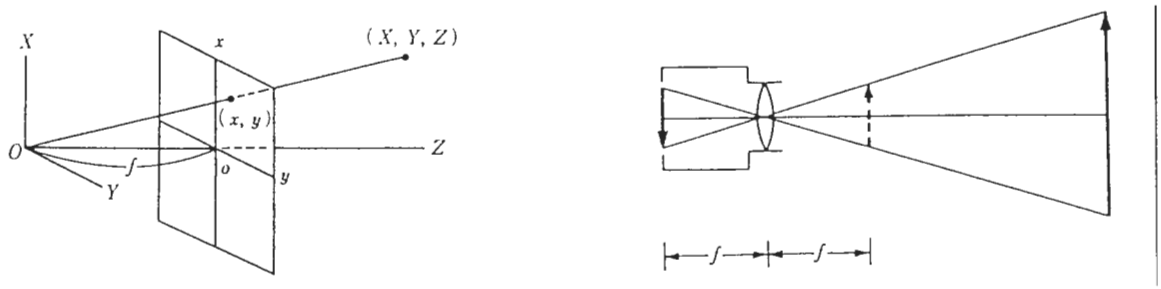


図３－３　原点Oを視点とする透視変換とカメラ撮像のモデル

次ページ図３－４に示す様に、平行投影（カメラ－対象間の距離を無限遠に仮定）の場合、画像中の点Pは画像と垂直な直線Pl上に存在することとなるが、透視投影変換（距離有限値）の場合、画像と垂直な直線Plに対してγなる角度を持った直線Pl'上に存

在することになる。このγの大きさは、カメラの持つ画角αと、測定点Pから画像中心oまでの距離、画像サイズoDから計算される。

$$\gamma = \alpha \frac{|\overrightarrow{oP}|}{|\overrightarrow{oD}|} \qquad (3-2)$$

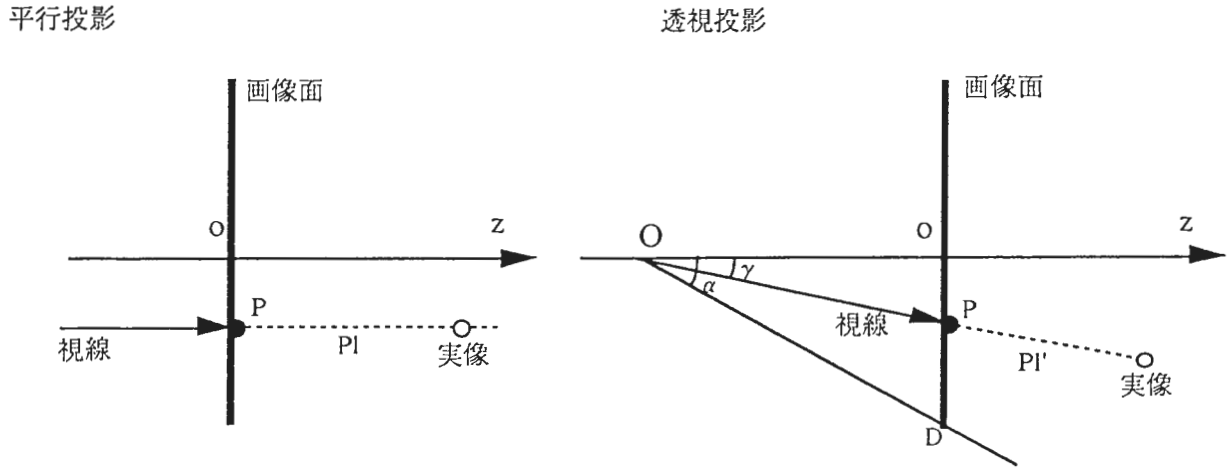


図３－４　平行投影と透視投影変換

この画角を考慮した３次元計測を行なった場合には、以下のように各座標値を算出する。まずx-z平面を考える。図３－５(a)のように空間をモデル化すると、

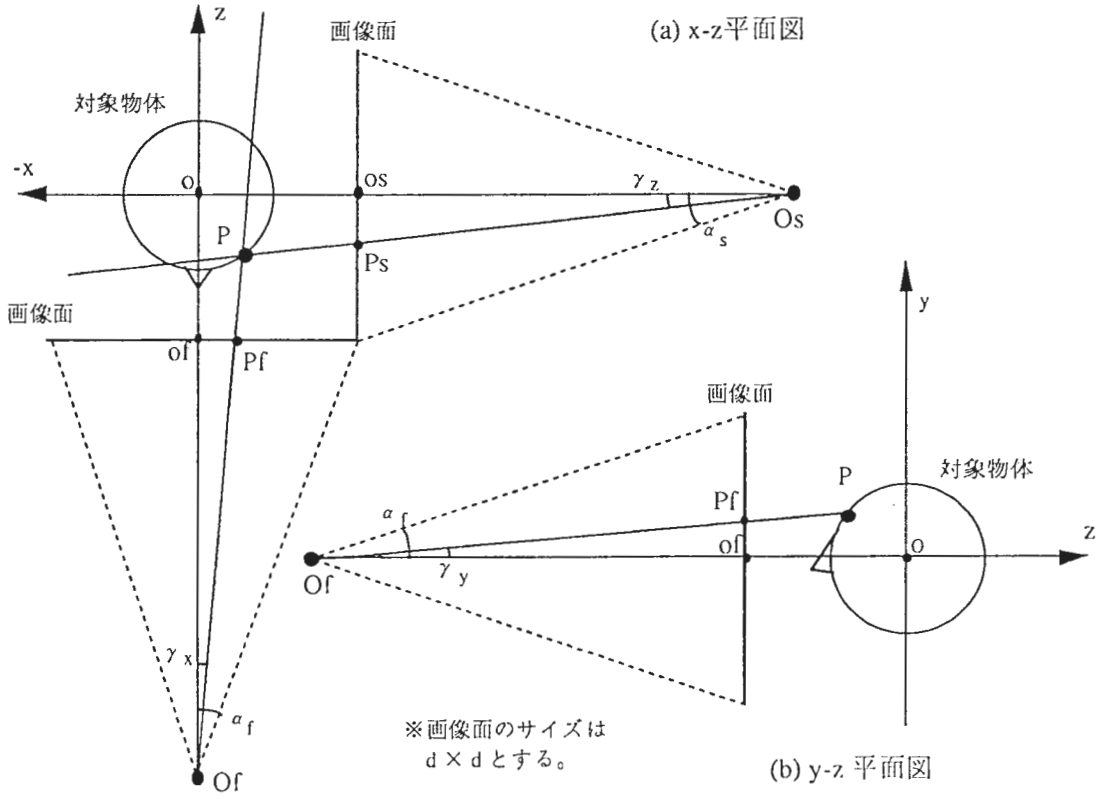


図３－５　測定系のモデル

点Pの座標は点Pfを通り、z軸に対して角度γ_xを持った直線と、点Psを通りx軸に対して角度γ_zを持った直線との交点の座標となる。またここで求められたz値を元に、y-z平面を考えれば、図３－５(b)のように描き表されP点のy座標が求められる。

カメラOfから撮影した画像でαから（３－２）式を使って求めたγのx軸成分をγ_x、同様にしてカメラOsからの画像より求めたγのz軸成分をγ_zとする。

点Pfを通り、z軸に対して角度γ_xを持った直線の方程式は、

$$x = z \tan \gamma_x + f_x - \frac{d}{2}\tan \gamma_x \qquad (3-3)$$

同様に点Psを通りx軸に対して角度γ_zを持った直線の方程式は、

$$z = x \tan \gamma_z - s_z - \frac{d}{2}\tan \gamma_z \qquad (3-4)$$

ここで、

$$|\overrightarrow{of\ Pf}| = f_v \ , \ |\overrightarrow{os\ Ps}| = s_v$$

とした。ゆえに交点のx座標は、

$$x = \frac{f_x - \tan \gamma_x\left(s_z + \frac{d}{2}\tan \gamma_z + \frac{d}{2} \right)}{\left(1 - \tan \gamma_x \tan \gamma_z \right)} \qquad (3-5)$$

となり、更に（３－４）式よりzが求められる。

図３－５(b)において、

$$|\overrightarrow{Pf\ of}| = fv$$

とおけば、点Pfを通り、z軸に対してγ_yの傾きを持った直線の方程式は、

$$y = z \tan \gamma_y + f_y + \frac{d}{2}\tan \gamma_y \qquad (3-6)$$

となり、（３－４）式で求めたzを代入することによって交点のy座標が求められる。

ここで各γは、α、dとf_x, f_y, s_zによって与えられることから、測定点Pの３次元座標は各画像内に投影された座標値と画像サイズそして画角αのみから求めることが可能ということになる。

３.２　基準ベクトルによる座標変換

表情の分析には各点の無表情時の位置と、表情表出時の位置との差分である動き情報が必要である。

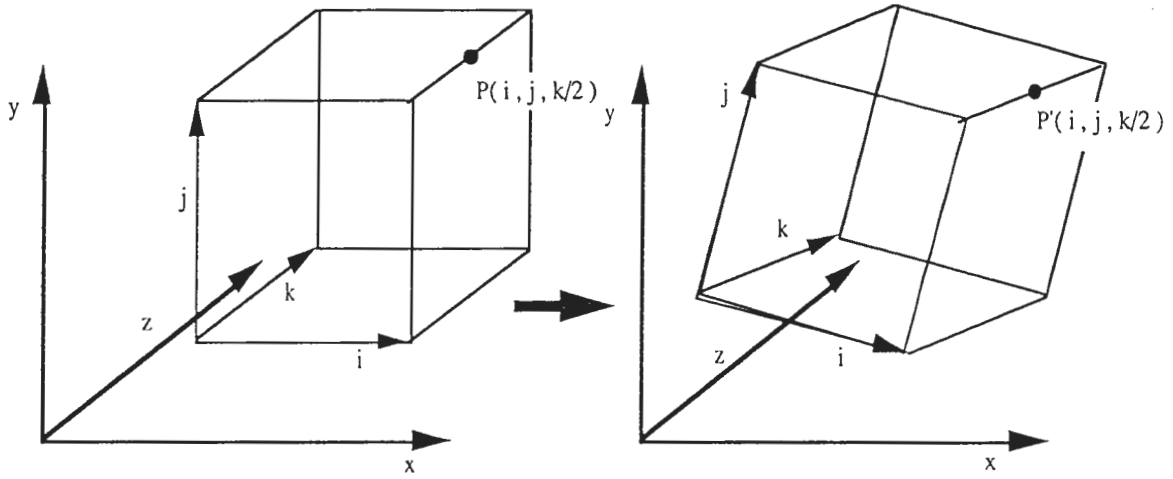
３.１.２の手法で、各測定点の３次元座標を得ることができた。しかし、この座標値から直接有用な動き情報を得ることは困難である。なぜなら、この測定系ではカメラが

外界に固定されているために、測定された点の動きベクトルには対象である頭部自体の動きも加わっているためである。

つまり測定された動きベクトルから何らかの手法を用いて、頭部の動きベクトルを減ずる必要がある。

その１つ受動的な方法として、頭部の動きベクトルを別に求める手法がある。本研究ではこの手法を応用し、対象の形状変化に左右されることのない、相対位置不変の測定点座標を元に、各測定値を座標変換する手法を取る。

具体的には、対象物体上に、ローカルな動きの影響を受けない、互いに交差する３次元ベクトルを３本決定し、他の任意の測定点の座標を、これらのベクトルで張られる空間に座標変換する。つまり、各ベクトル成分の加重線形和の形で表すのである。



対象物体が、絶対座標系に対してダイナミックにその位置を変えたとしても、対象物体内（i,j,k）の座標系で表された座標は影響を受けない。

図３－６　絶対座標系と対象物体内相対座標系

頭部に於てのこれら基準点として、鼻の頂点A、額中心最上部の点B、耳の外輪点Cを設け、基準ベクトルとして、$\vec{AB}$、$\vec{AC}$そして、$\vec{AB},\vec{AC}$の外積ベクトル$\vec{D}$を用いている。ただし、$\vec{D}$の大きさは、ベクトル$\vec{BC}$の大きさとしている。

こうして与えられた各ベクトルは、１平面とそれに垂直な軸として定義されているために、実空間上の任意の点を表すことが可能である。

実際の基準点を次ページ図３－７に示す。

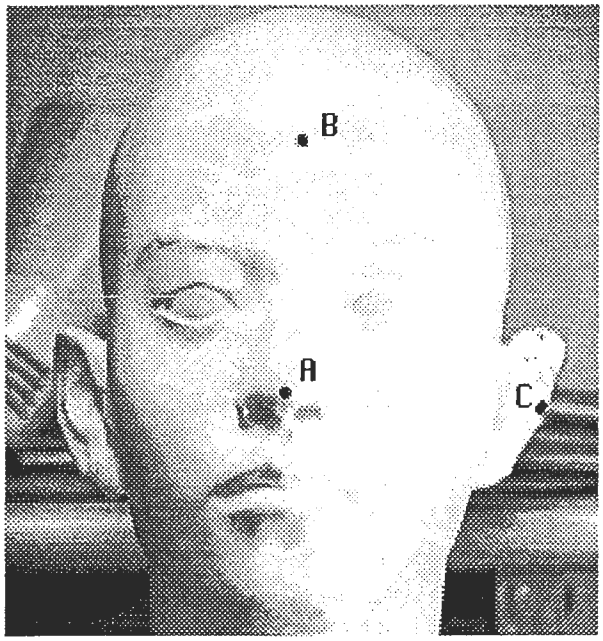


図３－７　顔部に設定した基準点

３.３　実験例

３章で説明した各手法を用いて実際に実験した結果を示す。

３.３.１　キャリブレーション

撮影実験に先だって、カメラの画角及び焦点位置の測定を図３－８の原理で行なった。

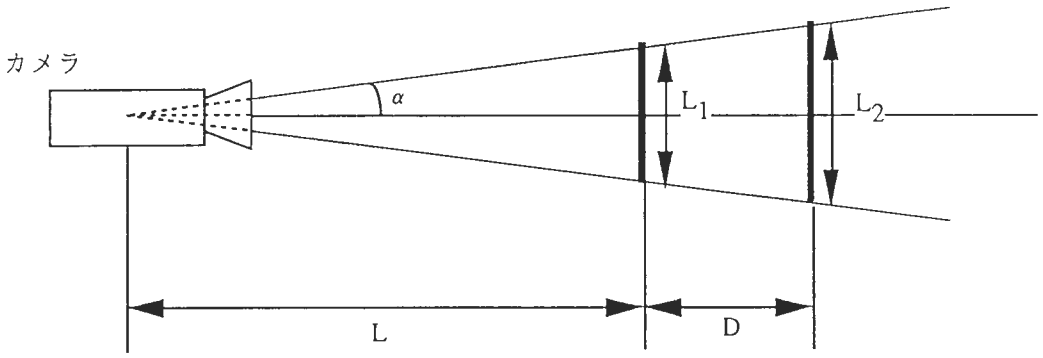


図３－８　カメラの画角及び焦点位置の測定

まずL_1の横幅を持つ物体を、カメラの光軸上に配置し、画面全体に対象が映り込むようにカメラと対象の距離を調節する。

次に$L_1<L_2$となる様な横幅を持つ物体に対しても同様の手順で調節する。このときカメラは固定である。求められたL_1L_2間の距離をDとし、カメラの焦点位置からL_1までの距離をLとすれば、

$$L = \frac{L_1 D}{(L_2 - L_1)} \qquad (3-7)$$

でLを求めることができる。そして画角αは、

$$\alpha = \tan^{-1}\frac{L_1/2}{L} \qquad (3-8)$$

となる。

実験環境の下で得られた値は、L = 3.1112(m) , α = 0.047126(rad)であった。

3.3.2　測定環境

２台のカメラの焦点位置から仮想画面までの距離（図３－５でのOf-os間,Os-os間の距離）は、３.３.１で行なったキャリブレーションでのLと等しい。

また、それぞれのカメラの持つ画角αfとαsも３.３.１でのαに等しい。

∠Of o Osは正確に直角になるようセッティングを行なっている。

今回の実験に関しては、２画面内に於ける対象点間の対応付けは、マニュアルで行なうため、照明条件は厳密に決定せず、室内環境光下で撮影を行なった。

3.3.3　測定実験

皮膚表面動き測定のためのマーカーは、顔の片面（左半面）で基準点込み56箇所貼付した。ここでは顔の表情は左右対称であると仮定し、半面の測定のみ行なっている。

図３－９　測定のためのマーカー位置

被験者には正面に対して約20度の傾で、点oに頭頂部が来るように座らせる。これは正側面画においての測定点の対応付けを容易にするためである。

無表情＞怒り＞悲しみ＞喜び＞驚き＞嫌悪＞恐怖＞無表情という表情変化を１５秒で被験者に表出してもらい、これを1/15秒単位でフレームメモリに記録した。データサンプル数は正面・側面合わせて、１５（秒）×１５（枚）×２（方向）で４５０フレームである。

また、正確に正面を向いた状態での画像も、正規化用に撮影した。

最後に、画面内における移動量（Pixel単位）をメートル単位に変換するための基準となる３次元構造をサイバーウエアで取得する。

サイバーウエアによって測定された基準ベクトル$\overrightarrow{AB}$(cyber)の大きさと、本実験によって得られる画面内座標の$\overrightarrow{AB}$(stereo)の大きさの間には、

$$a = \frac{\overrightarrow{AB}(\text{cyber})}{\overrightarrow{AB}(\text{stereo})} \qquad (3-9)$$

なる関係が成り立ち、このaを求めることによって、画像内での移動量を実空間での移動量に変換可能となる。

以上の手法を用いて、各フレームについて測定点の座標値を算出、そのフレーム内の基準ベクトルにより座標変換を施し、その値を正規化用画像の基準ベクトルに乗じて画面内での正確な３次元座標に変換する。そして、１フレーム目を基準とした各フレームでの測定点の移動量を算出、先に求めたaによってメートル単位に変換し、実際の３次元的な移動量となる。

無表情が数フレーム連続するシーケンス内での、各測定点における移動ベクトルの値を平均したところ、約0.24(mm)となり、最大移動量-10.115(mm)から+9.40(mm)と比較して±1.2％の測定精度があることが分かった。

本実験での結果は付録１にその一部を添える。

第４章

表情再現

本章では先の計測で得られた測定点の移動量を元に、モデルを変形する手法について述べる。

４.１　測定点とモデルとの対応

ここで対象となるモデルは、サイバーウエアで取得されたものに、顔各特徴部位（目・口等）の意味づけを行なったものであり、ノード数・ポリゴン数は任意である。

測定時にサイバーウエアで取得したモデルでは、テクスチャにマーカーが現われており、また顔各部位への意味づけもなされていないため使用できず、合成時にはあらかじめ作成したモデルを用いることになる。

よってこのモデル上での動きの基準となる制御点（測定点）の位置を求める必要がある。本来ならば、この合成に使用するモデルと分析時に取得したモデルの間で何らかの対応付けを行ない、制御点の座標を決定するのが正であるが、今回の実験では制御点の座標決定はマニュアルで行なっている。

また測定が顔の左半分のみであったため、仮定に基づき頭部の中心線を軸にした対称移動によって残った右半面にも制御点を設けている。

こうして得られたモデル上の全制御点（９８点）を図４－１に示す。

図４－１　制御対象モデルに与えた制御点

4.2　制御点以外の格子点の移動規則

第３章で求められている移動量は制御点だけのものである。実際のモデルには制御点以外にも多数の点があり、これらの点を制御点と協調して移動させなければ良好な再合成結果を得ることは出来ない。

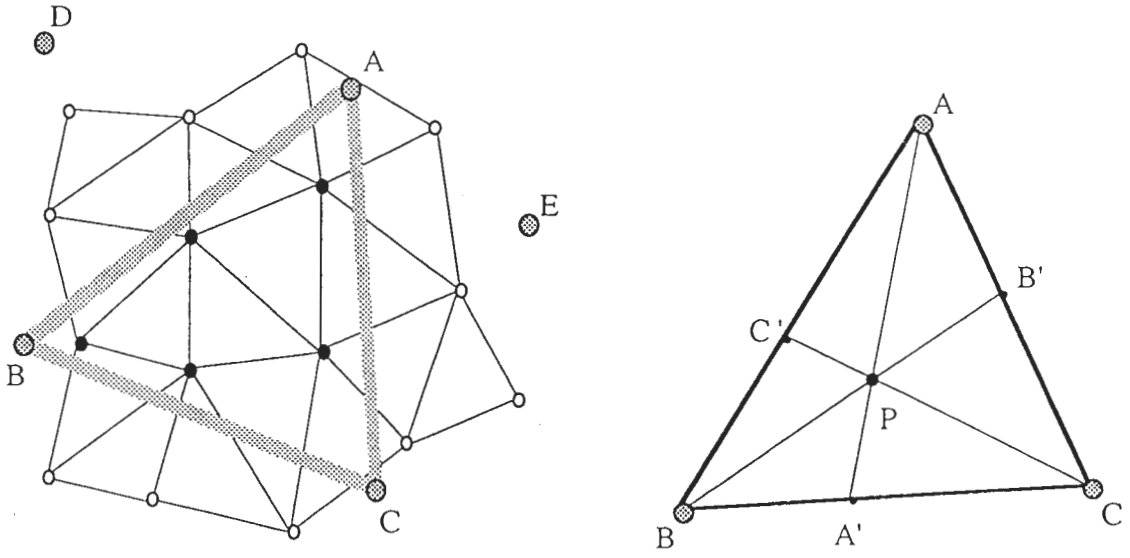
本章では特化したアルゴリズムを用いるのではなく一般的な規則でモデルの格子点を移動させ、自然な顔表情の再現をする手法を検討する。

4.2.1　制御ポリゴンによる手法

頬や、額といった比較的広範囲に渡って同一の動きを示す領域には、制御ポリゴンによる移動量の重み付けを行なう。

制御ポリゴンとは、３つの制御点により構成される３角形ポリゴンである。wavefront形式の３次元構造データに於て、テクスチャ上の座標が、このポリゴン内にある点は、このポリゴンを形成する３つの制御点のみから移動の影響を受ける。

３角形内部の点がそれぞれの頂点に当たる制御点からどの程度の影響を受けるかは内部の位置に依存する。（図４－２参照）



３角形ABC内部に位置する点（ ● ）は制御点A,B,Cのみから移動の影響を受ける。DやEの動きには協調しない

(a)　　(b)

図４－２　制御ポリゴンによる移動ルール

図４－２(b)に於て格子点Pが受ける動きの影響（重み）は次のように計算される。
制御点AからPを通り辺BCに下ろした足をA'、同様にB,Cからの足をB',C'とする。
P点のA点に対する影響の重みPWaは、

$$PWa = \frac{|\overrightarrow{PA'}|}{|\overrightarrow{AA'}|} \qquad (4-1)$$

同様に、PWb, PWc は、

$$PWb = \frac{|\overrightarrow{PB'}|}{|\overrightarrow{BB'}|} \qquad (4-2)$$

$$PWc = \frac{|\overrightarrow{PC'}|}{|\overrightarrow{CC'}|} \qquad (4-3)$$

となる。そして、PWa , PWb , PWcの間には、

$$PWa + PWb + PWc = \frac{|\overrightarrow{PA'}|}{|\overrightarrow{AA'}|} + \frac{|\overrightarrow{PB'}|}{|\overrightarrow{BB'}|} + \frac{|\overrightarrow{PC'}|}{|\overrightarrow{CC'}|} = 1 \qquad (4-4)$$

の関係がある。

実際の点Pの移動ベクトル$\overrightarrow{dP}$は、各制御点の移動ベクトル$\overrightarrow{dA},\overrightarrow{dB},\overrightarrow{dC}$より、

$$\overrightarrow{dP} = PWa \cdot \overrightarrow{dA} + PWb \cdot \overrightarrow{dB} + PWc \cdot \overrightarrow{dC} \qquad (4-5)$$

のように計算される。

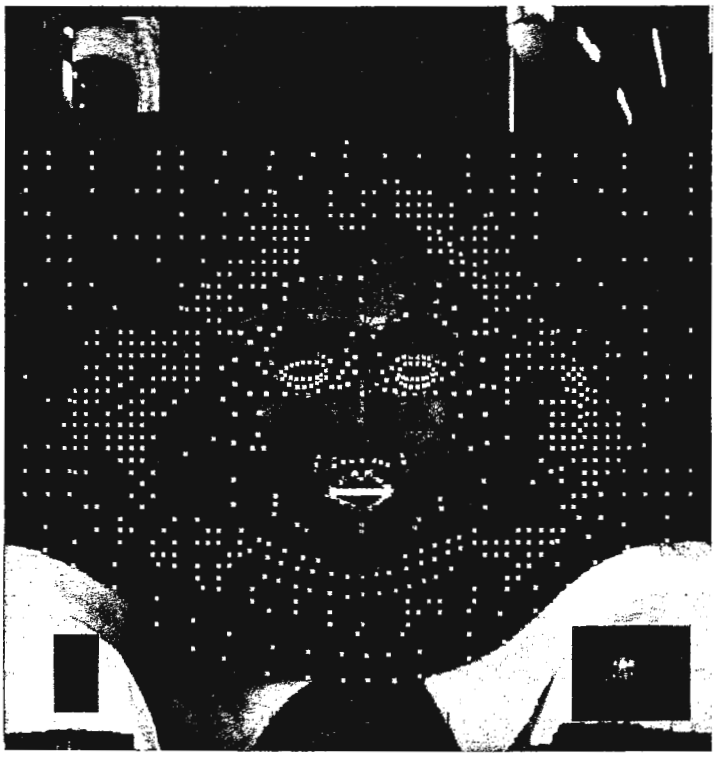
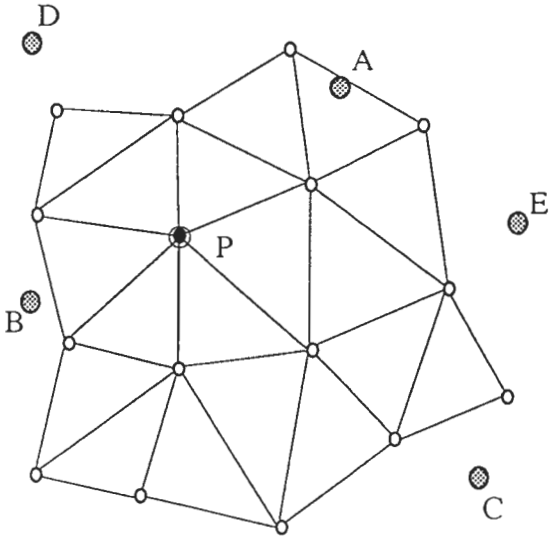


図４－３　制御ポリゴンによる重み計算を行なう箇所
（赤線で区切られた領域。緑の点が制御点）

4.2.2　重みを積算する手法

例えば口の領域では、制御ポリゴンによって重みを設定した場合、上唇と下唇が同じポリゴンに含まれ、本来の動きとはまったく異なる変形が加わってしまう。このように影響を受ける制御点が数多く、複雑な動きを示す箇所（目、口等）に於ては、単純に制御点と対象点の距離に反比例するように影響の大きさ（重み）を決定する。



点Pは周辺に存在する全ての制御点の影響を受ける。
影響の重みは、近いもの程大きく、遠くなるにしたがって減衰する。
制御点の影響範囲外では重みは0である。

左の例では、点Pは制御点Bから受ける影響が最も大きく、制御点Cから受ける影響が最も小さい。

図4－4　距離による重み

1つの制御点が影響を与える範囲は3次元的な球形で設定し、その中心部つまり制御点そのものの上に点が在った場合1に、球の表面上で0になるように線形に重みを設定する。

この手法では複数の制御点から影響を受ける点では重みの総和が1以上になる場合が考えられる。このように重みの総和が1を越えた場合には、1になるように正規化を行ない各制御点からの影響を調節する。

制御点の影響範囲である球の半径をr、点Pに影響する各制御点をCi (i = 1,2,⋯,n)とすれば、一つの制御点が与える影響の重みは、

$$PWCi = \frac{r - |\overrightarrow{PCi}|}{r} \qquad (4-6)$$

となる。これを影響する全ての制御点について計算し、各制御点の移動ベクトルを$\overrightarrow{dCi}$とするならば点Pの移動ベクトル$\overrightarrow{dP}$は、

$$\overrightarrow{dP} = \sum_{i=1}^{n} \frac{PWCi}{sum} \overrightarrow{dCi} \qquad (4-7)$$

ただし

$$\begin{cases} \sum_{i=1}^{n} PWCi > 1 & sum = \sum_{i=1}^{n} PWCi \\ \sum_{i=1}^{n} PWCi \leq 1 & sum = 1 \end{cases} \qquad (4-8)$$

また、唇やまぶたのように距離が近くても、上下別の動きをする点の場合、マニュアルにて重みを与えない制御点を決定する必要がある。

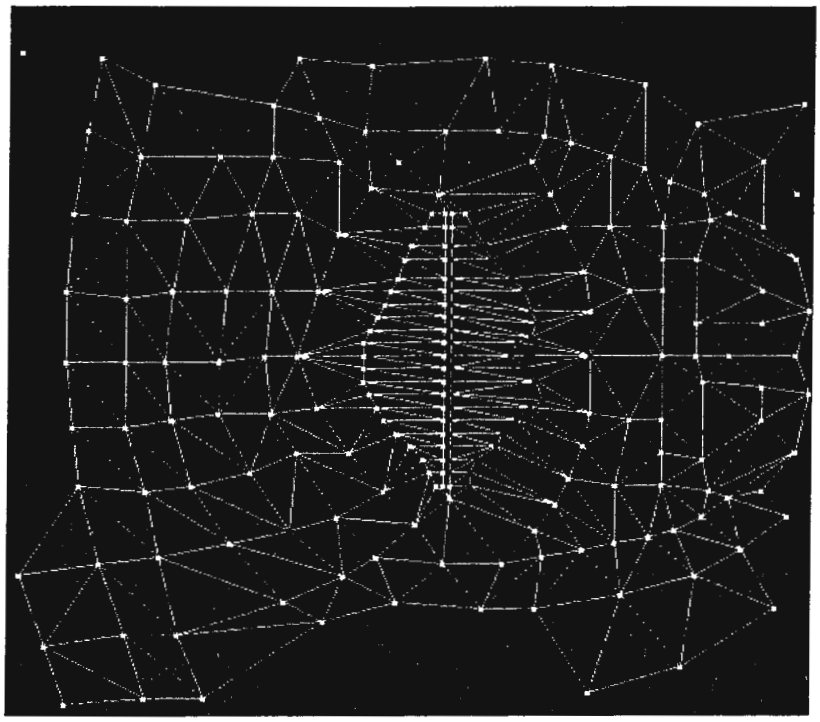


図４－５　特殊な場合（唇の例）

図４－５に於て、緑・青の点が制御点、黄・赤の点が格子点を表す。

青の制御点が、移動の影響を与える領域にある格子点を赤で示した。実際には青の制御点の影響範囲は唇全体に及ぶが、上唇と下唇は独立に運動すると仮定できるので（唇の両端点付近の動きにはこの仮定は適用できない）、マニュアルにて、下唇を構成する格子点を影響範囲から除外している。

第５章

測定箇所と表情再現品質の関係の検討

本章では、第３章、４章で述べてきた各手法を用いて再合成した画像の品質を、測定箇所との関係を中心に評価・検討してゆく。

５.１　顔全体に対する再現品質の検討[6]

本節では、制御点（測定点）の数を任意に増減した場合に表情の再現品質がどのように変化するかを検討する。

５.１.１　再合成結果

撮影実験にて被験者に表出してもらった６つの表情の内、喜びと驚きと悲しみに関して、測定箇所数別にまとめた。

測定箇所数は１０、３０、５０、７０、９８の５種類で、その各々の測定点位置を図５－２から５－６に示す。測定箇所の設定は、１０（最小）と９８（最大）の時を基準に、全ての領域について均等に点数が増減するように定めた。

全ての再現品質の基準となる、無表情時（各測定点移動量０時）の再合成画像は下図である。

図５－１　無表情時の再合成画像

図５－２　測定点数１０の場合の測定箇所

図５－３　測定点数３０の場合の測定箇所

図５－４　測定点数５０の場合の測定箇所

図５－５　測定点数７０の場合の測定箇所

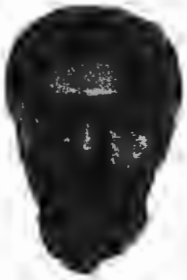
図５－６　測定点数９８の場合の測定箇所

以上の測定箇所の移動量を実画像から算出し、モデルの変形を行なって画像を再合成した結果を図に示す。

悲しみの表情を再合成したものを図５－７に、喜びの表情を再合成したものを図５－８に、同じく驚きの表情を図５－９に示す。

同時にそのフレームでの原画像も示す。

(a) 測定点１０



(b) 測定点３０

(c) 測定点５０

(d) 測定点７０

(e) 測定点１０

(f) 測定原画像

図５－７　悲しみの表情の再合成例

(a) 測定点１０

(b) 測定点３０

(c) 測定点５０

(d) 測定点７０

(e) 測定点９８

(f) 測定原画像

図５－８　喜びの表情の再合成例

(a) 測定点１０

(b) 測定点３０

(c) 測定点５０

(d) 測定点７０

(e) 測定点９８

(f) 測定原画像

図５－９　驚きの表情の再合成例

5.1.2　各表情時の再現品質の検討

悲しみの表情：

悲しみの表情の特徴は主に、目と眉の領域に集中している。

眉は全体的に下がり、目は細められる。

再合成結果を見ると、測定点数１０のもの以外はどれもほぼ変わらない結果となっている。

原画像では、目はほぼ完全に閉じられているが、再合成画像では閉じていない。これは目の測定点の設定と格子点変形規則の影響である。

目上部の測定点は、まぶたと眉の中間地点に与えられている。現在使用している格子点変形規則では、この測定点での移動量が最大になるように設定されているが、まぶたの領域では、移動量最大の点はまぶた最下端部であり、測定点のある箇所ではない。

このため、実際には最も移動量の大きいまぶた最下端部が測定点から距離を置いていることで、測定点の移動量よりも少ない移動量になってしまっている。

またこのことは目の下部でも言えることであり、この２つのことから目が完全に閉じないという結果になってしまっている。

まぶたの動きの再合成にはもう一つの問題点がある。まぶた自体の収縮が大きい点である。特に二重まぶたの人物の場合などは、無表情時に現われていない奥まぶたが目の開閉に携わる関係上、自然な再合成は困難である。

現在の測定点の位置では、点数の増減はあまり意味のないことがわかる。

表情全体で考えると悲しみの表情を再現するには、５０点の測定で十分ということになる。

喜びの表情：

喜びの表情では、眉が微少に下がり、頬が持ち上げられることによって目が細められる。またその影響で唇の両端が上がり、下唇が下がる。測定画像の場合顎は下がっていない。

再合成画像を比較すると、悲しみの表情とは違い、測定点数によって有為な差が認められる。

測定点数１０では、目の下部を制御する点が無いために、頬が上がり目が細められるという特徴が得られていない。口領域に関しても測定点数が少ないために、忠実な再現には至っていない。

測定点数３０では、目や口領域に関して、測定点の設定がアンバランスになっている為か、まったく違う動きをしてしまっている。

それ以上の測定点数では在る程度の品質で合成画像が得られているが、悲しみの場合と同様に目の動きが小さい。また頬が上がることによってできる鼻両翼のしわが再現できず、不自然感がいなめない。口内部に関しても、歯がモデル化されていないために不自然である。

ここでも５０～７０点以上の測定点数で在る程度の表情再現が可能であることが解る。

驚きの表情：

眉が持ち上げられそれに伴って目が見開かれるという特徴がある。また顎が下がり、顔下部の筋肉が弛緩する。

再合成画像を見ると測定点数に応じて表情の再現性が変化していることが確認できる。

測定点数１０の場合は、殆ど無表情時と変化が認められない。３０点以上の測定点数では徐々に表情が変化している。

原画像での特徴である、唇を突き出すような動きが、再合成画像では表現されていない。これは測定点の位置と唇の動きに問題があるからである。

唇の測定点は、唇の上端と下端に設定されている。これに対して、唇を突き出すといった動きの場合、唇の口内側が外側に突出する形で表現される。

これは、まぶたの場合と同様に無表情時に見えていない部分の出現であるため再合成は困難なのである。唇の外郭は表現されていても、内郭は表現されていないために違った印象を与えてしまうのである。

測定点数５０程である程度の再現品質は得られるが、点数を更に増やすことによってより高い品質を得ることが可能となる。

５.２　顔各部位に対する再現品質の検討

本節では先の合成結果から得られる知見をもとに、顔各部位への測定点数の割り振り、測定位置の検討等を行なう。

５.２.１　顔上部

額の動きに関しては、モデルにテクスチャを貼付するだけの合成法を用いる限り、点の増減による再現品質の相違は見られていない。額領域の顕著な動きとしてしわが挙げられるが、この合成法では表現できないからである。

また、額の筋肉の動きは非常に平面的かつ移動量の最大の点が眉付近になるため、実験で用いた距離による重み設定の移動規則がそのまま利用可能である。

以上のことから、額には測定点を設ける必要は無いと思われる。

ただし、今後しわの生成を考えた場合にはこの限りではない。

眉領域については、表情変化に非常に重要に関わってくる部位であるために、正確な表現が必要である。実験結果を考察するならば、眉の上部で最低３点の測定箇所が必要であろう。これは測定点数５０点の場合に眉領域に割り当てられた数字である。

５.１節で述べたとおり目領域の表現には、実験で用いた測定点箇所や変形規則では対応しきれていない。現在のようにマーカーを貼付し、その移動量を追跡する手法と移動規則では、正しく表現するのは不可能と思われる。現在求められうる移動量を元にして、自然な表情合成を行なうためには、目領域に特化した移動規則を定める必要があるであろう。

本手法で高い品質を表現するには、最低で４箇所必要であると思われる。

以上述べたことを考慮すると顔上部には総計１４点の測定点を設けるのが良いと思わ

れる。

5.2.2　頬・鼻領域

頬領域に関しては、測定点を増加させてもそれほどの品質向上は見られなかった。

頬の動きは、おおよその中心部が皮膚表面から盛り上がり、周囲がそれに引かれる特徴があるため、この中心部に測定点を設け、その影響範囲を最適に設定すれば、ある程度の品質が得られるものと考えられる。もしくは目の下部の点と鼻横の点を結んだ制御ポリゴンを利用する手法も考えられる。

鼻に関しては本来表情とは直接関係なく移動量も極小であるため、測定点を設ける必要は無いと思われるが、鼻の付けね部分は顔上部の動きの基準点（不動点）と成り得る点であるし、鼻孔横は頬や口の動きに関連してかなりダイナミックに移動する。そのために最低でも顔両側で３点は必要であろう。

また、鼻孔横から口の両端にかけて生成されるしわに関しての動きを抽出するためにこの上にも測定点が必要である。

頬の点を顔両側で４点と考えればこの領域に関しては９点程度必要であることになる。

5.2.3　顔下部領域

唇に関しては５.１.２節で述べた様な問題点が挙げられるが、全体的な再現品質を捕えた場合に８点程の測定点数で望む品質が得られていると言える。これは全体の測定点数５０点の場合の唇に与えられている数字である。

しかし、この数で高い品質の再現を目指す場合、目領域と同様に特別な変形規則を設ける必要がある。もしくは、唇内郭の動きを抽出する何らかの手法を用いることになるであろう。

現在の変形規則および画像合成手法では、口を尖らせるといった動作や、唇をめくるなどといった動作は、表現不可能である。

顎の動きは複雑であるが、本実験においては測定点数の増加に伴う品質の明らかな向上は見られなかった。これは顎の動き自体が小さかったためと思われる。本来の顎周辺の皮膚の動きは平面的ではなく、顎の骨にそって曲面状の動きを示すものであるために、少ない測定点数に基づく現在の変形規則では再現が困難なものである。

しかし、本実験に用いた程度の表情の強度であれば、明らかな品質劣化には繋がらないということが判明した。

顎領域には、全体で測定点数５０の場合に割り当てられた点数である３点で十分と考えられる。

ただし、これ以上に強度の強い表情が入力されたときには合成画像の顎部分が歪む可能性がある。

次ページ図５－１０にこうして求められた測定位置を記した。

図５－１０　品質を考慮した測定位置（３４点）

第６章

まとめ

本研究の一つの目的であった、表情表出時の顔皮膚表面の動き測定では、２台のカメラを用いたステレオ計測によって、３次元的な測定が可能となった。

測定実験による皮膚表面の３次元動きベクトルは、最大の動きに対する誤差を１％程度に押さえており、今後の表情分析や合成において十分に利用価値のあるデータということができる。

この測定方法の確立によって、従来不可能だった表情の時変化を追跡することが可能となり、人間の表情表出自体の分析にも活用することを検討中である。

もう一つの目的である表情再合成の品質と動き測定点数との関連の検討では、最大９８点という測定点から１０点までという幅広い点数による合成を試み、最適な測定点数は３５点前後であることを明らかにした。

しかし今回は、筆者の主観による分析のみであり、複数被験者による主観評価を実施していないため、有意であるとは言い難い。

また、表情表出時の皮膚表面の動き測定に於ても、被験者は１名であり多分に被験者個人の特徴が現われていると考えられる。今回合成に用いたモデルは被験者本人のモデルであったために問題はなかったが、今後一般的な解析を必要とした場合、複数被験者による測定が不可欠になると思われる。

今後の課題としては上記に挙げた、

○複数被験者による表情測定

○主観評価実験

に加え、

○動画像からの測定点自動追跡

○顔の特徴部分に於ける移動規則の検討

○ActionUnitに基づく測定点位置の設定

○筋肉の動きを考慮した移動規則の検討

○個人性の吸収

等がある。

今回行なった実験において、最大の測定点数である９８点の場合でも、その再合成画像は原画像と比べ大きな品質の劣化が見られた。

通信などに用いるための様々な規制を無くし、表情の高品質合成だけを考慮した場合でも、再合成画像を原画像に近付けるのは容易ではない。

この分野での「表情の自然性」の難しさを再認識させられる研究であった。

謝辞

このような研究の場を与えてくださった成蹊大学森島繁生助教授に、研究及び実験の環境整えていただき、研修期間中様々な面で筆者を導いて下ださった北村泰一氏に、そして長時間に及ぶ実験の被験者を快く引受てくださった越知武氏に深く感謝致します。

また、お世話になった通信システム研究所知能処理研究室の研究員の皆様に厚くお礼申し上げます。

参考文献

[1] 大谷淳、北村泰一、竹村治雄、岸野文郎
"臨場感通信会議における３次元顔画像の実時間表示"
信学技法　HC92-61 pp23-28 , 1993.1

[2] 小林宏、原文雄
"ニューラルネットによる人の顔の混合表情の認識"
信学技法　HC91-56 pp17-22 , 1992.3.10

[3] 崔、原島
"顔の３次元モデルに基づく表情の記述と合成"
信学会論文誌、-Vol.J73-A,No.7,pp1270-1280, 1990

[4] 金谷健一
「画像理解－３次元認識の数理－」
森北出版株式会社　1990.5.24　第１版

[5] 谷内田正彦編
Maruzen Advanced Technology ＜電子・情報・通信編＞
「コンピュータビジョン」
丸善株式会社　1990.3.31

[6] P.Ekman and W.V.Friesen
"Facial Action Coding system"
Consulting Psychologists Press, 1977.

表情表現を考慮した顔特徴点抽出に関する検討

- 表情変化による顔皮膚表面の時間的変位の計測箇所と
表情再現品質の関係の基礎検討 -

(付録)

付録. 1　表情表出時の顔表面皮膚の動き実測データ例
付録. 2　顔に貼付したマーカー位置及び番号
付録. 3　顔画像再合成までの一連の流れ
付録. 4　ソースリスト
付録. 5　MOディスクの内容

坂口　竜己　TATSUMI SAKAGUCHI
大谷　淳　JUN OHYA
岸野　文郎　FUMIO KISHINO

1993. 9. 2

ATR通信システム研究所

付録.1

表情表出時の顔表面皮膚の動き
実測データ例

（測定点が１３点、フレーム数６０の予備実験のデータ）

本実験のデータは別添

exp2'

正面を向いた時の移動量にかんさんしたもの.

Aug 8 1993 13:46:45 **movement2** Page 1

```
 1 61 13
 2
 3 mouth_side
 4 1        23.688002       -47.213672       22.705186
 5 2        -0.334867       -0.243690        1.207159
 6 3        -0.194476       -0.180177        1.180458
 7 4        -0.348303       -0.361776        1.164305
 8 5        -0.435171       -0.202147        1.171320
 9 6        -0.412268       -0.275983        1.168263
10 7        -0.287384       -0.273756        1.051396
11 8        -0.159168       -0.293796        1.022363
12 9        -0.175640       -0.401366        0.944186
13 10       -0.179365       -0.339480        1.167285
14 11       0.263426        -0.677050        0.966682
15 12       0.036870        -0.450169        0.778277
16 13       0.377544        -0.496732        0.670143
17 14       0.584786        -0.729130        0.227441
18 15       0.485015        -0.607552        0.265092
19 16       0.711761        -0.902924        0.186848
20 17       1.067815        -0.954323        0.011840
21 18       1.353367        -0.974700        -0.445529
22 19       1.163280        -1.316288        -0.799491
23 20       1.522668        -1.452548        -0.736616
24 21       1.300472        -1.345081        -0.864526
25 22       1.532829        -1.861134        -0.847643
26 23       1.251926        -1.656795        -0.882492
27 24       1.186217        -1.753769        -1.002240
28 25       1.150846        -1.727181        -0.892371
29 26       1.023033        -1.792915        -0.667726
30 27       1.199421        -1.633740        -0.797738
31 28       0.689697        -1.421629        -0.506718
32 29       0.657312        -0.851464        -0.390639
33 30       0.619193        -0.857359        -0.120821
34 31       0.644636        -0.781616        0.050472
35 32       0.457118        -0.399146        0.217647
36 33       0.470552        -0.404970        0.129156
37 34       0.428237        -0.647709        -0.101674
38 35       0.382849        -0.456933        0.053345
39 36       0.433942        -0.605907        0.092555
40 37       0.589401        -0.586280        -0.012607
41 38       0.688485        -0.023798        0.621798
42 39       0.753234        1.693046         1.958329
43 40       0.814842        3.003195         2.990755
44 41       2.641137        5.762905         5.847885
45 42       4.052714        7.467256         7.285485
46 43       5.091155        8.140863         8.002243
47 44       5.557777        8.327280         8.482678
48 45       5.580754        8.689918         8.747416
49 46       5.758515        8.314926         8.956734
50 47       5.997477        8.069964         9.767856
51 48       5.615662        8.320279         9.974349
52 49       5.593199        8.218125         10.351541
53 50       5.710299        8.284380         10.588326
54 51       5.678529        8.311713         10.481157
55 52       5.711100        8.133366         10.401600
56 53       5.237249        7.759210         10.097305
57 54       4.955421        7.674800         9.909513
58 55       4.412011        6.806712         9.411460
59 56       3.853323        6.465673         8.897752
60 57       2.862878        4.990441         7.414981
61 58       2.210251        3.931098         6.177329
62 59       1.674052        2.442871         4.595058
63 60       1.359989        1.748210         2.913183
64 61       0.772458        1.746277         2.609557
65
66 mouth_bottom
```

Aug 8 1993 13:46:45 **movement2** Page 2

```
 67 1        3.638276        -52.068234       6.215804
 68 2        -0.443508       -0.274154        0.939750
 69 3        -0.312355       -0.093540        1.054833
 70 4        -0.297612       -0.181739        1.017381
 71 5        -0.638364       0.077528         0.986332
 72 6        -0.326311       -0.025991        1.049073
 73 7        -0.370993       -0.044818        0.946489
 74 8        -0.369630       -0.013840        0.981481
 75 9        -0.442009       -0.291517        0.908886
 76 10       -0.355769       -0.217837        0.867993
 77 11       0.026209        -0.472346        0.868624
 78 12       -0.097842       -0.160037        0.641963
 79 13       0.092148        -0.284133        0.503535
 80 14       0.243100        -0.296265        0.108458
 81 15       0.247475        -0.207585        0.075391
 82 16       0.461261        -0.643618        -0.169754
 83 17       0.448960        -0.544194        -0.186882
 84 18       0.829567        -0.726274        -0.539448
 85 19       0.645662        -0.944631        -0.750293
 86 20       0.760741        -1.040396        -0.535894
 87 21       0.697329        -0.922957        -0.642949
 88 22       0.978761        -1.296403        -0.609750
 89 23       0.557497        -1.237046        -0.585898
 90 24       0.683706        -1.232123        -0.676077
 91 25       0.571126        -1.115355        -0.754792
 92 26       0.380659        -1.117273        -0.337285
 93 27       0.588741        -1.021961        -0.599668
 94 28       -0.000134       -0.676186        -0.323989
 95 29       0.071527        -0.259261        -0.195221
 96 30       -0.113579       -0.010309        0.074317
 97 31       -0.125549       -0.023607        0.024132
 98 32       -0.180302       0.290234         0.554987
 99 33       -0.106424       0.482674         0.365821
100 34       -0.080421       0.237817         0.251422
101 35       -0.280818       0.494389         0.245968
102 36       -0.042863       0.372541         0.632269
103 37       -0.079914       0.241705         0.377324
104 38       -0.153073       -0.087918        0.590405
105 39       -0.119164       0.104464         1.438204
106 40       0.315513        -0.580738        1.847756
107 41       0.977283        -2.092174        3.134419
108 42       1.269199        -2.650646        4.017070
109 43       1.307523        -2.754954        4.426551
110 44       1.296831        -2.695331        4.619812
111 45       1.196833        -2.409552        4.651111
112 46       1.376683        -3.025879        4.918630
113 47       1.836686        -2.927057        5.299540
114 48       1.557731        -2.582599        5.501376
115 49       1.747836        -2.398743        5.802055
116 50       1.889597        -2.439069        5.750459
117 51       1.693099        -2.448240        5.648962
118 52       1.807840        -2.560319        5.588957
119 53       1.538125        -2.570926        5.485146
120 54       1.228576        -2.599459        5.322882
121 55       0.902064        -2.262993        4.661784
122 56       0.648424        -1.570959        4.173109
123 57       -0.287012       -0.435386        3.077795
124 58       -0.438050       -0.512732        2.598338
125 59       -0.733482       -0.640386        1.581877
126 60       -0.618824       0.666682         1.105358
127 61       -0.386613       0.728610         1.368576
128
129 mouth_top
130 1        0.280147        -27.444398       7.022328
131 2        -0.398830       -0.036989        0.406755
132 3        -0.369227       -0.023079        0.541496
```

133	4	-0.405748	-0.069994	0.577201
134	5	-0.417349	0.100056	0.539165
135	6	-0.284462	-0.031234	0.722848
136	7	-0.311093	-0.061942	0.473486
137	8	-0.258779	0.024009	0.526248
138	9	-0.306776	-0.101750	0.537232
139	10	-0.213685	-0.117983	0.565486
140	11	0.037470	-0.022225	0.750074
141	12	-0.153615	0.070045	0.653990
142	13	0.044226	0.123053	0.710215
143	14	0.013421	-0.250822	0.492790
144	15	-0.058017	-0.035529	0.610902
145	16	0.240094	-0.396812	0.324457
146	17	0.209736	-0.509785	0.031180
147	18	0.275335	-0.967818	-0.425316
148	19	0.012879	-1.267871	-0.590487
149	20	0.110850	-1.384684	-0.395182
150	21	0.084863	-1.044387	-0.419847
151	22	0.338165	-1.347552	-0.424030
152	23	0.145632	-1.171611	-0.432487
153	24	0.069666	-1.120177	-0.479718
154	25	0.017606	-1.221092	-0.295655
155	26	0.003934	-1.127503	-0.296623
156	27	0.217378	-1.119505	-0.453134
157	28	-0.458343	-0.848150	-0.217932
158	29	-0.146884	-0.421573	-0.107818
159	30	-0.269244	-0.475699	0.358495
160	31	-0.298490	-0.151718	0.349116
161	32	-0.104612	0.060661	0.502921
162	33	-0.348547	0.081480	0.422699
163	34	-0.262303	-0.276868	0.171012
164	35	-0.202471	-0.046468	0.190236
165	36	-0.418908	-0.229853	0.657354
166	37	-0.284677	-0.178592	0.380271
167	38	-0.074330	0.282068	1.031454
168	39	0.178155	0.768183	2.186624
169	40	0.291697	1.402238	2.435420
170	41	0.538901	3.125006	4.212423
171	42	0.518299	4.201457	4.979557
172	43	0.558325	4.681173	5.457683
173	44	0.573813	5.073551	5.777932
174	45	0.564090	5.119261	5.768768
175	46	0.296787	4.990024	5.763700
176	47	0.548161	4.979122	5.894734
177	48	0.333422	5.372093	5.877083
178	49	0.338279	5.282677	5.929216
179	50	0.410642	5.308313	5.853662
180	51	0.325430	5.373768	5.971227
181	52	0.404762	5.275160	5.852622
182	53	0.355629	5.005908	5.761414
183	54	0.457354	4.921821	5.612704
184	55	0.367965	4.268443	5.285855
185	56	0.468079	4.154418	4.668424
186	57	0.336312	3.218802	4.110154
187	58	0.182030	2.876375	3.513278
188	59	0.168353	2.173387	2.828735
189	60	0.109220	1.376756	1.965868
190	61	0.139591	0.537342	1.696874
191				
192	jaw_bottom			
193	1	2.768923	-84.508281	13.708966
194	2	-0.508494	-0.839779	1.527801
195	3	-0.485992	-0.488051	1.602721
196	4	-0.548955	-0.790265	1.657658
197	5	-0.695215	-0.449896	1.616504
198	6	-0.658265	-0.642201	1.641676

199	7	-0.406683	-0.695063	1.505511
200	8	-0.400287	-0.631662	1.348285
201	9	-0.591487	-0.896908	1.424879
202	10	-0.385690	-0.622096	1.258284
203	11	0.063833	-1.186834	1.490739
204	12	-0.312490	-0.998512	0.861344
205	13	0.097716	-1.142573	0.518673
206	14	0.175347	-1.359634	-0.070240
207	15	0.161849	-1.056525	-0.255654
208	16	0.382207	-1.624939	-0.699026
209	17	0.441112	-1.820089	-1.099065
210	18	0.965588	-1.950883	-1.298993
211	19	0.956655	-2.428477	-1.684854
212	20	1.027416	-2.605977	-1.449927
213	21	1.016062	-2.182292	-1.446633
214	22	1.291703	-2.667973	-1.414413
215	23	0.829410	-2.534203	-1.379935
216	24	1.046027	-2.527101	-1.502022
217	25	0.920601	-2.175263	-1.441261
218	26	0.484944	-2.467739	-1.077473
219	27	0.768726	-2.191859	-1.432074
220	28	0.181673	-1.864677	-0.974328
221	29	0.025383	-0.977761	-0.657902
222	30	-0.101681	-0.757263	-0.339630
223	31	-0.170643	-0.375197	-0.231011
224	32	-0.124932	-0.086015	0.303986
225	33	-0.136466	0.094869	0.058783
226	34	-0.335454	0.053250	-0.005893
227	35	-0.266503	0.185999	0.216742
228	36	-0.188643	-0.014410	0.339158
229	37	-0.283442	0.066541	0.193399
230	38	-0.094343	-0.377732	0.261596
231	39	-0.269121	-0.290809	0.390726
232	40	-0.393013	-0.591429	-0.132659
233	41	-0.286023	-1.455368	-0.335386
234	42	-0.246826	-2.379219	-0.190118
235	43	-0.067896	-2.932156	-0.167745
236	44	-0.031607	-3.103475	-0.197735
237	45	-0.277555	-2.818117	-0.424187
238	46	-0.153746	-3.394007	-0.171010
239	47	0.217101	-3.548360	0.036157
240	48	0.016500	-3.241651	0.144989
241	49	0.065356	-3.463800	0.132449
242	50	0.321753	-3.582601	0.333012
243	51	0.037961	-3.234986	0.152434
244	52	0.174553	-3.162489	0.110755
245	53	-0.084010	-3.065869	0.001958
246	54	-0.338726	-2.676203	0.220364
247	55	-0.146017	-2.334926	0.145500
248	56	-0.234155	-1.047848	0.361974
249	57	-0.453425	-0.231327	0.324192
250	58	-0.735339	0.155523	0.382502
251	59	-0.590383	0.003110	0.265926
252	60	-0.475596	0.638943	0.367518
253	61	-0.443982	0.620654	0.632685
254				
255	jaw_top			
256	1	31.996117	-67.612666	32.928936
257	2	-0.177314	-0.232381	1.020385
258	3	-0.211212	-0.058198	0.954200
259	4	-0.197976	-0.221189	0.936669
260	5	-0.366394	-0.117750	1.000586
261	6	-0.126955	-0.203081	1.104817
262	7	-0.227321	-0.225230	0.899554
263	8	-0.155093	-0.275081	0.923003
264	9	-0.018792	-0.274430	0.836762

265	10	-0.133546	-0.081897	0.810101
266	11	0.383355	-0.734942	0.806832
267	12	0.152826	-0.724840	0.378827
268	13	0.422129	-0.681928	0.144099
269	14	0.731898	-1.066611	-0.162876
270	15	0.702768	-0.839799	-0.425905
271	16	0.856416	-1.170240	-0.615064
272	17	1.011873	-1.254221	-0.884779
273	18	1.519752	-1.368356	-1.405671
274	19	1.362666	-1.836037	-1.695234
275	20	1.670451	-2.249845	-1.409501
276	21	1.417965	-1.672748	-1.317590
277	22	1.842080	-2.114258	-1.493391
278	23	1.383464	-2.187649	-1.395377
279	24	1.558298	-2.132584	-1.574466
280	25	1.377637	-1.995563	-1.477382
281	26	1.259329	-1.973876	-1.212345
282	27	1.490064	-1.939880	-1.397124
283	28	0.860061	-1.703335	-1.090169
284	29	0.770442	-0.901396	-0.828416
285	30	0.638643	-0.766474	-0.508553
286	31	0.649565	-0.609725	-0.376399
287	32	0.773138	-0.264482	-0.076714
288	33	0.587314	-0.121899	-0.134914
289	34	0.583806	-0.436860	-0.217280
290	35	0.597552	-0.155667	-0.083458
291	36	0.515319	-0.300909	0.009648
292	37	0.603103	-0.334715	-0.080485
293	38	0.556667	-0.037195	0.007158
294	39	0.600918	1.139971	0.443835
295	40	0.858275	1.815578	0.368607
296	41	1.012030	3.875003	1.269327
297	42	1.331473	5.375497	1.606677
298	43	1.629011	6.062985	1.968996
299	44	1.840415	6.367981	2.259065
300	45	1.805698	6.994235	2.335513
301	46	1.690648	6.685028	2.691907
302	47	1.908718	6.870068	3.252131
303	48	1.714143	7.110552	3.539352
304	49	1.707949	7.298429	3.709378
305	50	1.895956	7.195662	3.774798
306	51	1.633362	7.228806	3.821325
307	52	1.840405	6.996305	3.797374
308	53	1.645797	6.843121	3.733510
309	54	1.372189	6.549918	3.670757
310	55	1.223839	5.877559	3.293157
311	56	1.069650	5.641188	3.046363
312	57	0.570969	4.442100	2.609362
313	58	0.291330	3.467447	2.391679
314	59	0.382637	2.504908	1.865578
315	60	0.292316	1.936968	0.964496
316	61	-0.052633	2.029947	1.113170
317				
318	cheek_inside			
319	1	21.791173	-19.350124	21.657690
320	2	0.220445	-0.199400	0.472386
321	3	0.265235	-0.147943	0.511058
322	4	0.111686	-0.386928	0.563114
323	5	0.028076	-0.120814	0.508701
324	6	0.169502	-0.271305	0.524676
325	7	0.243305	-0.165009	0.234493
326	8	0.257470	-0.176529	0.455774
327	9	0.303058	-0.327661	0.418786
328	10	0.346276	-0.201047	0.402527
329	11	0.470768	-0.426107	0.563316
330	12	0.297324	-0.253312	0.301526

331	13	0.570036	-0.049384	0.309073
332	14	0.496309	-0.081166	0.412398
333	15	0.534555	-0.087330	0.325548
334	16	0.521153	-0.178306	0.277229
335	17	0.455907	0.118112	0.062354
336	18	0.470643	0.211364	0.101380
337	19	0.159381	0.020697	0.174271
338	20	0.431233	-0.119230	0.139088
339	21	0.357028	-0.078042	0.064864
340	22	0.544229	-0.287990	0.049928
341	23	0.381977	-0.249351	0.144921
342	24	0.518663	-0.242020	0.189947
343	25	0.421276	-0.216630	-0.006481
344	26	0.297387	-0.306499	0.217675
345	27	0.493175	-0.292584	0.054317
346	28	0.248912	-0.282434	0.351999
347	29	0.271220	-0.068595	0.268510
348	30	0.242317	-0.125536	0.491594
349	31	0.340149	0.010951	0.327134
350	32	0.341357	-0.166458	0.549639
351	33	0.354961	-0.036184	0.548618
352	34	0.241191	-0.335011	0.359885
353	35	0.312337	-0.128645	0.612232
354	36	0.255934	-0.229804	0.597370
355	37	0.349118	-0.298582	0.366749
356	38	0.691157	0.118048	0.466947
357	39	1.154520	1.395878	1.149954
358	40	1.697232	2.276893	0.697130
359	41	2.997471	4.465239	0.137710
360	42	4.385436	5.609505	1.573711
361	43	5.392102	6.215193	3.606692
362	44	5.509688	6.464320	3.273752
363	45	5.322709	6.784035	2.763379
364	46	5.671319	6.579575	4.236530
365	47	5.951261	6.574598	4.017271
366	48	5.874183	6.739097	3.887639
367	49	6.049413	6.880677	4.570086
368	50	5.816812	6.700044	3.018275
369	51	5.285091	6.711971	2.621541
370	52	5.652042	6.517763	3.301991
371	53	5.453460	6.425905	3.318974
372	54	4.767962	6.239662	2.200222
373	55	4.932798	5.530429	3.071141
374	56	4.838148	5.169672	4.650187
375	57	4.438694	4.123241	6.214527
376	58	2.597368	3.383234	0.843964
377	59	2.123901	2.385930	0.373453
378	60	1.914185	1.679632	0.657273
379	61	1.647160	1.464733	0.441197
380				
381	cheek_outside			
382	1	42.039618	-17.289340	26.802411
383	2	0.589025	-0.450668	0.022934
384	3	0.546469	-0.303195	0.000689
385	4	0.543210	-0.219250	0.011666
386	5	0.465623	-0.291438	-0.054225
387	6	0.550140	-0.256171	-0.044696
388	7	0.510841	-0.285757	-0.115491
389	8	0.541928	-0.322706	-0.093293
390	9	0.637448	-0.389922	-0.145024
391	10	0.675559	-0.208588	-0.072890
392	11	0.949510	-0.422149	-0.073523
393	12	0.717468	-0.368084	-0.312759
394	13	1.028230	-0.226114	-0.406185
395	14	1.216747	-0.258306	-0.638288
396	15	1.229629	-0.083206	-0.918702

397	16	1.280420	-0.095632	-1.097946
398	17	1.428789	0.048930	-1.140261
399	18	1.527507	0.057801	-1.450240
400	19	1.177095	-0.039951	-1.219479
401	20	1.328189	-0.374731	-1.144396
402	21	1.364277	-0.175850	-1.140755
403	22	1.439067	-0.319574	-1.205453
404	23	1.132940	-0.324996	-1.135460
405	24	1.206087	-0.342504	-1.325713
406	25	1.161158	-0.144624	-1.276814
407	26	0.999910	-0.431315	-1.270789
408	27	1.225318	-0.293295	-1.031557
409	28	0.990536	-0.338582	-1.014806
410	29	1.025697	-0.087245	-0.803309
411	30	0.977846	-0.035340	-0.697082
412	31	1.052020	-0.243606	-0.587156
413	32	0.958458	-0.020888	-0.466727
414	33	0.904507	0.023640	-0.422761
415	34	0.791086	-0.290212	-0.671036
416	35	0.779081	-0.111895	-0.422081
417	36	0.815076	-0.399690	-0.340837
418	37	0.915836	-0.290253	-0.298553
419	38	1.486835	0.033179	-0.779821
420	39	2.305446	1.043831	-1.352910
421	40	2.975446	2.083096	-2.192504
422	41	4.640564	4.391416	-3.247044
423	42	5.663137	5.882399	-3.792959
424	43	6.208384	6.687064	-4.125698
425	44	6.612540	7.010960	-4.395763
426	45	6.579026	7.369471	-4.364527
427	46	6.558964	7.433091	-4.527088
428	47	6.681897	7.283048	-4.623836
429	48	6.734666	7.585518	-4.765189
430	49	6.738301	7.644794	-4.753912
431	50	6.689441	7.737083	-4.833772
432	51	6.485840	7.483571	-4.783952
433	52	6.675839	7.397900	-4.425175
434	53	6.266129	7.002902	-4.301777
435	54	6.119471	6.828555	-4.244360
436	55	5.926885	6.208773	-3.777436
437	56	5.567817	5.620011	-3.231595
438	57	4.703028	4.702733	-2.924237
439	58	3.985760	3.795562	-2.621644
440	59	3.296726	2.670665	-2.053490
441	60	2.691431	1.902974	-1.774360
442	61	2.249885	1.700721	-1.437604
443				
444	eye_bottom			
445	1	31.440988	7.540249	26.025346
446	2	0.492174	-0.266653	-0.008331
447	3	0.409487	-0.023169	-0.018833
448	4	0.328624	-0.111479	-0.068171
449	5	0.435400	-0.104872	-0.068367
450	6	0.496960	-0.106797	0.038198
451	7	0.447006	-0.052000	-0.025132
452	8	0.368005	-0.133010	0.030087
453	9	0.146950	0.044163	-0.018684
454	10	0.096971	0.183735	-0.032434
455	11	0.203033	0.093382	0.026364
456	12	0.072327	0.249279	-0.067136
457	13	0.288335	0.447801	-0.189031
458	14	0.114653	0.570273	-0.450154
459	15	0.193878	0.834937	-0.385623
460	16	0.230235	0.713312	-0.575131
461	17	0.420544	1.022793	-0.686652
462	18	0.453154	1.056732	-0.894899

463	19	0.430331	0.953655	-1.025372
464	20	0.469079	0.883419	-0.843554
465	21	0.336773	0.808235	-0.775912
466	22	0.427449	0.826014	-0.876775
467	23	0.371411	0.734102	-0.748277
468	24	0.194131	0.825725	-0.882539
469	25	0.229549	0.750486	-1.004254
470	26	-0.103209	0.830026	-0.720406
471	27	0.139066	0.763886	-0.819353
472	28	-0.219403	0.712447	-0.666727
473	29	-0.158323	0.767517	-0.593450
474	30	-0.005167	0.511592	-0.411972
475	31	0.288934	0.490603	-0.403521
476	32	0.547460	0.238904	-0.241742
477	33	0.483614	0.229455	-0.369458
478	34	0.410526	-0.052701	-0.403884
479	35	0.491934	0.104560	-0.110686
480	36	0.363049	-0.036628	-0.079226
481	37	0.437007	0.035698	-0.226584
482	38	0.873823	0.355945	-0.185721
483	39	1.219004	1.252289	-0.454513
484	40	1.732987	2.114230	-0.759096
485	41	2.370145	3.991088	-1.701031
486	42	2.981456	5.052090	-2.089591
487	43	3.186910	5.748642	-2.221665
488	44	3.340060	6.000823	-2.515486
489	45	3.304895	6.271546	-2.634510
490	46	3.139044	6.101068	-2.705665
491	47	3.389495	6.247625	-2.593931
492	48	3.045164	6.447057	-2.751915
493	49	2.952342	6.457957	-2.677386
494	50	2.915815	6.457544	-2.669862
495	51	2.867243	6.525110	-2.617960
496	52	3.029004	6.324522	-2.347416
497	53	2.770289	6.030956	-2.305547
498	54	2.880005	5.759912	-2.199271
499	55	2.732903	5.248756	-1.750727
500	56	2.562447	4.733798	-1.565213
501	57	2.278349	4.062210	-1.412813
502	58	1.862741	3.321518	-1.297952
503	59	1.527645	2.298319	-1.057087
504	60	1.129536	1.708942	-0.825679
505	61	0.964428	1.432633	-0.595630
506				
507	eye_side			
508	1	51.318109	19.637380	43.673147
509	2	0.372141	-0.343741	-0.293902
510	3	0.420918	-0.220800	-0.371995
511	4	0.324705	-0.075683	-0.429810
512	5	0.433656	-0.232280	-0.412111
513	6	0.376713	-0.142446	-0.259443
514	7	0.371263	-0.123984	-0.331530
515	8	0.446145	-0.203623	-0.190459
516	9	0.278413	-0.161853	-0.454250
517	10	0.291384	-0.149415	-0.517445
518	11	0.301532	-0.128967	-0.740244
519	12	0.110439	-0.156303	-0.863875
520	13	0.084652	-0.180590	-0.914232
521	14	0.123589	-0.127453	-1.162053
522	15	0.037086	-0.039287	-1.330591
523	16	0.053018	-0.036736	-1.425571
524	17	0.083692	0.087389	-1.513910
525	18	0.408638	0.138133	-1.440716
526	19	0.295947	0.254303	-1.511827
527	20	0.310554	0.323788	-1.417721
528	21	0.225837	0.268731	-1.462302

529	22	0.298873	0.164344	-1.583612
530	23	0.333612	0.052117	-1.358832
531	24	0.257620	0.155116	-1.672510
532	25	0.183158	0.140662	-1.720270
533	26	0.127570	0.273759	-1.437458
534	27	0.208359	0.190500	-1.411266
535	28	0.138988	0.097934	-1.415539
536	29	0.081199	0.084914	-1.292410
537	30	0.204641	0.028044	-1.183834
538	31	0.534633	-0.030385	-0.782625
539	32	0.548837	0.068153	-0.563831
540	33	0.532006	-0.157690	-0.661950
541	34	0.319018	-0.105825	-0.598595
542	35	0.447880	-0.041027	-0.376344
543	36	0.485463	-0.087338	-0.587668
544	37	0.539063	-0.134912	-0.600245
545	38	0.497007	-0.244052	-0.627341
546	39	0.556508	-0.047554	-0.713649
547	40	0.498547	0.498708	-1.056526
548	41	0.527218	1.249819	-1.670363
549	42	0.593299	1.567667	-2.384000
550	43	0.775185	2.110003	-2.911756
551	44	0.865006	2.048468	-3.752851
552	45	0.848861	2.498286	-3.788103
553	46	0.759065	2.328523	-3.797459
554	47	0.846597	2.308979	-3.973616
555	48	0.874525	2.338606	-4.373823
556	49	1.018118	2.522171	-4.394687
557	50	0.875765	2.542500	-4.484938
558	51	0.872596	2.348191	-4.229356
559	52	0.873251	2.268378	-4.140376
560	53	0.833073	2.425933	-4.105957
561	54	0.664683	1.938324	-3.954708
562	55	0.996593	1.808690	-3.446311
563	56	1.089631	1.692667	-2.939542
564	57	0.912623	0.851701	-2.670492
565	58	0.740986	0.688258	-2.150202
566	59	0.705451	0.436115	-1.710799
567	60	0.758350	0.174321	-1.550318
568	61	0.625146	0.140654	-1.307000
569				
570	eyeblow_center			
571	1	3.783747	52.367225	9.290642
572	2	-0.137677	-0.044288	0.071322
573	3	-0.032975	-0.125032	0.077871
574	4	-0.177915	0.168650	0.025830
575	5	-0.092251	0.048582	0.001176
576	6	0.009711	0.114631	0.150518
577	7	-0.120250	0.121519	0.092702
578	8	0.018739	0.045604	0.105624
579	9	-0.167018	-0.283906	-0.111576
580	10	-0.201723	-0.367351	-0.135561
581	11	-0.473164	-0.475916	-0.362898
582	12	-0.493192	-0.628706	-0.608194
583	13	-0.499754	-0.790887	-0.597561
584	14	-0.833497	-1.138423	-1.063228
585	15	-0.765332	-1.394954	-1.118575
586	16	-0.857721	-1.697466	-1.412441
587	17	-1.047083	-1.994328	-1.549013
588	18	-0.904073	-2.511378	-1.646225
589	19	-0.886792	-2.810997	-1.444556
590	20	-0.816406	-2.663819	-1.503000
591	21	-0.812476	-2.862786	-1.483306
592	22	-0.758081	-2.851428	-1.593118
593	23	-0.855536	-2.861496	-1.554996
594	24	-1.041045	-2.674725	-1.661487

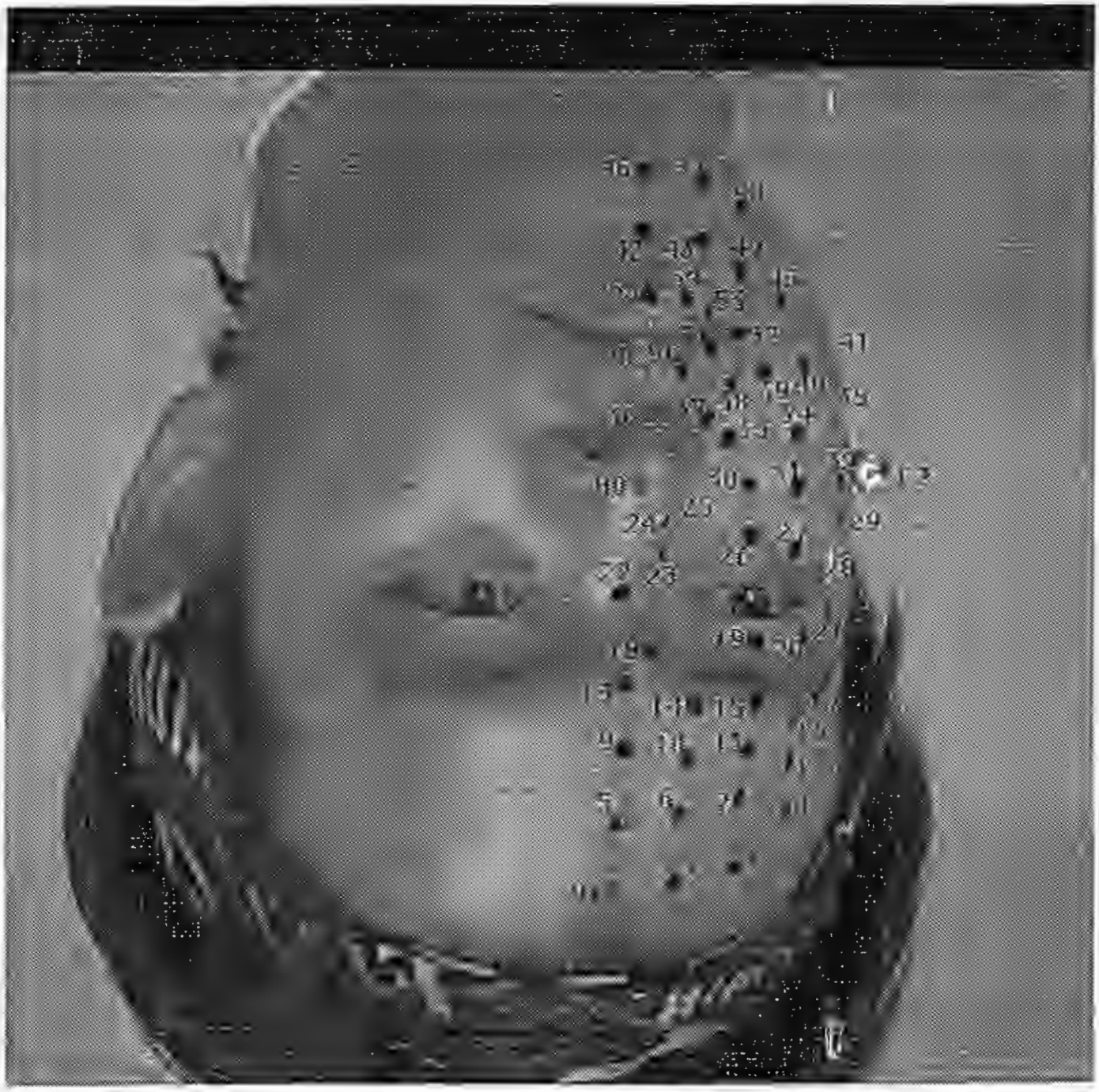
595	25	-0.941748	-2.705885	-1.538290
596	26	-0.859020	-2.873419	-1.581468
597	27	-1.065500	-2.461441	-1.605650
598	28	-1.291477	-2.138112	-1.358658
599	29	-1.195897	-1.662663	-1.279763
600	30	-1.229519	-1.517329	-1.157281
601	31	-0.770503	-1.049136	-0.887086
602	32	-0.646633	-0.730378	-0.560859
603	33	-0.559892	-0.612339	-0.662027
604	34	-0.593938	-0.496463	-0.526883
605	35	-0.592974	-0.427315	-0.406830
606	36	-0.308310	-0.447852	-0.334931
607	37	-0.279551	-0.428064	-0.344362
608	38	-0.238176	-0.423131	-0.280320
609	39	-0.311836	-0.414830	-0.262940
610	40	-0.254760	-0.348952	-0.450376
611	41	-0.366250	-0.322656	-0.493065
612	42	-0.195960	-0.524936	-0.329748
613	43	-0.338772	-0.461756	-0.332608
614	44	-0.155297	-0.376536	-0.390568
615	45	-0.239388	-0.235182	-0.432873
616	46	-0.311604	-0.278584	-0.514696
617	47	-0.057455	-0.300570	-0.547010
618	48	-0.238293	-0.093951	-0.508541
619	49	-0.248960	-0.089824	-0.365573
620	50	-0.047625	-0.085708	-0.327531
621	51	-0.408864	0.007236	-0.397456
622	52	-0.179375	0.014618	-0.305911
623	53	-0.094151	0.066243	-0.387240
624	54	-0.131997	-0.095864	-0.458357
625	55	-0.167297	-0.166926	-0.351483
626	56	-0.029590	0.024970	-0.463595
627	57	0.169726	-0.100855	-0.307930
628	58	-0.033910	-0.106299	-0.142888
629	59	-0.096125	-0.102366	-0.277146
630	60	-0.017316	-0.100053	-0.110513
631	61	-0.084677	-0.028216	0.033728

付録.2

顔に貼付したマーカー位置及び番号

１枚目：本実験（６表情変化）で用いたもの
（９８測定点）

２枚目：追加実験（AU・口形変化）で用いたもの
（１００測定点）

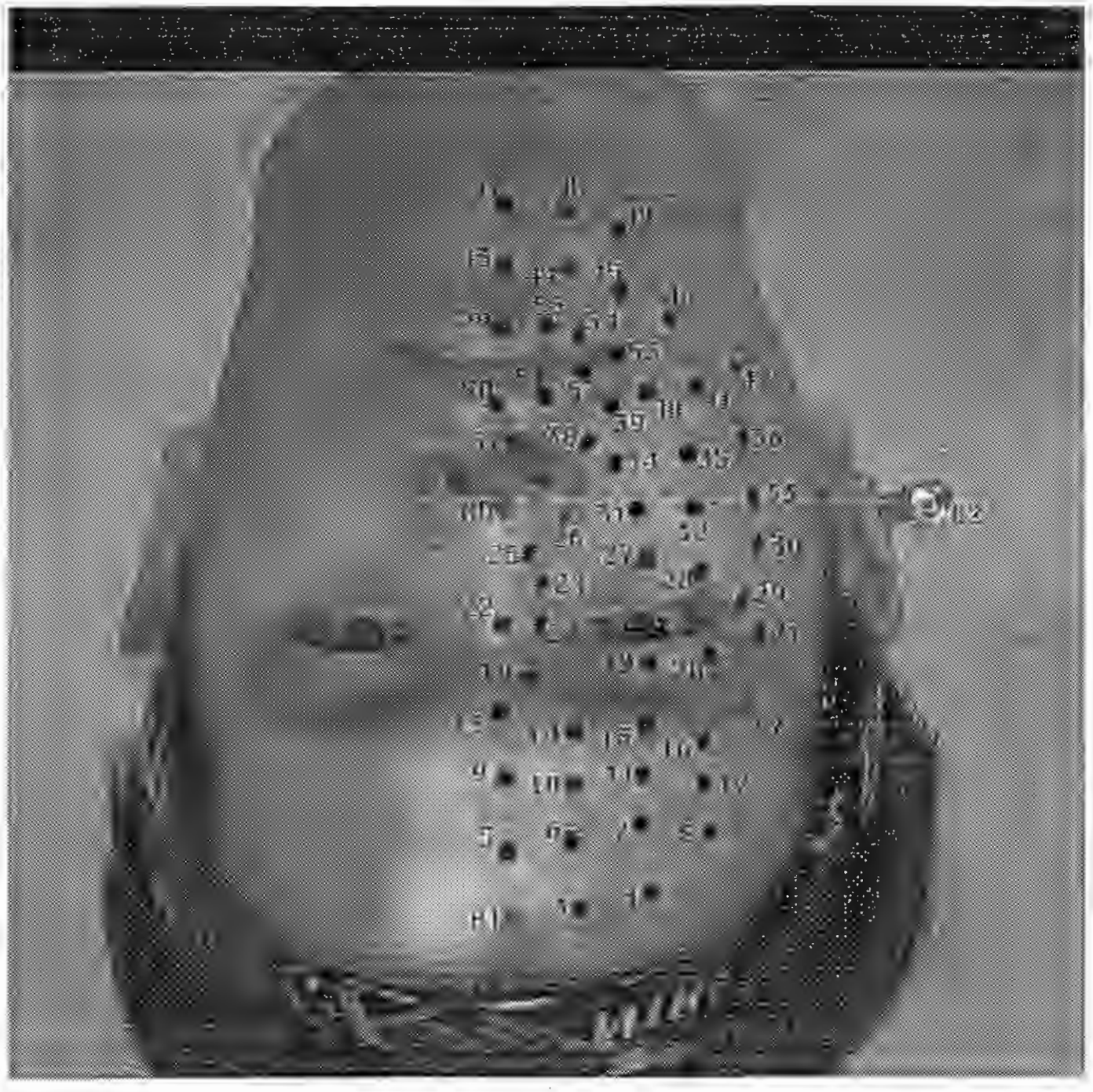


実験①

56点

(0~55)

顔全体 98点



実験②.③

57点.

(0~56)

顔全体 100点.

付録.3

顔画像再合成までの一連の流れ

（○graphicaでの撮影）

I. ~/atr/src/2Dto3Dconvert/exp2.1
正面画と側面画での点の座標の取得

II. ~/atr/src/2Dto3Dconvert/exp4.1
得られた正面側面それぞれの座標値から３次元のベクトルを生成
出力：lvector（固定）

III. ~/atr/src/2Dto3Dconvert/exp5
３次元ベクトルを１フレーム目からの移動量に換算
出力：movement

IV. ○~/atr/src/cyber/getgrid(imgdisp併用)
顔上の基準点をochi_grid_1上の座標に変換（一部手作業）
出力：control（手入力）

V. ○~/atr/src/model/triweight
基準点同士で３角形を構成し、他の格子点の移動量にかかわる重みを求める。
出力：cargroup（重みファイル）

VI. ○~/atr/src/model/dispweight
重みの修正、削除などを行なう
出力：cargroup（重みファイル）

VII. ○~/atr/src/model/deform
movementファイル及び重みファイル等から、モデルを変形し
Wavefrontフォーマットで出力する
出力：Wavefrontファイル

VIII. ○~/live
WaveFrontファイルを視覚化する

I. 正面画と側面画から測定点の座標を取得する

(/home/tatsu/atr/src/2Dto3Dconvert/exp2.1)

準備：

まず、測定した画像データ（.r .g .b 汎用フォーマット）を、準備する。

それぞれファイルネームは001.r等の様にフレーム番号（３桁）+拡張子（プレーン識別子）で構成されていること。MOに保存されているデータはすべてこの名前に統一されている。

正面画像を、

/home/tatsu/atr/data/fimage

ディレクトリに、側面画像を、

/home/tatsu/atr/data/simage

ディレクトリにそれぞれコピーする。

※正面画と側面画のデータはともにファイル名が同一なので注意

※ディスク容量の関係で一度に大量のデータをコピーできない場合は何枚かに分けて行なう。その際もファイル名はMOに保存されているままのファイルネームで良い。

全ての制御点について入力する順番を決定する。（付録に仮決定した図を付ける）

```
cd /home/tatsu/atr/src/2Dto3Dconvert
```

でディレクトリの移動を行なう。

※プログラム内で相対パスを使用しているために、かならずカレントを2Dto3Dconvertにすること。

プログラムの実行：

```
exp2.1 ↵
```

プログラムの使用方法：

1.まず実行するとファイル名を尋ねてくるので、正面画のファイルネームを入力する。例えば、

```
fimage/001 ↵
```

等のようにすればよい。拡張子は省略する。相対パスでも絶対パスでも可。

2.続いて正面画か側面画かを尋ねてくるので、正面画であるから

```
f ↵
```

とする。

3.座標表示のためのウインドウ位置を指定するようにメッセージが表示されるので、任意の場所に持っていき、マウス左ボタンで確定する。

4.拡大表示のためのウインドウ位置を指定するようにメッセージが表示されるので、任意の場所に持っていき、マウス左ボタンで確定する。

5.全体画像表示のためのウインドウ位置を指定するようにメッセージが表示されるので、任意の場所に持っていき、マウス左ボタンで確定する。

※ウインドウ位置を一度確定すると移動が出来ないので、ウインドウが重ならないように、また実行したウインドウから出力されるメッセージが見えるように配置する。

6.鼻頂点の基準点（A）のだいたいの位置を合わせマウス左ボタンでクリックする。

7.拡大ウインドウ内の+記号をマーカーの中心に合わせ、左ボタンをクリックする。

このように一つのマーカー位置に対して２回クリックすることになる。以後この作業は「位置を指定する」とだけ言う。

8.同様にB（額上部）、C（耳に付けられたクリップの先端）の基準点の位置を指定する。

9.先に決めた順番で制御点の位置を指定する。正面画での制御点の入力の際は、位置の指定の後にその点の名前の入力を要求される。ここでは自分が分かりやすい名前を付ければよい。点の順番をあらかじめ決めている場合には、名前は適当でよい。（q 以外）

この名前の入力は、全体画像表示のためのウインドウ内にカーソルをおいて行なえる。

10.全ての制御点の入力が終了したら、適当な位置でマウスをクリックし、名前入力のときに、q を入力する。

11.正面画での座標入力が終了すると、

/home/tatsu/atr/data/fgrid

ディレクトリの下に、そのフレームの情報を出力したファイルができる。

12.次に再びプログラムを実行し、今度は側面画のファイルネームを入力する。またファイルの種類では側面画の s を入力する。

13.先と同様の手順でウインドウを配置する。

14.A点の位置の指定を行なう。

15.B点、C点の位置の指定を行なう。側面画の場合は、名前を入力する代わりに、これで正しいかどうかの確認をしてくるので、よければ y を、だめならば n を入力する。

16.各制御点の位置を入力する。15.と同様に確認してくるので答える。

17.全ての点を入力し終ると、自動的に、

/home/tatsu/atr/data/sgrid

に情報を出力してプログラムが終了する。

18.全てのフレームに対して正面画と側面画での座標入力を繰り返す。

※途中で間違いに気づいた場合、全体画像表示のためのウインドウをquitしてプログラムの実行を止め、最初からやり直す。

II. 得られた正面側面それぞれの座標値から３次元のベクトルを生成

（ ~/atr/src/2Dto3Dconvert/exp4.1 ）

準備：

あらかじめすべてのフレームに対して座標を求めておく。（exp2.1を使用）
また出力ファイルであるlvectorと同じ名前のファイルが、

~/atr/src/2Dto3Dconvert

ディレクトリにある場合はリネームしておく。
カレントディレクトリを

~/atr/src/2Dto3Dconvert

に移動する。

プログラムの実行：

exp4.1ソ

プログラムの使用方法：
実行すると、フレーム数を聞いてくるので、入力する。
MOに入っているデータの場合２２６と入力する。

実行すると

~/atr/src/data/fgrid
~/atr/src/data/sgrid

ディレクトリの下にあるファイルを元に、３次元座標を計算して、
カレントディレクトリに

lvector

というファイルを出力して終了する。

このlvectorファイルをエディターで開き、見てみるとヘッダ部分はこのようになっているはずである。

```
226 57
***.******** ***.******** ***.********
***.******** ***.******** ***.********
***.******** ***.******** ***.********
top_of_nose_(_point_A_)
forehead_center_(_point_B_)
ear_center_(_point_C_)
```

この ***.******** 部分には、何かの数字が書かれている筈だが、ここを

-4.653403 -16.028393 -102.034932
2.207494 101.531775 22.826604
112.014798 -5.304143 128.662775

に書き直す。
これは基準ベクトルの値で、サイバーウエアから取得した正しい値である。
書き換えを行なわないと、画像の１フレーム目の値がそのまま使われてしまうために座標が狂う。

III.　３次元ベクトルを１フレーム目からの移動量に換算

(~/atr/src/2Dto3Dconvert/exp5)

準備：
あらかじめ exp4.1 を用いて、lvectorファイルを作成しておく。
カレントディレクトリを
~/atr/src/2Dto3Dconvert
に移動する。
カレントディレクトリに movement という名前のファイルが存在する場合、リネームしておく。

プログラムの実行：
exp5 ↵

プログラムの使用方法：
このプログラムはファイルネーム固定のlvectorファイルを読み込み、ファイルネーム固定のmovementファイルを書き出すので、実行するだけである。

出力：movement

※この movement ファイルは、モデルの変形に直接利用するファイルなので、内容を書き直したりしてははいけない。

IV.　顔上の基準点をochi_grid_1上の座標に変換

（　~/atr/src/cyber/getgrid(imgdisp併用 ）　）

一部手作業

測定に用いた制御点の、ochi_grid_1上での座標を求め、制御点ファイルを作成する。

具体的には、サイバーからの出力である .color イメージファイルをochi_grid_1.colorと見比べながら、ochi_grid_1.color上の座標値（x , y）を記録し、

~/atr/src/cyber/getgrid

を使ってモデル上の３次元座標に変換し、規定されたフォーマットにしたがって制御点ファイルを作る。

非常に大変かつ重要な作業なので、この制御点ファイルはあらかじめ用意した、

~/atr/src/model/control_exp3.2

というファイルがあるので、この作業は通常必要ない。

※このファイルは付録につけた、制御点の入力順番に準拠しているため、順番を変えた場合は使用できなくなる。

getgridの使い方：

getgrid [サイバーの出力するバイナリファイル名]↵

とし、あとは指示にしたがって、イメージデータ上の座標をx , y の順で入力してやればよい。入力した点に対応する３次元座標が出力される。

制御点ファイルのフォーマット：

あらかじめ用意した制御点ファイルを参照のこと。

出力：control（手入力）

V.　制御点ポリゴンを設定する。

（~/atr/src/model/triweight ）

基準点同士で３角形を構成し、他の格子点の移動量にかかわる重みを求める。

準備：

制御点ファイルを作成しておくこと。（あらかじめ用意したファイルを使う）

カレントディレクトリを~/atr/src/modelに移動する。

実行方法：

triweight [wavefront file] [制御点ファイル] [出力用重みファイル]↵

実際には tw.s というスクリプトを書いてあるので、その内容を書き変えて実行する形である。

wavefrontファイルの名前には拡張子を付けないで書く。

出力用重みファイルの名前は任意でよい。ただし、カレントに同じ名前のファイルが在る場合にはオーバーライトしてしまうので注意が必要。

使用方法：

実行するとウインドウの表示位置及び大きさを要求されるので、マウスで指定し、左ボタンで確定する。なるべく大きく取るほうが良い。

また実行したウインドウにはメッセージが表示され、また入力要求もあるので、隠さないようにする。

一度確定したウインドウの大きさ及び位置は変更できないので注意。

画面上では、格子点が黄色、制御点が緑で示される。

マウスの左ボタンで３角形の各頂点にあたる制御点を選択していき、３つ選択すると確認してくるので、実行ウインドウ内で、y もしくはnで答えればよい。

３角形が確定すると、その頂点が赤い線で結ばれ、内部に含まれる格子点も赤くなる。

複数のポリゴンを指定する場合は繰り返せば良い。

頂点の選択は、左ボタンの押された座標に一番近い制御点を選択するようになっている。途中で間違えた場合は確認の前であれば、nと答えることにより、無視されるが、確認後の場合には、quitして最初からやり直す。

すべてのポリゴンを指定したら、マウスの中ボタンをクリックすると確認の後セーブ終了となる。

セーブされるファイルは各格子点がどの制御点からどれだけの重みで影響を受けるかを記したもので、モデル変形等に直接関わるので、内容の書き換えはしないほうがよい。

VI.　重みの修正、削除などを行なう

（~/atr/src/model/dispweight）

triweightで作成された重みについて、口や目周辺の複雑な動きをする領域で、修正や削除を行なうためのプログラムである。

どのような場所の格子点について重みの削除や修正が必要かは、本編を参照のこと。

実行方法：

```
dispweight [WaveFront File] [ControlPoint File] [Weight File]
```

で、アーギュメントの省略は不可。またチェックもしていないので、アーギュメント

を省略した場合の動作は保証できない。

通常は dw.s というスクリプトを使用して実行する。

もし、上記のファイルが存在しない場合は、入力されたファイル名とともにエラーメッセージを表示し、シェルに戻る。

WaveFront Fileは、サイバーウェアの出力ファイルから作られた .obj で終るファイルである。ControlPoint Fileは、制御点のの座標情報を記したファイルで、IV.で作成したものである。なお、重みを求めるときに使ったものと同じでなければならない。

Weight Fileは、triweightによって出力された重みファイルである。

操作方法：

各モード共通

マウス右ボタン　モードの切り替え

モードはマウスの右ボタンをクリックすると、

DISP MODE->CONTROL POINT MODE->EDIT MODE->FILE MODE->DISP MODE

と変化する。

DISP MODE:

マウス左ボタン　画像の平行移動

左ボタンを押したままマウスをドラッグすると、マウスの動きに合わせて画面が平行移動する。

マウス中ボタン　画像の拡大縮小

中ボタンを押したままマウスを上下にドラッグすると、拡大・縮小する。

上にドラッグすると拡大、下にドラッグすると縮小である。

CONTROL POINT MODE:

マウス左ボタン　制御点の選択

制御点の影響範囲を知りたいときに利用する。

制御点の近くで左ボタンをクリックすると、その制御点から影響を受ける格子点が赤で表示される。

マウス中ボタン

拡張のための予備である。現在はなにもしない。

EDIT MODE:

マウス左ボタン　格子点の選択

格子点がどの制御点から影響を受けるかを知りたい時に利用する。

知りたい格子点の付近で左ボタンをクリックすると選択された格子点が赤で表示され、影響を及ぼす制御点が水色で表示される。また、最も大きな影響を与える制御点が青色で表示されます。

マウス中ボタン　制御点の選択

現在選択されている格子点の受ける影響を変更したい時に利用します。
重みを変更したい制御点の付近で、中ボタンをクリックするとその制御点が格子点に与える重みが%で表示される。
変更したいときには、重みを%単位で入力する。変更したくないときには、n を入力する。
このコマンドは、格子点が選択されていない場合（マウスの左ボタン）無視される。

FILE MODE:
マウス左ボタン　グループファイルのセーブ
EDIT MODEで変更した重みを保存したい時に利用する。
一度クリックすると、保存をするかどうか確認してくるので保存したい場合は y 、したくない場合は n を入力する。
yを入力すると、ファイル名を尋ねて来るので、保存するファイルの名前を入力する。
マウス中ボタン　プログラムの終了
プログラムを終了したいときに利用する。
一度クリックすると、終了してよいかどうかを確認してくるので終了したいときには y を、終了したくないときには n を入力する。
なお、終了を選択した場合にはグループファイルの保存は行なわれないので、重みの変更を行なった際は、終了する前に必ず保存（マウス左ボタン）する。

VII. movementファイル及び重みファイル等から、モデルを変形する
(~/atr/src/model/deform)

movementファイル及び重みファイル等から、モデルを変形しWavefrontフォーマットで出力するプログラムである。

実行方法：
deform [wavefront file] [movement file] [重みファイル] [出力ファイル] [制御点ファイル]
実際は de.s というスクリプトを書き換えて実行する。
アーギュメントの省略は不可。

プログラムの使用方法：
いままで作成してきた各種のファイルをアーギュメントで指定し、実行するだけである。
出力ファイルは、アーギュメントに指定されたファイル名に、フレーム番号（3桁）と .obj という拡張子が付加された、wavefront形式のファイルである。

VIII. WaveFrontファイルを視覚化する
（~/live　）

WaveFrontファイルを視覚化するプログラムである。
パスが通してあるので、どのディレクトリからでも実行可能。

実行方法：
live [wavefrontfile] [.colorイメージファイル]

起動後の使用方法は、従来の live , face と同様である。

付録.4

プログラムソースリスト

```
 1
 2 #include        <gl/gl.h>
 3 #include        <gl/device.h>
 4
 5 #include        "std.h"
 6 #include        "wavefront.h"
 7 #include        "cimage.h"
 8
 9 double CPvalueget();
10
11 /*****************************************************
12   WaveFront Format のデータファイルを読み込む
13 *****************************************************/
14
15 void ReadWaveFront( filename , wf , zoom )
16 char            *filename;
17 WaveFront       *wf;
18 int             zoom;
19 {
20   FILE          *fp;
21   int           i = 0 , j;
22   int           ml1 , ml2 , ml3 , tl1 , tl2 , tl3;
23   float         dx , dy , dz;
24   char          buff[256];
25   char          dum[100] , dum1[100];
26
27   printf( "zoom = %d\n" , zoom );
28
29 /* Wave Front Format のファイルを開く */
30
31   if( NULL == ( fp = fopen( filename , "r" ) ) ){
32     printf( "Cannot open file .. %s ( ReadWaveFront@readwavefront )\n" ,
33             filename );
34     exit( 1 );
35   }
36
37 /* Wave Front Format のヘッダ部分を飛ばす */
38
39   fgets( buff , 256 , fp );
40   while( buff[0] == '#' ) fgets( buff , 256 , fp );
41
42 /* 3 次元構造データ (v) を読む */
43
44   while( buff[1] != 't' ){
45     sscanf( buff , "%s %f %f %f" , dum , &dx , &dy , &dz );
46     wf->mgrid[i].x = dx * (float)zoom;
47     wf->mgrid[i].y = dy * (float)zoom;
48     wf->mgrid[i++].z = dz * (float)zoom;
49     fgets( buff , 256 , fp );
50
51 }
52   wf->gridnum = i;
53   printf( "gridnum = %d\n" , wf->gridnum );
54   i = 0;
55
56 /* テクスチャ上の座標を読む */
57
58   while( buff[1] == 't' ){
59     sscanf( buff , "%s %f %f %f" , dum , &dx , &dy , &dz );
60
61     wf->tgrid[i].x = dx;
62     wf->tgrid[i].y = dy;
63     wf->tgrid[i++].z = dz;
64     fgets( buff , 256 , fp );
65   }
66   wf->tgridnum = i;
```



```
67   printf( "tgridnum = %d\n" , i );
68   i = 0;
69
70 /* 情報を読む */
71
72   while( buff[0] != 'f' ){
73     sscanf( buff , "%s %s" , dum , dum1 );
74     if( strcmp( dum , "usemtl" ) == 0 );
75     if( strcmp( dum , "usemap" ) == 0 ) strcpy( wf->colfname , dum1 );
76     fgets( buff , 256 , fp );
77   }
78
79 /* リンク情報を読む */
80
81   i = 0;
82   GetWFLink( buff , wf , i );
83   i++;
84   while( 0 == feof( fp )){
85     fgets( buff , 256 , fp ) ;
86     GetWFLink( buff , wf  , i );
87     i++;
88   }
89
90   wf->linknum = i;
91
92   fclose( fp );
93 }
94
95 /*****************************************
96   読み込まれた文字列から数値列への変換
97 *****************************************/
98
99 GetWFLink( buff , wf , num )
100 char            *buff;
101 WaveFront       *wf;
102 int             num;
103 {
104   char          tmp1[50] , tmp2[50] , tmp3[50] , dum[50];
105
106   sscanf( buff , "%s %s %s %s" , dum , tmp1 , tmp2 , tmp3 );
107
108   Separate( tmp1 , &(wf->link[num].model.g1) , &(wf->link[num].tex.g1) );
109   Separate( tmp2 , &(wf->link[num].model.g2) , &(wf->link[num].tex.g2) );
110   Separate( tmp3 , &(wf->link[num].model.g3) , &(wf->link[num].tex.g3) );
111 }
112
113 /*****************************************
114   読み込まれた文字列から数値列への変換 (サブ)
115 *****************************************/
116 Separate( tmp , mg , tg )
117 char            *tmp;
118 int             *mg , *tg;
119 {
120   char          mch[10];
121   int           i = 0;
122
123   while( *tmp != '/' ) mch[i++] = *(tmp++);
124
125   mch[i] = '\0';
126   *mg = atoi( mch );
127   *tg = atoi( ++tmp );
128
129 }
130
131 /*****************************************
132   テクスチャ上の座標に対して一番近い点の
```

```
133   番号を返す
134 ****************************************/
135
136 int NearestPoint( wf , x , y )
137 WaveFront       *wf;
138 double          x , y;
139 {
140   double        dis , mindis ;
141   int           min , i;
142
143   min = 0;
144   mindis = sqrt(sqrdb( wf->tgrid[0].x - x ) + sqrdb( wf->tgrid[0].y - y ));
145
146   for( i = 1 ; i < wf->tgridnum ; i++ ){
147     dis = sqrt(sqrdb( wf->tgrid[i].x - x ) + sqrdb( wf->tgrid[i].y - y ));
148     if( mindis > dis ){
149       mindis = dis; min = i;
150     }
151   }
152   return( min );
153 }
154
155 int TextToReal( wf , num )
156 WaveFront               *wf;
157 int                     num;
158 {
159   int           i;
160
161   for( i = 0 ; i < wf->linknum ; i++ ){
162     if( wf->link[i].tex.g1 == (num +1)) return( wf->link[i].model.g1 -1);
163     if( wf->link[i].tex.g2 == (num +1)) return( wf->link[i].model.g2 -1);
164     if( wf->link[i].tex.g3 == (num +1)) return( wf->link[i].model.g3 -1);
165   }
166 }
167
168 #ifdef IRIS
169 void WFwriteOP( wf , siten , sizex , sizey , x , y , dot , col , clsw )
170 WaveFront       *wf;
171 float           sizex , sizey;
172 int             dot , clsw;
173 vector2         siten;
174 long            x , y ;
175 short           col;
176 {
177   int           i , j;
178   float         vert[2];
179
180   if( clsw == CLEAR_ON ){
181     Color( BLACK );
182     clear();
183   }
184
185   pntsize( (short)dot );
186   Color( (Colorindex)col );
187
188   for( i = 0 ; i < wf->tgridnum ; i++ ){
189     vert[0] = (float)wf->tgrid[i].x * sizex - (float)siten.x;
190     vert[1] = (float)wf->tgrid[i].y * sizey - (float)siten.y;
191
192     if( vert[0] > 0.0 && vert[0] < (float)x && vert[1] > 0.0 && vert[1] < (float
    )y ){
193       bgnpoint();
194       v2f( vert );
195       endpoint();
196     }
197   }
```



```
198 }
199
200 /*************************************************************
201   WaveFront のテクスチャ上の座標を基に画面上に線を引く
202 *************************************************************/
203
204 void WFwrite( wf , siten , sizex , sizey , x , y , dot )
205 WaveFront       *wf;
206 float           sizex , sizey;
207 int             dot;
208 vector2         siten;
209 long            x , y ;
210 {
211   int           i , j;
212   float         vert[3][3];
213
214   vert[0][2] = 0.0 ; vert[1][2] = 0.0 ; vert[2][2] = 0.0 ;
215
216   Color( BLACK );
217   clear();
218
219   for( i = 0 ; i < wf->linknum ; i++ ){
220     Color( WHITE );
221     vert[0][0] = (float)wf->tgrid[wf->link[i].tex.g1-1].x * sizex - (float)siten
    .x;
222     vert[0][1] = (float)wf->tgrid[wf->link[i].tex.g1-1].y * sizey - (float)siten
    .y;
223     vert[1][0] = (float)wf->tgrid[wf->link[i].tex.g2-1].x * sizex - (float)siten
    .x;
224     vert[1][1] = (float)wf->tgrid[wf->link[i].tex.g2-1].y * sizey - (float)siten
    .y;
225     vert[2][0] = (float)wf->tgrid[wf->link[i].tex.g3-1].x * sizex - (float)siten
    .x;
226     vert[2][1] = (float)wf->tgrid[wf->link[i].tex.g3-1].y * sizey - (float)siten
    .y;
227
228     if( vert[0][0] * vert[0][1] > 0 && vert[1][0] * vert[1][1] > 0 &&
229        vert[2][0] * vert[2][1] > 0 ){
230       if( vert[0][0] < x && vert[0][1] < y && vert[1][0] < x &&
231          vert[1][1] < y && vert[2][0] < x && vert[2][1] < y ){
232
233         bgnclosedline();
234         v3f( vert[0] );
235         v3f( vert[1] );
236         v3f( vert[2] );
237         endclosedline();
238
239         if( dot != DOT_OFF ){
240           Color( YELLOW );
241           pntsize( (short)dot );
242           bgnpoint();
243           v2f( vert[0] );
244           v2f( vert[1] );
245           v2f( vert[2] );
246           endpoint();
247         }
248       }
249     }
250   }
251 }
252
253
254
255 AllCPwrite( ip , siten , wsizex , wsizey , x , y , dot )
256 InflPoint       *ip;
257 vector2         siten;
```

```
258 float           wsizex , wsizey;
259 long            x , y ;
260 int             dot;
261 {
262   int           i , j;
263   float         vert[2];
264   float         ipx , ipy;
265
266   for( i = 0 ; i < ip->pnum ; i++ ){
267     ipx = (float)ip->tgrid[i].x * wsizex - (float)siten.x;
268     ipy = (float)ip->tgrid[i].y * wsizey - (float)siten.y;
269     if( ipx > 0 && ipy > 0 && ipx < x && ipy < y ){
270       vert[0] = ipx;
271       vert[1] = ipy;
272       pntsize( (short)dot );
273       Color( GREEN );
274       bgnpoint();
275       v2f( vert );
276       endpoint();
277     }
278   }
279 }
280 int CPointget( ip , siten , wsizex , wsizey , mval )
281 InflPoint               *ip;
282 vector2                 siten;
283 float                   wsizex , wsizey;
284 short                   mval[];
285 {
286   int           i , j , minnum ;
287   double        dis , mindis;
288   float         x , y;
289
290   minnum = 0;
291   mindis = 10000.0;
292
293   for( i = 0 ; i < ip->pnum ; i++ ){
294     x = (float)ip->tgrid[i].x * wsizex - (float)siten.x;
295     y = (float)ip->tgrid[i].y * wsizey - (float)siten.y;
296
297     dis = sqrdb( (double)( x - (float)mval[0] ) ) + sqrdb( (double)( y -
298             (float)mval[1] ) );
299     dis = sqrt( dis );
300     if( dis < mindis ){
301       mindis = dis;
302       minnum = i;
303     }
304   }
305   return( minnum );
306 }
307
308 Chweight( ig , tpo , num , val , moto )
309 InflGroup               *ig;
310 int                     tpo , num;
311 double          val , moto;
312 {
313   int           i , j , max;
314   double        maxval;
315   if( moto > 0.0 ){
316     for( i = 0 ; i < (ig+tpo)->totcp ; i++ ){
317       if( (ig+tpo)->cpnum[i] == num ){
318         (ig+tpo)->cpweight[i] = val;
319       }
320     }
321   }else{
322     (ig+tpo)->cpnum[(ig+tpo)->totcp] = num;
323     (ig+tpo)->cpweight[(ig+tpo)->totcp] = val;
```



```
324     ((ig+tpo)->totcp)++;
325   }
326
327   maxval = 0.0 ; max = (ig+tpo)->near;
328   for( i = 0 ; i < (ig+tpo)->totcp ; i++ ){
329     if( maxval < (ig+tpo)->cpweight[i] ){
330       maxval = (ig+tpo)->cpweight[i];
331       max = (ig+tpo)->cpnum[i];
332     }
333   }
334   (ig+tpo)->near = max;
335
336   printf( "Done.. \n" );
337 }
338
339 double CPvalueget( ig , tpo , num )
340 InflGroup               *ig;
341 int                     tpo , num ;
342 {
343   int i;
344   for( i = 0 ; i < (ig+tpo)->totcp ; i++ )
345     if( (ig+tpo)->cpnum[i] == num ) return( (ig+tpo)->cpweight[i] );
346   return( -1.0 );
347 }
348
349 int Pointget( wf , siten , sizex , sizey , x , y , mval )
350 WaveFront               *wf;
351 vector2                 siten;
352 float                   sizex , sizey ;
353 long                    x , y;
354 short                   mval[];
355 {
356   int           i , j;
357   float         ipx , ipy;
358
359   ipx = ( (float)mval[0] + (float)siten.x ) / sizex;
360   ipy = ( (float)mval[1] + (float)siten.y ) / sizey;
361 /*
362   printf( "ipx = %f ipy = %f\n" , ipx , ipy );
363 */
364   i = NearestPoint( wf , (double)ipx , (double)ipy );
365
366   printf( "nearest = %d\n" , i );
367   return( i );
368 }
369
370 Pwrite( wf , num ,  siten , sizex , sizey , x , y , dot )
371 WaveFront               *wf;
372 vector2                 siten;
373 float                   sizex , sizey ;
374 long                    x , y;
375 int                     dot , num ;
376 {
377   float                 vert[2];
378
379   vert[0] = (float)( wf->tgrid[num].x ) * sizex - (float)siten.x;
380   vert[1] = (float)( wf->tgrid[num].y ) * sizey - (float)siten.y;
381
382   Color( RED );
383   pntsize( (short)dot );
384   bgnpoint();
385   v2f( vert );
386   endpoint();
387 }
388
389 CPwrite( wf , ip , ig , tpo , siten , wsizex , wsizey , x , y )
```

```
390 WaveFront               *wf;
391 InflPoint              *ip;
392 InflGroup              *ig;
393 int                    tpo;
394 vector2                siten;
395 float                  wsizex , wsizey;
396 long                   x , y;
397 {
398   int                  i , j;
399   float                vert[2];
400   float                ipx , ipy ;
401
402   AllCPwrite( ip , siten , wsizex , wsizey , x , y , DOT_ON_HUGE );
403
404   for( i = 0 ; i < (ig+tpo)->totcp ; i++ ){
405     vert[0] = (float)(ip->tgrid[(ig+tpo)->cpnum[i]].x) * wsizex - (float)siten.x
    ;
406     vert[1] = (float)(ip->tgrid[(ig+tpo)->cpnum[i]].y) * wsizey - (float)siten.y
    ;
407     if( (ig+tpo)->cpnum[i] == (ig+tpo)->near ) Color( BLUE );
408     else                                       Color( CYAN );
409     if( (ig+tpo)->cpweight[i] != 0.0 ){
410       pntsize( DOT_ON_HUGE );
411       bgnpoint();
412       v2f( vert );
413       endpoint();
414     }
415   }
416 }
417
418 CPInfAreaWrite( wf , ip , ig , tpo , siten , wsizex , wsizey , x , y )
419 InflPoint              *ip;
420 InflGroup              *ig;
421 int                    tpo;
422 vector2                siten;
423 float                  wsizex , wsizey;
424 long                   x , y;
425 WaveFront              *wf;
426 {
427   int                  i , j;
428   float                vert[2];
429
430   WFwriteOP( wf , siten , wsizex , wsizey , x , y , DOT_ON_MID , YELLOW ,
431             CLEAR_OFF );
432   AllCPwrite( ip , siten , wsizex , wsizey , x , y , DOT_ON_HUGE );
433
434   vert[0] = (float)ip->tgrid[tpo].x * wsizex - (float)siten.x;
435   vert[1] = (float)ip->tgrid[tpo].y * wsizey - (float)siten.y;
436
437   Color( BLUE );
438   pntsize( (short)DOT_ON_HUGE );
439   bgnpoint();
440   v2f( vert );
441   endpoint();
442
443   for( i = 0 ; i < wf->tgridnum ; i++ ){
444     for( j = 0 ; j < (ig+i)->totcp ; j++ ){
445       if( (ig+i)->cpnum[j] == tpo && (ig+i)->cpweight[j] > 0.0 ){
446         Pwrite( wf , i , siten , wsizex , wsizey , x , y , DOT_ON_MID );
447       }
448     }
449   }
450 }
451
452
453
```



```
454 #endif
455
456 /*
457 main()
458 {
459   char          filename[100];
460   WaveFront     wf;
461   int           i;
462
463   strcpy( filename , "/home/tatsu/atr/data/cyber/ochi_head_1.obj" );
464
465   ReadWaveFront( filename , &wf );
466
467   i = NearestPoint( &wf , 0.5 , 0.5 );
468
469   printf( "i = %d x = %f , y = %f\n" , i , wf.tgrid[i].x , wf.tgrid[i].y );
470
471   printf( "gnum = %d , lnum = %d\n" , wf.gridnum , wf.linknum );
472
473 }
474 */
475
476 void NormalizeIG( ig , gnum )
477 InflGroup       *ig;
478 int             gnum;
479 {
480   int           i , j;
481   double        total;
482
483   for( i = 0 ; i < gnum ; i++ ){
484     total = 0.0;
485     for( j = 0 ; j < (ig+i)->totcp ; j++ )
486       total += (ig+i)->cpweight[j];
487     if( total > 1.0 ){
488       for( j = 0 ; j < (ig+i)->totcp ; j++ )
489         (ig+i)->cpweight[j] /= total;
490     }
491   }
492 }
```

```
 1 /*
 2   TRIANGLE WEIGHT CALCULATE
 3   三角形の内部に限定した重みづけをするプログラム
 4   三角形の外部については、以前と同じ方法で重みづけする
 5   出力フォーマットも以前の weight file と同様
 6
 7   usage triweight [WaveFront file] [ControlPoint file] [(output)Weight file]
 8   (WaveFrontfileは拡張子をつけないで)
 9
10   1993.8.15(sun)  Tatsumi Sakaguchi
11 */
12
13 #include        <gl/gl.h>
14 #include        <gl/device.h>
15
16 #include        "std.h"
17 #include        "cimage.h"
18 #include        "wavefront.h"
19
20 main( argc , argv )
21 int             argc;
22 char            *argv[];
23 {
24   int           i = 0 , j = 0;
25   WaveFront     wf;
26   InflGroup     ig[MAXGRIDNUM];
27   InflPoint     ip;
28   ImageBuffer   *bufptr;
29   vector2       siten;
30
31   char          fname[100];
32   long          x , y , orgx , orgy;
33   Device        mdev[2] , dev;
34   short         mval[2] , val;
35   char          answer[10];
36   int           pushbutton = 0;
37   int           tpo[MAXLINKNUM][3];
38   float         vert[2];
39
40   bufptr = (ImageBuffer *)malloc( 512 * 512 * 4 );
41   siten.x = 0.0 ; siten.y = 0.0;
42
43   sprintf( fname , "%s.obj" , argv[1] );
44   ReadWaveFront( fname , &wf , 10 );
45   printf( "%s.obj file read OK!\n" , fname );
46
47   sprintf( fname , "%s.color" , argv[1] );
48   ImageLoad( fname , bufptr , 0 , 512 , 512 , ECHO_OFF );
49   printf( "%s.color file read OK!\n" , fname );
50
51   ReadControl( argv[2] , &ip );
52   printf( "%s file read OK!\n\n" , argv[2] );
53
54   ClearIG( &wf , ig );
55
56   foreground();
57
58   winopen( argv[1] );
59   RGBmode();
60   gconfig();
61
62   clear();
63
64   getsize( &x , &y );
65   getorigin( &orgx , &orgy );
66
```



```
67   rectzoom( x / 512.0 , y / 512.0 );
68   lrectwrite( 0 , 0 , 511 , 511 , (unsigned long *)bufptr );
69
70   WFwriteOP( &wf , siten , (float)x ,(float)y , x , y , DOT_ON_SMALL , YELLOW ,
71             CLEAR_OFF );
72   AllCPwrite( &ip , siten , (float)x , (float)y , x , y , DOT_ON_LARGE );
73
74   qdevice( MOUSEX );
75   qdevice( MOUSEY );
76   qdevice( MIDDLEMOUSE );
77   qdevice( LEFTMOUSE );
78   qdevice( RIGHTMOUSE );
79
80   mdev[0] = MOUSEX;
81   mdev[1] = MOUSEY;
82
83   while( 1 ){
84     switch( dev = qread( &val ) ){
85     case LEFTMOUSE:
86       if( pushbutton == 1 ){
87         getdev( 2 , mdev , mval );
88         mval[0] -= orgx;
89         mval[1] -= orgy;
90         tpo[j][i] = CPointget( &ip , siten , (float)x , (float)y , mval );
91
92         vert[0] = (float)(ip.tgrid[tpo[j][i]].x) * (float)x - (float)siten.x;
93         vert[1] = (float)(ip.tgrid[tpo[j][i]].y) * (float)y - (float)siten.y;
94
95         Color( RED );
96         pntsize( DOT_ON_LARGE );
97         bgnpoint();
98         v2f( vert );
99         endpoint();
100        i++;
101
102        if( i == 3 ){
103/*
104          Confirm( "Are these point correct?" , 512 , 512 );
105*/
106          input_text( "Are these point correct? ( y / n )" , answer );
107          if( strcmp( answer , "y" ) == 0 ){
108            MakeWeight( &wf , &ip , tpo[j] , ig , siten , x , y );
109            AllCPwrite( &ip , siten , (float)x , (float)y , x , y ,
110                        DOT_ON_LARGE );
111            j++ ; i = 0;
112          }else     i = 0;
113        }
114        printf( "Prease select Control point ( no.%d )\n" , i );
115        pushbutton = 0;
116      }else     pushbutton = 1;
117      break;
118    case MIDDLEMOUSE:
119      input_text( "Do you realy want to save and quit? ( y / n )"
120                , answer );
121      if( strcmp( answer , "y" ) == 0 ){
122        DivGroup( &wf , &ip , ig );
123        WriteGroup( argv[3] , wf.tgridnum , ig );
124        printf( "Done..\n" );
125        exit(1);
126      }
127      qreset();
128      printf( "command canceled ..\n" );
129      break;
130    default:
131      break;
132    }
```

```
133   ]
134 ]
135
136 MakeWeight( wf , ip , tpo , ig , siten , x , y )
137 WaveFront       *wf;
138 int             tpo[];
139 InflGroup       *ig;
140 InflPoint       *ip;
141 vector2         siten;
142 long            x , y;
143 {
144   int           i , j;
145   int           retval , maxw;
146   vector2       pos;
147   vector3       weight;
148   float         vert[2];
149
150   for( i = 0 ; i < wf->tgridnum ; i++ ){
151     if( (ig+i)->totcp == 0 ){
152       pos.x = wf->tgrid[i].x;
153       pos.y = wf->tgrid[i].y;
154       retval = JudgeTriangle( ip->tgrid[tpo[0]] , ip->tgrid[tpo[1]] ,
155                               ip->tgrid[tpo[2]] , pos );
156       if( retval == INNER_TRIANGLE ){
157         Pwrite( wf , i , siten , (float)x , (float)y , x , y ,
158                 DOT_ON_SMALL );
159         TriWeight( ip->tgrid[tpo[0]] , ip->tgrid[tpo[1]] , ip->tgrid[tpo[2]]
160                    , pos , &weight );
161         (ig+i)->totcp = 3;
162         if( weight.x > weight.y ){
163           if( weight.x > weight.z )     maxw = 0;
164           else                          maxw = 1;
165         }else{
166           if( weight.y > weight.z )     maxw = 1;
167           else                          maxw = 2;
168         }
169         (ig+i)->near = tpo[maxw];
170         (ig+i)->cpnum[0] = tpo[0];
171         (ig+i)->cpnum[1] = tpo[1];
172         (ig+i)->cpnum[2] = tpo[2];
173         (ig+i)->cpweight[0] = weight.x;
174         (ig+i)->cpweight[1] = weight.y;
175         (ig+i)->cpweight[2] = weight.z;
176       ]
177     ]
178   ]
179   Color( RED );
180   bgnclosedline();
181   vert[0] = (float)(ip->tgrid[tpo[0]].x) * (float)x - (float)siten.x;
182   vert[1] = (float)(ip->tgrid[tpo[0]].y) * (float)y - (float)siten.y;
183   v2f( vert );
184   vert[0] = (float)(ip->tgrid[tpo[1]].x) * (float)x - (float)siten.x;
185   vert[1] = (float)(ip->tgrid[tpo[1]].y) * (float)y - (float)siten.y;
186   v2f( vert );
187   vert[0] = (float)(ip->tgrid[tpo[2]].x) * (float)x - (float)siten.x;
188   vert[1] = (float)(ip->tgrid[tpo[2]].y) * (float)y - (float)siten.y;
189   v2f( vert );
190   endclosedline();
191
192 }
193
194 ClearIG( wf , ig )
195 InflGroup       *ig;
196 WaveFront       *wf;
197 {
198   int   i;
```



```
199
200   for( i = 0 ; i < wf->tgridnum ; i++ )
201     (ig+i)->totcp = 0;
202
203 ]
204
205
206
207
```

```
/* New standard I/O library       */
/* 1993.7.19 By Tatsumi Sakaguchi */

#ifdef IRIS
#include        <gl.h>
#include        <gl/image.h>
#endif

#include        "std.h"
#include        "cimage.h"

int ImageLoadPlane( filename , bufptr , size , mode , echob )
     char               *filename;
     ImageBuffer        *bufptr;
     int                size;
     int                mode , echob;
{
  int           i ;
  FILE          *fp;
  char          *ex;

  if( NULL == ( fp = fopen( filename , "r" )) ){
    if( echob == ECHO_ON ) printf( "Cannot open file .. %s (at ImageLoadPlane)\n" ,
                                   filename );
    return( FOPENERROR_CODE );
  }

  *ex = *(filename+(strlen(filename)-1));

  if( mode == RGB_MODE ){
    for( i = 0 ; i < size ; i++ ) (bufptr+i)->r = fgetc( fp );
    for( i = 0 ; i < size ; i++ ) (bufptr+i)->g = fgetc( fp );
    for( i = 0 ; i < size ; i++ ) (bufptr+i)->b = fgetc( fp );
  }else{
    switch( *ex ){
    case        'r':
      for( i = 0 ; i < size ; i++ ) (bufptr+i)->r = fgetc( fp );
      return( FSUCCESS_CODE );
    case        'g':
      for( i = 0 ; i < size ; i++ ) (bufptr+i)->g = fgetc( fp );
      return( FSUCCESS_CODE );
    case        'b':
      for( i = 0 ; i < size ; i++ ) (bufptr+i)->b = fgetc( fp );
      return( FSUCCESS_CODE );
    case        'y':
      for( i = 0 ; i < size ; i++ )
        (bufptr+i)->r = (bufptr+i)->g = (bufptr+i)->b = fgetc( fp );
      return( FSUCCESS_CODE );
    default:
      if( echob == ECHO_ON ) printf( "Illegal extention in %s.(at ImageLoadPlane)\n"
                                     , filename );
      return( FANOTHERERROR_CODE );
    }
  }
}

int ImageLoad( filename , bufptr , num , sizex , sizey , echob )
     char               *filename;
     ImageBuffer        *bufptr;
     int                sizex , sizey , num , echob;
{
  char  fname[200] , *ex;
  int   res , x , y , rflag = 0;

```



```
  ex = filename + ( strlen( filename ) - 2 );

  if( strcmp( ex , ".y" ) == 0 ){
    res = ImageLoadPlane( filename , bufptr , sizex * sizey , R_G_B_MODE , echob );
    if( res > 0 ) return( FERROR_CODE );
    else{
      printf( ".y gray scale standard format ,  read OK!\n" );
      return( FSUCCESS_CODE );
    }
  }

  strcpy( fname , filename );

  ex = filename + ( strlen( filename ) - 4 );

  if( strcmp( ex , ".rgb" ) != 0 && strcmp( ex , "olor" ) != 0 ){
    if( num == 0 ){
      sprintf( fname , "%s.r" , filename );
      res = ImageLoadPlane( fname , bufptr , sizex * sizey , R_G_B_MODE , echob );
      sprintf( fname , "%s.g" , filename );
      res += ImageLoadPlane( fname , bufptr , sizex * sizey , R_G_B_MODE , echob );
      sprintf( fname , "%s.b" , filename );
      res += ImageLoadPlane( fname , bufptr , sizex  * sizey, R_G_B_MODE , echob );
    }else{
      sprintf( fname , "%s%03d.r" , filename );
      res = ImageLoadPlane( fname , bufptr , sizex  * sizey, R_G_B_MODE , echob );
      sprintf( fname , "%s%03d.g" , filename );
      res += ImageLoadPlane( fname , bufptr , sizex * sizey , R_G_B_MODE , echob );
      sprintf( fname , "%s%03d.b" , filename );
      res += ImageLoadPlane( fname , bufptr , sizex * sizey , R_G_B_MODE , echob );
    }
    if( res == 0 ){
      printf( ".r .g .b standard format , read OK!\n" );
      return( FSUCCESS_CODE );
    }else{
      sprintf( fname , "%s.rgb" , filename );
      rflag = 1;
    }
  }

  if( strcmp( ex , ".rgb" ) == 0 || rflag == 1 ){
    res = ImageLoadPlane( fname , bufptr , sizex * sizey , RGB_MODE , echob );
    if( res == 0 ){
      printf( ".rgb standard format , read OK!\n" );
      return( FSUCCESS_CODE );
    }
    if( res > 0 && rflag == 0 ) return( FERROR_CODE );
    sprintf( fname , "%s.color" , filename );
  }

#ifdef IRIS
  if( strcmp( ex , "olor" ) == 0 || rflag == 1 ){
        res = IrisImageLoad( fname , bufptr , &x , &y , echob );
    if( res == 0 ){
      printf( ".color IRIS format , read OK!\n" );
      return( FSUCCESS_CODE );
    }
  }
#endif
```

```
124
125     return( FERROR_CODE );
126 }
127
128 #ifdef IRIS
129 int IrisImageLoad( filename , bufptr , xsize , ysize , echob )
130      char               *filename;
131      ImageBuffer        *bufptr;
132      int                *xsize , *ysize , echob;
133 {
134   IMAGE         *imagefd;
135   int           data_xsize , data_ysize , data_zsize;
136   short         rbuf[4096] , gbuf[4096] , bbuf[4096];
137   short int     *rpl , *gpl , *bpl;
138   int           i , j;
139
140   if(( imagefd=iopen( filename , "r" ) ) == NULL ){
141     if( echob == ECHO_ON ) printf( "Cannot open file .. %s (at IrisImageLoad)\n"

142                                    , filename );
143     return( FOPENERROR_CODE );
144   }
145
146     data_xsize = *xsize = imagefd->xsize;
147     data_ysize = *ysize = imagefd->ysize;
148     data_zsize = imagefd->zsize;
149
150     for( i = 0 ; i < data_ysize ; i++ ){
151       getrow( imagefd , rbuf , i , 0 );
152       getrow( imagefd , gbuf , i , 1 );
153       getrow( imagefd , bbuf , i , 2 );
154       rpl = rbuf;
155       gpl = gbuf;
156       bpl = bbuf;
157       for( j = 0 ; j < data_xsize ; j++ ){
158         (bufptr+j+data_xsize*i)->r = (unsigned char)(*rpl++);
159         (bufptr+j+data_xsize*i)->g = (unsigned char)(*gpl++);
160         (bufptr+j+data_xsize*i)->b = (unsigned char)(*bpl++);
161       }
162     }
163   return( FSUCCESS_CODE );
164 }
165 #endif
166
167 int ImageSave( filename , bufptr , num , sizex , sizey , echob , mode )
168      char               *filename;
169      ImageBuffer        *bufptr;
170      int                num , sizex , sizey , echob , mode;
171 {
172   FILE                  *fp;
173   int                   i , j;
174   char                  fname[100];
175
176   switch( mode ){
177   case R_G_B_MODE:
178     sprintf( fname , "%s.r" , filename );
179     if( NULL == ( fp = fopen( fname , "w" ) ) ){
180       if( echob == ECHO_ON )
181         printf( "Cannot open file .. %s (ImageSave@stdioI)\n" , fname );
182       return( FOPENERROR_CODE );
183     }
184     for( i = 0 ; i < sizex * sizey ; i++ )
185       fputc( (bufptr+i)->r , fp );
186     fclose( fp );
187
188     sprintf( fname , "%s.g" , filename );
```



```
189     if( NULL == ( fp = fopen( fname , "w" ) ) ){
190       if( echob == ECHO_ON )
191         printf( "Cannot open file .. %s (ImageSave@stdioI)\n" , fname );
192       return( FOPENERROR_CODE );
193     }
194     for( i = 0 ; i < sizex * sizey ; i++ )
195       fputc( (bufptr+i)->g , fp );
196     fclose( fp );
197
198     sprintf( fname , "%s.b" , filename );
199     if( NULL == ( fp = fopen( fname , "w" ) ) ){
200       if( echob == ECHO_ON )
201         printf( "Cannot open file .. %s (ImageSave@stdioI)\n" , fname );
202       return( FOPENERROR_CODE );
203     }
204     for( i = 0 ; i < sizex * sizey ; i++ )
205       fputc( (bufptr+i)->b , fp );
206     fclose( fp );
207
208     return( FSUCCESS_CODE );
209     break;
210   default:
211     break;
212   }
213 }
214
215
216
217
218
219
220
221
222
223
```

```
 1 #include        <stdio.h>
 2 #include        <sys/types.h>
 3 #include        <sys/dir.h>
 4 #include        <sys/file.h>
 5
 6 #define         OPEN_ERROR      0
 7 #define         READ_ERROR      -1
 8
 9 Bell( n )
10 int     n;
11 {
12         int     i;
13
14         for( i = 0 ; i < n ; i++ ) printf( "%c\n" , 0x7 );
15 }
16
17 input_text( text , text1 )
18 char            text[],text1[];
19 {
20         printf( "%s .. " , text );
21         scanf( "%s" , text1 );
22 }
23
24 input_int( text , num )
25 char            text[];
26 int             *num;
27 {
28         printf("%s .. " , text );
29         scanf( "%d" , num );
30 }
31
32 float avr( buf , num )
33 int             buf[];
34 int             num;
35 {
36         float   tot = 0.0 ;
37         int     i;
38
39         for( i = 0 ; i < num ; i++ ){
40                 tot = tot + (float)abs(buf[i]) ;
41         }
42         return( tot / (float)num );
43 }
44
45 float bunsan( buf , num , avr )
46 int             buf[];
47 int             num;
48 float           avr;
49 {
50         float   tot = 0.0;
51         float   sb , sa;
52         int     i;
53
54         for( i = 0 ; i < num ; i++ ){
55                 sb = (float)abs(buf[i]) - avr;
56                 tot = tot + ( sb * sb ) ;
57         }
58         return( tot / (float)num  );
59 }
60
61 input_double( text , num )
62      char       text[];
63      double     *num ;
64 {
65   float fl;
66
```



```
67   printf( "%s .. " , text );
68   scanf( "%f" , &fl );
69   *num = (double)fl;
70 }
71
72 input_float( text , num )
73      char       text[];
74      float      *num ;
75 {
76   printf( "%s .. " , text );
77   scanf( "%f" , num );
78 }
```

```
#include                "std.h"
#include                "cmath.h"

double          Gaiseki( );
double          Setten();

main()
{
  vector2       fav , sav , fcv , fbv , scv , sbv , fxv , sxv;
  vector3       bv , cv , xv , hv , ov , av;
  vector3       oav , obv , ocv , onv , resv , oxv;
  vector3       cyhv;

  double          cya , cyb , cyc , cynorm_value , cynvalue;
  double          cybvalue , cycvalue;

  char          fnamei[100] , fnameo[100];

  double        norm_value , nvalue , bvalue , cvalue;
  double        a , b , c;

  double        ganma1 , ganma2 , cent1 , cent2;

/* ファイルから規準ベクトルを読み込む */

  strcpy( fnamei , "../data/testvector.2d" );
  strcpy( fnameo , "../data/testvector.cyber" );

  kvector_fromfile( fnamei , &fav , &fbv , &fcv , &sav , &sbv , &scv );

/* 対象ベクトル設定 */

  fxv.x = 374.0;
  fxv.y = 296.0;
  sxv.x = 162.0;
  sxv.y = 321.0;
/*
  printf( "fav = (%f,%f) , fbv = (%f,%f) \n" , fav.x,fav.y,fbv.x,fbv.y);
  printf( "fcv = (%f,%f) , fxv = (%f,%f) \n" , fcv.x,fcv.y,fxv.x,fxv.y);
  printf( "sav = (%f,%f) , sbv = (%f,%f) \n" , sav.x,sav.y,sbv.x,sbv.y);
  printf( "scv = (%f,%f) , sxv = (%f,%f) \n" , scv.x,scv.y,sxv.x,sxv.y);
*/
/* 画角による視差の修正 */

  ganma1 = 0.047126792;
  ganma2 = 0.047126792;

  cent1 = 223.0;
  cent2 = 220.0;

  chdimension( fav , sav , ganma1 , ganma2 , cent1 , cent2 , &av );
  chdimension( fbv , sbv , ganma1 , ganma2 , cent1 , cent2 , &bv );
  chdimension( fcv , scv , ganma1 , ganma2 , cent1 , cent2 , &cv );
  chdimension( fxv , sxv , ganma1 , ganma2 , cent1 , cent2 , &xv );

  printf( "*** 正規化前 ***\n" );
  printf( "av = (%f,%f,%f)\n" , av.x , av.y , av.z );
  printf( "bv = (%f,%f,%f)\ncv = (%f,%f,%f)\nxv = (%f,%f,%f)\n",
                bv.x,bv.y,bv.z,cv.x,cv.y,cv.z,xv.x,xv.y,xv.z);

  CoordConvert3( &av , &bv , &cv , &xv );
  printf( "*** 正規化後 ***\n" );

  printf( "av = (%f,%f,%f)\n" , av.x , av.y , av.z );
  printf( "bv = (%f,%f,%f)\ncv = (%f,%f,%f)\nxv = (%f,%f,%f)\n",
                bv.x,bv.y,bv.z,cv.x,cv.y,cv.z,xv.x,xv.y,xv.z);
```



```
  printf( "Value of bv = %f\n" , sqrt( bv.x * bv.x + bv.y * bv.y +
                                       bv.z * bv.z )  );
  printf( "Value of cv = %f\n" , sqrt( cv.x * cv.x + cv.y * cv.y +
                                       cv.z * cv.z )  );

/* 法線ベクトルの設定 */

  norm_value = Gaiseki( bv , cv , &ov );
  printf( "ov = (%f,%f,%f)\n" , ov.x , ov.y , ov.z );
  printf( "norm_value(ov) = %f \n" , norm_value );

/* 平面の方程式決定 */

  Plane( ov , norm_value , &a , &b , &c );

/* 平面と直線の交点を求める */

  nvalue = Setten( norm_value , xv , a , b , c , &hv );

/* 平面内での各成分への分解 */

  Normalize( bv , cv , hv , &bvalue , &cvalue );

  printf( "bval = %f, cval = %f, nval = %f\n" , bvalue,cvalue,nvalue);

/* サイバーの出力との比較 */

  ovector_fromfile( fnameo , &oav , &obv , &ocv , &onv );

  resv.x = obv.x * bvalue + ocv.x * cvalue + onv.x * nvalue ;
  resv.y = obv.y * bvalue + ocv.y * cvalue + onv.y * nvalue ;
  resv.z = obv.z * bvalue + ocv.z * cvalue + onv.z * nvalue ;

  printf( "\n3D a vector ( %f , %f , %f )\n" , oav.x , oav.y , oav.z );
  printf( "3D b vector ( %f , %f , %f )\n" , obv.x , obv.y , obv.z );
  printf( "3D c vector ( %f , %f , %f )\n" , ocv.x , ocv.y , ocv.z );
  printf( "3D n vector ( %f , %f , %f )\n" , onv.x , onv.y , onv.z );
  printf( "Result vector( %f , %f , %f )\n" , resv.x , resv.y ,
          resv.z );

  printf( "Result vector + a ( %f , %f , %f )\n" , oav.x + resv.x,
           resv.y + oav.y , oav.z - resv.z );

  printf( "Value of obv = %f\n" , sqrt( obv.x * obv.x + obv.y * obv.y +
                                        obv.z * obv.z ));
  printf( "Value of ocv = %f\n" , sqrt( ocv.x * ocv.x + ocv.y * ocv.y +
                                        ocv.z * ocv.z ));

/* サイバーでの係数を求める */

  oxv.x = 49.598448;
  oxv.y = -22.400282;
  oxv.z = 98.541856;

  oxv.x = oxv.x - oav.x; oxv.y = oxv.y - oav.y; oxv.z = oav.z - oxv.z;
  printf( "\noxv = (%f,%f,%f)\n" , oxv.x , oxv.y , oxv.z );

  cynorm_value = Gaiseki( obv , ocv , &onv );

  printf( "onv = (%f,%f,%f)\n" , onv.x , onv.y , onv.z );

  printf( "norm_value(onv) = %f \n" , cynorm_value );

  Plane( onv , cynorm_value , &cya , &cyb , &cyc );
  cynvalue = Setten( cynorm_value , oxv , cya , cyb , cyc , &cyhv );
  Normalize( obv , ocv , cyhv , &cybvalue , &cycvalue );
```

```
133   printf( "bvalue = %f , cvalue = %f , nvalue = %f\n" ,
134                cybvalue , cycvalue , cynvalue );
135
136 }
137
138 ovector_fromfile( fname , av , bv , cv , nv )
139 char    *fname;
140 vector3 *av , *bv , *cv , *nv;
141 {
142   FILE          *fp;
143   float         x , y , z;
144
145   if( NULL == ( fp = fopen( fname , "r" ))){
146     printf( "Cannot open file %s (in ovector_fromfile)\n" , fname );
147     exit(1);
148   }
149
150   fscanf( fp , "%f %f %f" , &x , &y , &z );
151   av->x = (double)x ; av->y = (double)y ; av->z = (double)z;
152   fscanf( fp , "%f %f %f" , &x , &y , &z );
153   bv->x = (double)x ; bv->y = (double)y ; bv->z = (double)z;
154   fscanf( fp , "%f %f %f" , &x , &y , &z );
155   cv->x = (double)x ; cv->y = (double)y ; cv->z = (double)z;
156   fscanf( fp , "%f %f %f" , &x , &y , &z );
157   nv->x = (double)x ; nv->y = (double)y ; nv->z = (double)z;
158
159   fclose( fp );
160 }
161
162 kvector_fromfile( fname , fav , fbv , fcv , sav , sbv , scv )
163 char    *fname;
164 vector2 *fav , *fbv , *fcv , *sav , *sbv , *scv;
165 {
166   FILE          *fp;
167   float         x , y;
168
169   if( NULL == (fp = fopen( fname , "r" ) )) {
170     printf( "Cannot open file %s (in kvector_fromfile)\n" , fname );
171     exit(1);
172   }
173
174   fscanf( fp , "%f %f" , &x , &y );
175   fav->x = (double)x ; fav->y = (double)y;
176   fscanf( fp , "%f %f" , &x , &y );
177   fbv->x = (double)x ; fbv->y = (double)y;
178   fscanf( fp , "%f %f" , &x , &y );
179   fcv->x = (double)x ; fcv->y = (double)y;
180   fscanf( fp , "%f %f" , &x , &y );
181   sav->x = (double)x ; sav->y = (double)y;
182   fscanf( fp , "%f %f" , &x , &y );
183   sbv->x = (double)x ; sbv->y = (double)y;
184   fscanf( fp , "%f %f" , &x , &y );
185   scv->x = (double)x ; scv->y = (double)y;
186
187   fclose( fp );
188 }
189
190
191
```

```
#include <stdio.h>
#include <stdarg.h>
#include <sys/types.h>
#include <sys/stat.h>
#include <alloca.h>
#include "img.h"

void
readImageFile(name, w, h, img)
char            *name;
int             w, h;
unsigned long   *img;
{
    FILE        *rfp, *gfp, *bfp;
    int         i, j;
    long        xy, siz, t;
    struct stat st;
    static char buff[256];

    sprintf(buff, "%s.r", name);
    rfp = fopen(buff, "r");
    if ( rfp == NULL ) {
        errorExit(0, "%s cannot open.\n", buff);
    }

    sprintf(buff, "%s.g", name);
    gfp = fopen(buff, "r");
    if ( rfp == NULL ) {
        errorExit(0, "%s cannot open.\n", buff);
    }

    sprintf(buff, "%s.b", name);
    bfp = fopen(buff, "r");
    if ( rfp == NULL ) {
        errorExit(0, "%s cannot open.\n", buff);
    }

    for ( i = 0; i < h; i++ ) {
        t = i * w;
        for ( j = 0; j < w; j++ ) {
            img[t] = fgetc(rfp) + (fgetc(gfp) << 8) + (fgetc(bfp) << 16);
            t++;
        }
    }
    fclose(rfp);        fclose(gfp);    fclose(bfp);
}


void
toColor(mono, x, y, color)
unsigned char   *mono;
int             x, y;
unsigned long   *color;
{
    long        i, xy;
    unsigned char       c;

    xy = x * y;
    for ( i = 0; i < xy; i++ ) {
        c = mono[i];
        color[i] = c + (c << 8) + (c << 16);
    }
]

toHueColor(hue, x, y, color)
unsigned char   *hue;
```



```
int             x, y;
unsigned long   *color;
{
    long        i, xy;
    unsigned char       c;
    unsigned char       r, g, b;

    xy = x * y;
    for ( i = 0; i < xy; i++ ) {
        c = hue[i];
        convHSVToRGB(c, 0xff, 0x80, &r, &g, &b);
        color[i] = r + (g << 8) + (b << 16);
    }
}


void
toMono(orig, w, h, mode, dest)
unsigned long *orig;
int     w, h;
char    mode;
unsigned char *dest;
{
  long          xy;
  long          i;
  unsigned short        c;
  unsigned char h0, v0, c0;
  unsigned char *p;

  xy = w * h;
  if ( mode == 'H' || mode == 'V' || mode =='S' ) {
    if ( mode == 'H' )          p = &h0;
    else if ( mode == 'V' )     p = &v0;
    else                        p = &c0;
    for ( i = 0; i < xy; i++ ) {
      convRGBToHSV(RLevel(orig[i]),GLevel(orig[i]), BLevel(orig[i]),
                   &h0, &c0, &v0);
      dest[i] = (short)((*p));
    }
  } else if ( mode == '3' ) {
    for ( i = 0; i < xy; i++ ) {
      dest[i] = (short)(((short)RLevel(orig[i]) +
                         (short)GLevel(orig[i]) +
                         (short)BLevel(orig[i])) / 3);
    }
  } else if ( mode == 'G' ) {
    for ( i = 0; i < xy; i++ ) {
      dest[i] = ToGray(orig[i]);
    }
  }
}

void
readMonoFile( name, mode, w, h, color, mono)
char            *name;
char             mode;
int              w, h;
unsigned long   *color;
unsigned char   *mono;
{
    FILE        *fp;
    char        sfx;
    int         i, j;
    long        xy, siz, t;
    struct stat st;
    static char buff[256];
```

```
133
134     switch ( mode ) {
135       case 'H' :
136           sfx = 'h'; break;
137         case 'V' :
138             sfx = 'v'; break;
139         case 'S' :
140             sfx = 's'; break;
141         case '3' :
142             sfx = 'm'; break;
143         case 'G' :
144             sfx = 'n'; break;
145         default:
146           return;
147     }
148
149     sprintf(buff, "%s.%c", name, sfx);
150     fp = fopen(buff, "r");
151     if ( fp == NULL ) {
152       if ( color == NULL ) {
153           color = alloca(sizeof(unsigned long) * w * h);
154           readImageFile(name, w, h, color);
155       }
156       toMono( color, w, h, mode, mono);
157       fp = fopen(buff, "w");
158       fwrite( mono, sizeof(unsigned char), w * h, fp);
159     } else {
160       fread( mono, sizeof(unsigned char), w * h, fp);
161     }
162     fclose(fp);
163 }
```

```
 1
 2
 3
 4 /* IMAGE DISPLAY */
 5 /* imgdisp       */
 6
 7 #include        <gl/gl.h>
 8 #include        <gl/device.h>
 9
10 #include        "std.h"
11 #include        "cimage.h"
12 #define X       0
13 #define Y       1
14 #define XY      2
15
16 main()
17 {
18   char  fname[100];
19   int   i = 1  , j;
20   ImageBuffer *bufptr;
21   short val;
22   Device mdev[XY];
23   short mval[XY];
24   long org[XY];
25
26   bufptr = (ImageBuffer *)malloc( 512 * 512 * 4 );
27
28   input_text( "File name" , fname );
29
30   if( FSUCCESS_CODE != ImageLoad( fname , bufptr , 0 , 512 , 512 , ECHO_OFF )) e
   xit(1);
31
32
33   prefsize( 512 , 512 );
34   winopen( fname );
35
36   RGBmode();
37   gconfig();
38   clear();
39   qdevice( TIMER0 );
40   getorigin(&org[X], &org[Y]);
41 /*
42   pixmode( PM_TTOB , 1 );
43 */
44   lrectwrite( 0 , 0 , 511 , 511 , (unsigned long *)bufptr );
45
46   noise( TIMER0 , getgdesc( GD_TIMERHZ) / 10 );
47
48   mdev[X] = MOUSEX;
49   mdev[Y] = MOUSEY;
50
51   gflush();
52   while( 1 ) {
53     if( qread( &val )) {
54       getdev( XY , mdev , mval );
55       printf( "x = %d , y = %d\n" , mval[X] - org[X], mval[Y] - org[Y] );
56     }
57   }
58 }
```

```
 1 #include        "std.h"
 2 #include        "wavefront.h"
 3
 4 main( argc , argv )
 5 int             argc;
 6 char            *argv[];
 7 {
 8   WaveFront             wf;
 9   InflPoint             ip;
10   InflGroup             ig[MAXGRIDNUM];
11
12   ReadWaveFront( argv[1] , &wf , 10 );
13
14   ReadControl( argv[2] , &ip );
15
16   DivGroup( &wf , &ip , ig );
17
18   WriteGroup( argv[3] , wf.tgridnum , ig );
19
20   printf( "Done.\n" );
21 }
```

```
#include                "std.h"
#include                "cmath.h"

#define         MAXPOINTNUM     100
#define         MAXFRAMENUM     100

double          Gaiseki( );
double          Setten();

main()
{
  vector3       av , bv , cv , ov;
  vector3       objectvp[MAXPOINTNUM] , objectv[MAXPOINTNUM][MAXFRAMENUM];

  double        nvalue , bvalue , cvalue;
  float         x , y , z;

  FILE          *fp , *fpi;

  int           i , j ;
  int           pnum , fnum;

  char          pname[MAXPOINTNUM][50];

/* 出力用のファイルオープン */

  if( NULL == ( fp = fopen( "movement" , "w" ) ) ){
    printf( "Cannot open file .. movement (exp5:main)\n" );
    exit(1);
  }

/* 入力用のファイルオープン */

  if( NULL == ( fpi = fopen( "lvector" , "r" ) ) ){
    printf( "Cannot open file .. lvector (exp5:main)\n" );
    exit(1);
  }

/* lvector ファイルから基本情報を得る */

  fscanf( fpi , "%d %d" , &fnum , &pnum );
  fscanf( fpi , "%f %f %f" , &x , &y , &z );
  av.x = (double)x ; av.y = (double)y ; av.z = (double)z;
  fscanf( fpi , "%f %f %f" , &x , &y , &z );
  bv.x = (double)x ; bv.y = (double)y ; bv.z = (double)z;
  fscanf( fpi , "%f %f %f" , &x , &y , &z );
  cv.x = (double)x ; cv.y = (double)y ; cv.z = (double)z;

  for( i = 0 ; i < pnum ; i++ ) fscanf( fpi , "%s" , pname[i] );

/* movement ファイルに基本情報を書き込む */

  fprintf( fp , "%d %d\n" , fnum , pnum );

/* 基本情報を基に外積ベクトルおよびベクトルの大きさを求める */

  nvalue = Gaiseki( bv , cv , &ov );
  bvalue = sqrt(sqrdb( bv.x ) + sqrdb( bv.y ) + sqrdb( bv.z ));
  cvalue = sqrt(sqrdb( cv.x ) + sqrdb( cv.y ) + sqrdb( cv.z ));

/* 1フレーム目のデータの読み込み */

  for( i = 0 ; i < pnum - 3  ; i++ ){
    fscanf( fpi , "%f %f %f" , &x , &y , &z );
    objectvp[i].x = (double)x * bv.x + (double)y * cv.x + (double)z * ov.x;
    objectvp[i].y = (double)x * bv.y + (double)y * cv.y + (double)z * ov.y;
```



```
    objectvp[i].z = (double)x * bv.z + (double)y * cv.z + (double)z * ov.z;
  }

/* 2フレーム目以降のデータの読み込みと計算 */

  for( j = 1 ; j < fnum ; j++ ){
    for( i = 0 ; i < pnum - 3 ; i++ ){
      fscanf( fpi , "%f %f %f" , &x , &y , &z );
      objectv[i][j].x = (double)x * bv.x + (double)y * cv.x + (double)z * ov.x;
      objectv[i][j].y = (double)x * bv.y + (double)y * cv.y + (double)z * ov.y;
      objectv[i][j].z = (double)x * bv.z + (double)y * cv.z + (double)z * ov.z;
      objectv[i][j].x -= objectvp[i].x;
      objectv[i][j].y -= objectvp[i].y;
      objectv[i][j].z -= objectvp[i].z;
    }
  }
  fclose( fpi );

/* 2フレーム目以降のデータの書きだし */

  for( i = 0 ; i < pnum -3 ; i++ ){
    fprintf( fp , "\n%s\n" , pname[3+i] );
    fprintf( fp , "1\t%f\t%f\t%f\n" ,  objectvp[i].x , objectvp[i].y  ,
             objectvp[i].z );
    for( j = 1 ; j < fnum ; j++ ){
      fprintf( fp , "%d\t%f\t%f\t%f\t\n" , j+1 , objectv[i][j].x ,
               objectv[i][j].y , objectv[i][j].z );
    }
  }

  fclose( fpi );
}
```

```
 1 #include                "std.h"
 2 #include                "cmath.h"
 3
 4 #define          MAXPOINTNUM     100
 5
 6 double           Gaiseki( );
 7 double           Setten();
 8
 9 main()
10 {
11   vector2        fv[MAXPOINTNUM] , sv[MAXPOINTNUM];
12   vector3        cyv[MAXPOINTNUM];
13
14   vector2        fav , sav , fcv , fbv , scv , sbv , fxv , sxv;
15   vector3        bv , cv , xv , hv , ov , av;
16   vector3        oav , obv , ocv , onv , resv , oxv;
17   vector3        cyhv;
18
19   double           cya , cyb , cyc , cynorm_value , cynvalue;
20   double           cybvalue , cycvalue;
21
22   char           fnamef[100] , fnames[100] , fnamec[100];
23   int            basicframe;
24
25   double         norm_value , nvalue , bvalue , cvalue;
26   double         a , b , c;
27
28   double         ganma1 , ganma2 , cent1 , cent2;
29
30   FILE           *fp;
31
32   int            pnum , i , j , fnum;
33   char           pname[MAXPOINTNUM][50];
34
35 /* 出力用ファイルオープン */
36
37   if( NULL == ( fp = fopen( "lvector" , "w" ) ) ){
38     printf( "Cannot open file .. lvector (exp4:main)\n" );
39     exit(1);
40     }
41
42 /* 入力データのフレーム数 */
43   printf( "kokomadeha kiteruyo\n" );
44   input_int( "number of frames" , &fnum );
45   input_int( "Basic Frame No. " , &basicframe );
46   fprintf( fp , "%d " , fnum );
47
48 /* 画角による視差の修正の為の各パラメータ値設定 */
49
50   ganma1 = 0.047126792;
51   ganma2 = 0.047126792;
52
53   cent1 = 245.0;
54   cent2 = 224.55;
55
56 /* 基準画像の基準ベクトルを算出・出力 */
57
58   sprintf( fnamef , "../../data/fgrid/front%03d.grid" , basicframe );
59   sprintf( fnames , "../../data/sgrid/side%03d.grid" , basicframe );
60
61   kvector_fromfile( fnamef , fv , pname , &pnum);
62   kvector_fromfile( fnames , sv , pname , &pnum);
63
64   chdimension( fv[0] , sv[0] , ganma1 , ganma2 , cent1 , cent2 , &av );
65   chdimension( fv[1] , sv[1] , ganma1 , ganma2 , cent1 , cent2 , &bv );
66   chdimension( fv[2] , sv[2] , ganma1 , ganma2 , cent1 , cent2 , &cv );
```



```
67
68   CoordConvertB( &av , &bv , &cv );
69
70   fprintf( fp , "%d\n" , pnum );
71   fprintf( fp , "%f %f %f\n" , av.x , av.y , av.z );
72   fprintf( fp , "%f %f %f\n" , bv.x , bv.y , bv.z );
73   fprintf( fp , "%f %f %f\n" , cv.x , cv.y , cv.z );
74   for( i = 0 ; i < pnum ; i++ )     fprintf( fp , "%s\n" , pname[i] );
75
76 /************** フレームループ ***************/
77
78   for( j = 1 ; j <= fnum ; j++ ){
79
80 /* ファイルから各ベクトルを読み込む */
81
82     sprintf( fnamef , "../../data/fgrid/front%03d.grid" , j );
83     sprintf( fnames , "../../data/sgrid/side%03d.grid" , j );
84
85     kvector_fromfile( fnamef , fv , pname , &pnum);
86     kvector_fromfile( fnames , sv , pname , &pnum);
87
88 /* 規準ベクトルの正規化 */
89
90     chdimension( fv[0] , sv[0] , ganma1 , ganma2 , cent1 , cent2 , &av );
91     chdimension( fv[1] , sv[1] , ganma1 , ganma2 , cent1 , cent2 , &bv );
92     chdimension( fv[2] , sv[2] , ganma1 , ganma2 , cent1 , cent2 , &cv );
93
94     CoordConvertB( &av , &bv , &cv );
95
96 /* 法線ベクトルの設定 */
97
98     norm_value = Gaiseki( bv , cv , &ov );
99
100 /* 平面の方程式決定 */
101
102     Plane( ov , norm_value , &a , &b , &c );
103
104
105 /************ ポイントループ ************/
106
107     for( i = 3 ; i < pnum ; i++ ){
108
109       chdimension( fv[i] , sv[i] , ganma1 , ganma2 , cent1 , cent2 , &xv );
110       CoordConvertX( &av , &xv );
111
112 /* 平面と直線の交点を求める */
113
114       nvalue = Setten( norm_value , xv , a , b , c , &hv );
115
116 /* 平面内での各成分への分解 */
117
118       Normalize( bv , cv , hv , &bvalue , &cvalue );
119
120 /* ファイルへの情報出力 */
121
122       fprintf( fp , "%f %f %f\n" , bvalue , cvalue , nvalue );
123
124     }
125
126 /**** The end of ポイントループ **********/
127
128   }
129
130 /**** the end of フレームループ **********/
131
132   fclose( fp );
```

```
133 }
134
135 kvector_fromfile( fname , v , pname , pnum )
136 char    *fname , pname[MAXPOINTNUM][50];
137 vector2 *v;
138 int     *pnum;
139 {
140   FILE          *fp;
141   float         x , y;
142   int           i;
143   vector2       *tmpv;
144
145   tmpv = v;
146
147   if( NULL == (fp = fopen( fname , "r" ) )) {
148     printf( "Cannot open file %s (in kvector_fromfile)\n" , fname );
149     exit(1);
150   }
151
152   fscanf( fp , "%d" , pnum );
153 /*  printf( "pnum = %d\n" , *pnum );
154 */
155   for( i = 0 ; i < *pnum ; i++ ){
156     fscanf( fp , "%s %f %f" , pname[i] , &x , &y );
157     v->x = (double)x ;
158     (v++)->y = (double)y;
159 /*  printf( "x = %f , y = %f\n" , x , y );
160   printf( "name = %s : (%f,%f)\n" , pname[i] , (v-1)->x , (v-1)->y );
161 */  }
162   v = tmpv;
163   fclose( fp );
164 }
165
166
167
168
```

```
  1 /*
  2    IMAGE DISPLAY & GET COODINATES OF FEATURE POINT
  3    FOR EXPERIMENT , RECONSTRUCT 3D-COODINATES FROM STEREO IMAGE
  4
  5    CODING : 1993.7.28 T.Sakaguchi @ ATR Communication System Research Lab.
  6    MACHINE: IRIS Series with Graphics Library
  7 */
  8
  9 #include        <gl/gl.h>
 10 #include        <gl/device.h>
 11
 12 #include        "std.h"
 13 #include        "cimage.h"
 14 #include        "cmath.h"
 15
 16 #define X       0
 17 #define Y       1
 18 #define XY      2
 19
 20 #define GRAY40          64
 21 #define POINTNUM        60
 22 #define ZOOM            10
 23
 24 long          mainwindowid , textwindowid , zoomwindowid;
 25 long          readsize[XY] , size[XY] , orgz[XY];
 26
 27 main()
 28 {
 29   char  fname[100] , modec[10];
 30   char  *filenum , filename[100];
 31
 32   int   i  , j , mode , pnum = 3 , eflag = 1;
 33   ImageBuffer *bufptr;
 34   short val;
 35   Device mdev[XY];
 36   short mval[XY];
 37   long org[XY];
 38   int pushflag = 0;
 39
 40   vector3       ov[POINTNUM] , av , xv;
 41   vector2       fv[POINTNUM] , sv[POINTNUM] , sav , sxv;
 42   int           np;
 43   float         xt , yt , zt;
 44   char          dum[50];
 45   FILE          *fp , *fp2;
 46   char          pname[100][50];
 47
 48   bufptr = (ImageBuffer *)malloc( 512 * 512 * 4 );
 49
 50   strcpy( pname[0] , "top_of_nose_(_point_A_)" );
 51   strcpy( pname[1] , "forehead_center_(_point_B_)" );
 52   strcpy( pname[2] , "ear_center_(_point_C_)" );
 53
 54   input_text( "File name" , fname );
 55   input_text( "Which Mode f)ront view or s)ide view" , modec );
 56
 57   filenum = fname + strlen( fname ) - 3;
 58
 59   if( FSUCCESS_CODE != ImageLoad( fname , bufptr , 0 , 512 , 512 , ECHO_OFF )) e
    xit(1);
 60
 61   if( modec[0] == 's' ){
 62     sprintf( filename , "/home/tatsu/atr/data/fgrid/front%s.grid" , filenum );
 63     printf( "%s\n" , filename );
 64     if( NULL == (fp = fopen( "/home/tatsu/atr/data/cyber.grid" , "r" ))) exit(2)
    ;
```



```
 65     printf( "%s\n" , filename );
 66     if( NULL == (fp2 = fopen( filename  , "r" ))) exit(1);
 67     printf( "%s\n" , filename );
 68     fscanf( fp2 , "%d" , &pnum );
 69     printf( "%s\n" , filename );
 70     fscanf( fp , "%f %f %f" , &xt , &yt , &zt );
 71     printf( "%s\n" , filename );
 72     ov[0].x = (double)xt;
 73     ov[0].y = (double)yt;
 74     ov[0].z = (double)zt;
 75     fscanf( fp2 , "%s %f %f" , dum , &xt , &yt );
 76     strcpy( pname[0] , dum );
 77     fv[0].x = (double)xt;
 78     fv[0].y = (double)yt;
 79     printf( " kokomade OK\n" );
 80     for( i = 1 ; i < pnum ; i++ ){
 81 /*      fscanf( fp , "%f %f %f" , &xt , &yt , &zt );
 82       ov[i].x = (double)xt - ov[0].x;
 83       ov[i].y = (double)yt - ov[0].y;
 84       ov[i].z = ov[0].z - (double)zt;
 85 */      fscanf( fp2 , "%s %f %f" , dum , &xt , &yt );
 86       strcpy( pname[i] , dum );
 87       fv[i].x = (double)xt;
 88       fv[i].y = (double)yt;
 89       printf( "fv[%d] = (%f,%f)\n" , i , fv[i].x , fv[i].y );
 90       printf( "np = %d\n" , np );
 91     ]
 92     fclose( fp2 );
 93     fclose( fp );
 94   ]
 95
 96   prefsize( 512 , 25 );
 97   printf( "\nSelect position of \"Coordinate & Text Window\"\n" );
 98   winopen( "Coordinate & Text Window" );
 99   textwindowid = winget();
100   mapcolor( GRAY40 , 102 , 102 , 102 );
101
102   prefsize( 400 , 400 );
103   printf( "Select position of \"Zoom Window\"\n" );
104   winopen( "Zoom Window" );
105   zoomwindowid = winget();
106   RGBmode();
107   gconfig();
108
109   getorigin(&orgz[X], &orgz[Y]);
110   getsize(&size[X], &size[Y]);
111
112   readsize[X] = size[X] / ZOOM / 2;
113   readsize[Y] = size[Y] / ZOOM / 2;
114   rectzoom((float)ZOOM, (float)ZOOM);
115
116   prefsize( 512 , 512 );
117   printf( "Select position of \"Main Window\"\n" );
118   sprintf( dum , "Main Window (%s)" , fname );
119   winopen( dum );
120   mainwindowid = winget();
121
122   scanf( "%d" , &i );
123   RGBmode();
124   gconfig();
125   clear();
126
127   getorigin(&org[X], &org[Y]);
128
129   lrectwrite( 0 , 0 , 511 , 511 , (unsigned long *)bufptr );
130
```

```
131   qdevice( KEYBD );
132
133   mdev[X] = MOUSEX;
134   mdev[Y] = MOUSEY;
135 /*
136   gflush();
137 */
138   if( modec[0] == 'f' ){
139     printf( "\nPlease Select the Point A (front view)\n" );
140     select_button( &fv[0] , mdev , org );
141     printf( "Point A Coordinate = ( %f , %f )\n" , fv[0].x , fv[0].y );
142
143     printf( "\nPlease Select the Point B (front view)\n" );
144     select_button( &fv[1] , mdev , org );
145     printf( "Point B Coordinate = ( %f , %f )\n" , fv[1].x , fv[1].y );
146
147     printf( "\nPlease Select the Point C (front view)\n" );
148     select_button( &fv[2] , mdev , org );
149     printf( "Point C Coordinate = ( %f , %f )\n" , fv[2].x , fv[2].y );
150
151     while( eflag ){
152       printf( "\nPlease Select the Point X[%d] (front view )\n" , pnum );
153       select_button( &fv[pnum] , mdev , org );
154       printf( "Point X[%d] Coordinate = ( %f , %f )\n"
155             , pnum ,fv[pnum].x,fv[pnum].y );
156
157       GLscanf_char( "\nWhat's this points name ( q -> quit )" , pname[pnum] );
158
159       printf( "\nThis point is %s\n" , pname[pnum] );
160       if( pname[pnum][0] == 'q' ) eflag = 0;
161       else                        pnum++;
162     ]
163     printf( "fv[0] = (%f , %f)\n" , fv[0].x , fv[0].y );
164
165     sprintf( filename , "/home/tatsu/atr/data/fgrid/front%s.grid" , filenum );
166
167     if( NULL == ( fp = fopen( filename , "w" ))){
168       printf( "Cannot ope file .. %s\n" , filename );
169       exit(1);
170     }
171     printf( "\nOpen succeed .. front%s.grid , now writting.\n" , filenum );
172     fprintf( fp , "%d\n" , pnum );
173     for( i = 0 ; i < pnum ; i++ )
174       fprintf( fp , "%30s %12.8f %12.8f\n" , pname[i] , fv[i].x , fv[i].y );
175     fclose( fp );
176
177   ]else{
178     printf( "\nPlease Select the Point A (side view)\n" );
179     select_button( &sv[0] , mdev , org );
180     printf( "Point A Coordinate = ( %f , %f )\n" , sv[0].x , sv[0].y );
181
182     for( i = 1 ; i < pnum ; i++ ){
183       printf( "\nPlease Select the Point of %s(%d)\n" , pname[i] , i );
184       select_button( &sv[i] , mdev , org );
185       printf( "Point X[%d] Coordinate = ( %f , %f )\n" , sv[i].x , sv[i].y );
186       chdimensionP( fv[0] , sv[0] , 0.047126792 , 0.047126792
187                   , 230.0 , 260.0 , &av );
188       chdimensionP( fv[i] , sv[i] , 0.047126792 , 0.047126792
189                   , 230.0 , 260.0 , &xv );
190       printf( "av = (%f,%f,%f)\nxv = (%f,%f,%f)\n" ,av.x,av.y,av.z,xv.x,xv.y,xv.
    z);
191       xv.x -= av.x; xv.y -= av.y; xv.z = av.z - xv.z ;
192
193       printf( "Calculated Vector Value = ( %f , %f , %f )\n" ,
194             xv.x , xv.y , xv.z );
195       printf( "Optimal Vector Value = ( %f , %f , %f )\n" ,
```



```
196             ov[i].x , ov[i].y , ov[i].z);
197     if( 0 == GLscanf_ans( "OK? [ y)es or n)o ]" ) ) i--;
198     ]
199     sprintf( filename , "/home/tatsu/atr/data/sgrid/side%s.grid" , filenum );
200       if( NULL == ( fp = fopen( filename , "w" )))[
201       printf( "Cannot open file .. %s\n" , filename );
202       exit(1);
203     ]
204     printf( "\nOpen succeed .. side%s.grid , now writting.\n" , filenum );
205
206     fprintf( fp , "%d\n" , pnum );
207     for( i = 0 ; i < pnum ; i++ )
208       fprintf( fp , "%30s %12.8f %12.8f\n" , pname[i] , sv[i].x , sv[i].y );
209     fclose( fp );
210
211   ]
212   printf( "Done\n" );
213 ]
214
215 datasave( fname , pname , v , num )
216       char              *fname;
217       char              pname[100][50];
218       vector2           *v[100];
219       int               num;
220 [                                 .
221   FILE          *fp;
222   int           i;
223   vector2       tmpv;
224
225   printf( " v[0] = (%f , %f)\n" , v[0]->x , v[0]->y );
226
227   if( NULL == ( fp = fopen( fname , "w" ))){
228     printf( "Cannot open file .. %s (at datasave)\n" );
229     exit(1);
230   ]
231   printf( "\nOpen succeed .. %s , now writting.\n" , fname );
232
233   fprintf( fp , "%d\n" , num );
234
235   for( i = 0 ; i < num ; i++ )
236     fprintf( fp , "%30s %12.8f %12.8f\n" , pname[i] ,  v[0]->x , v[0]->y );
237
238   fclose( fp );
239   printf ( "Done.\n" );
240 ]
241
242 select_button( v , mdev , org )
243       vector2   *v;
244       Device mdev[XY];
245       long org[XY];
246 {
247   int           pushflag = 0;
248   short         mval[XY];
249   short         val;
250   char          message[100];
251   long          vert[XY];
252   vector2       idou;
253   vector2       tmpv;
254
255   qdevice( LEFTMOUSE );
256   qdevice( MOUSEX );
257   qdevice( MOUSEY );
258
259   while( 1 ) [
260     switch( qread( &val )) {
261     case LEFTMOUSE:
```

```
262       if( pushflag != 1 ) pushflag = 1;
263       else{
264         pushflag = 0;
265         getdev( XY , mdev , mval );
266         tmpv.x = mval[X] - org[X];
267         tmpv.y = mval[Y] - org[Y];
268         printf( "tmpv.x = %f .y = %f\n" , tmpv.x , tmpv.y );
269         while( 1 ) {
270           switch( qread(&val )) {
271           case LEFTMOUSE:
272             if( pushflag != 1 ) pushflag = 1;
273             else{
274               pushflag = 0;
275               getdev( XY , mdev , mval );
276               v->x = tmpv.x +  (double)( mval[X] - org[X] - tmpv.x ) / 10.0;
277               v->y = tmpv.y +  (double)( mval[Y] - org[Y] - tmpv.y ) / 10.0;
278               unqdevice( LEFTMOUSE );
279               unqdevice( MOUSEX );
280               unqdevice( MOUSEY );
281               qreset;
282               return;
283             ]
284             break;
285           case MOUSEX:
286           case MOUSEY:
287             getdev( XY , mdev , mval );
288             idou.x = mval[X] - org[X] - tmpv.x;
289             idou.y = mval[Y] - org[Y] - tmpv.y;
290             winset( textwindowid );
291             color( GRAY40 );
292             clear();
293             color( BLACK );
294             cmov2s( 5 , 5 );
295             sprintf( message , "X = %5.2f , Y = %5.2f" ,
296                     tmpv.x + idou.x  / 10.0 , tmpv.y + idou.y / 10.0 );
297             charstr( message );
298             winset( zoomwindowid );
299             rectcopy(tmpv.x + org[X] - orgz[X] - readsize[X], tmpv.y + org[Y] -
    orgz[Y] - readsize[X],
300                 tmpv.x + org[X] + readsize[X] - orgz[X] ,
301                 tmpv.y + org[Y] + readsize[Y] - orgz[Y] , 0, 0);
302             color( GREEN );
303             bgnline();
304             vert[X] = readsize[X] * ZOOM + idou.x ;
305             vert[Y] = readsize[Y] * ZOOM + idou.y - 3;
306             v2i( vert );
307             vert[Y] = readsize[Y] * ZOOM + idou.y + 3;
308             v2i( vert );
309             endline();
310             bgnline();
311             vert[X] = readsize[X] * ZOOM + idou.x - 3;
312             vert[Y] = readsize[Y] * ZOOM + idou.y;
313             v2i( vert );
314             vert[X] = readsize[X] * ZOOM + idou.x + 3;
315             v2i( vert );
316             endline();
317             winset( mainwindowid );
318             break;
319           default:
320             break;
321           ]
322         }
323       ]
324       break;
325     case MOUSEX:
326     case MOUSEY:
```



```
327       getdev( XY , mdev , mval );
328       winset( textwindowid );
329       color( GRAY40 );
330       clear();
331       color( BLACK );
332       cmov2s( 5 , 5 );
333       sprintf( message , "X = %3d , Y = %3d" , mval[X] - org[X] , mval[Y] - org[
    Y] );
334       charstr( message );
335       winset( zoomwindowid );
336       mval[X] -= orgz[X];
337       mval[Y] -= orgz[Y];
338       rectcopy(mval[X] - readsize[X], mval[Y] - readsize[Y],
339                     mval[X] + readsize[X], mval[Y] + readsize[Y], 0, 0);
340       color( GREEN );
341       pntsize( 2 );
342       bgnpoint();
343       vert[X] = size[X] / 2;
344       vert[Y] = size[Y] / 2;
345       v2i( vert );
346       endpoint();
347       winset( mainwindowid );
348       break;
349     defaults:
350       break;
351     }
352   ]
353 ]
354 int GLscanf_ans( st )
355      char       st[];
356 {
357   short          val;
358
359   winset( textwindowid );
360   color( GRAY40 );
361   clear();
362   color( BLACK );
363   cmov2s( 5 , 5 );
364   charstr( st );
365   winset( mainwindowid );
366
367   while( 1 ){
368     if( qread( &val ) == KEYBD ){
369       if( val == (short)('y') )
370         return(1);
371       if( val == (short)('n') )
372         return(0);
373     }
374   ]
375 }
376
377 GLscanf_char( st , value )
378      char       st[] , value[];
379 {
380   short          val;
381   char           tmp[1];
382
383   printf( "%s .. " , st );
384
385   strcpy( value , "" );
386   qdevice( KEYBD );
387
388   winset( textwindowid );
389   color( GRAY40 );
390   clear();
391   color( BLACK );
```

```
392   cmov2s( 5 , 5 );
393   charstr( " Point Name .. " );
394   winset( mainwindowid );
395
396   while( 1 ){
397     switch( qread( &val ) ){
398     case KEYBD:
399       switch( val ){
400         case 13:
401           unqdevice( KEYBD );
402           qreset();
403           return(0);
404           break;
405         case 8:
406         case 127:
407           if( strlen( value ) != 0 ){
408             strcpy( value+strlen(value)-1 , "\0" );
409             winset( textwindowid );
410             color( GRAY40 );
411             clear();
412             color( BLACK );
413             cmov2s( 5 , 5 );
414             charstr( "Point Name .. " );
415             charstr( value );
416             winset( mainwindowid );
417           }
418           break;
419         case 32:
420           val = 95;
421         default:
422           sprintf( tmp , "%c" , val );
423           strcat( value , tmp );
424           winset( textwindowid );
425           color( GRAY40 );
426           clear();
427           color( BLACK );
428           cmov2s( 5 , 5 );
429           charstr( "Point Name .. " );
430           charstr( value );
431           winset( mainwindowid );
432       }
433       break;
434     default:
435       break;
436     }
437   }
438 }
439
440
441
```

```
#include        "std.h"
#include        "wavefront.h"

#define         THRESH          25.0


/*****************************************************
  WaveFrontのデータを制御点をもとにグループ分けする
*****************************************************/

DivGroup( wf , ip , ig )
WaveFront               *wf;
InflPoint               *ip;
InflGroup               *ig;
{
  int           i , j , jj , maxnum = 0;
  double        dis , max ;
  int           count;
  double        totalweight;

  for( j = 0 ; j < wf->tgridnum ; j++ ){
    if( (ig+j)->totcp == 0 ){
      jj = TextToReal( wf , j );
      count = 0;
      max = -2.0;
      totalweight = 0.0;
      for( i = 0 ; i < ip->pnum ; i ++ ){
        dis = sqrdb( ip->mgrid[i].x - wf->mgrid[jj].x ) +
          sqrdb( ip->mgrid[i].y - wf->mgrid[jj].y ) +
            sqrdb( ip->mgrid[i].z - wf->mgrid[jj].z );
        dis = sqrt( dis );
        if( j == 943 ){
          printf( "Ip(%f %f %f),Wf(%f %f %f)\n" , ip->mgrid[i].x , ip->mgrid[i].
y,
                ip->mgrid[i].z , wf->mgrid[jj].x , wf->mgrid[jj].y , wf->mgrid[
jj].z);
          printf( "dis = %f\n" , dis );
          printf( "jj =%d\n" , jj );
        }
        if ( dis < THRESH ){
          (ig+j)->cpweight[count] = ( THRESH - dis ) / THRESH;
          totalweight += (ig+j)->cpweight[count];
          (ig+j)->cpnum[count] = i;
          if( max < (ig+j)->cpweight[count] ) {
            max = (ig+j)->cpweight[count];
            maxnum = (ig+j)->cpnum[count] ;
          }
          count++;
        }
      }
      (ig+j)->totcp = count;
      (ig+j)->near = maxnum;
/*
      if( totalweight > 1.0 ){
        totalweight = 1.0 / totalweight;
        for( i = 0 ; i < count ; i++ )
          (ig+j)->cpweight[i] *= totalweight;
      }
*/
    }
  }
}

WriteGroup( filename , gnum , ig )
char            *filename;
int             gnum;
```



```
InflGroup       *ig;
{
  int           i , j , num , maxnum ;
  FILE          *fp;
  InflGroup     tmp;
  double        max;

  if( NULL == ( fp = fopen( filename , "w" ) ) ){
    printf( "Cannot open file ..%s(WriteGroup@divgroup)\n" , filename );
    exit( 1 );
  }

  fprintf( fp , "%d\n" , gnum );

  for( j = 0 ; j < gnum ; j++ ){
    tmp.totcp = (ig+j)->totcp;
    num = 0;
    for( i = 0 ; i < tmp.totcp ; i++ ){
      if( (ig+j)->cpweight[i] != 0.0 ){
        tmp.cpnum[num] = (ig+j)->cpnum[i];
        tmp.cpweight[num] = (ig+j)->cpweight[i];
        num++;
      }
    }
    tmp.totcp = num;
    max = 0.0;

    for( i = 0 ; i < num ; i++ ){
      if( max < tmp.cpweight[i] ){
        max = tmp.cpweight[i];
        maxnum = i;
      }
    }
    tmp.near = tmp.cpnum[maxnum];

    fprintf( fp , "%d %d\n" , tmp.totcp , tmp.near );
    for( i = 0 ; i < tmp.totcp ; i++ )
      fprintf( fp , "%d %f\n" , tmp.cpnum[i] , tmp.cpweight[i] );
  }

  fclose( fp );
}

ReadGroup( filename , ig )
char            *filename;
InflGroup       *ig;
{
  int           i , j , num ;
  FILE          *fp;
  int           tot , near , t;
  float         d;

  if( NULL == ( fp = fopen( filename , "r" ) ) ){
    printf( "Cannot open file ..%s(ReadGroup@divgroup)\n" , filename );
    exit( 1 );
  }

  fscanf( fp , "%d" , &num );

  for( j = 0 ; j < num ; j++ ){
    fscanf( fp , "%d %d" , &tot , &near );
    (ig+j)->totcp = tot ;
    (ig+j)->near = near;
    for( i = 0 ; i < tot ; i++ ){
      fscanf( fp , "%d %f" , &t , &d );
      (ig+j)->cpnum[i] = t;
```

```
131         (ig+j)->cpweight[i] = (double)d;
132       }
133     }
134
135     fclose( fp );
136   }
137
138   ReadControl( filename , ip )
139   char              *filename;
140   InflPoint         *ip;
141   {
142     FILE            *fp;
143     int             i , j , num;
144     float           x , y , z ;
145     char            chartmp[50];
146     int             side , mov;
147
148     if( NULL == ( fp = fopen( filename , "r" ) ) ){
149       printf( "Cannot open file ..%s(ReadControl@divgroup)\n" , filename );
150       exit( 1 );
151     }
152
153     fscanf( fp , "%d" , &num );
154     ip->pnum = num;
155
156     for( i = 0 ; i < num ; i++ ){
157
158       fscanf( fp , "%s" , chartmp );
159       strcpy( ip->cpname[i] , chartmp );
160
161       fscanf( fp , "%f %f" , &x , &y );
162       ip->tgrid[i].x = (double)x;
163       ip->tgrid[i].y = (double)y;
164
165       fscanf( fp , "%f %f %f" ,&x , &y , &z );
166       ip->mgrid[i].x = (double)x;
167       ip->mgrid[i].y = (double)y;
168       ip->mgrid[i].z = (double)z;
169
170       fscanf( fp , "%d %d" , &side , &mov );
171       ip->side[i] = side;
172       ip->movnum[i] = mov;
173
174     }
175
176     fclose( fp );
177   }
178
179
```

```
/*
  WaveFront Formatのデータとカラーイメージを読み込んで
  画面にワイヤフレームとテクスチャを表示するプログラム
  拡大、縮小、や点の座標の同定、格子点座標の算出なんか
  できるかもしれない

  usage  dispweight [WaveFrontFile] [ControllPointFile] [WeightFile]

*/

#include        <gl/gl.h>
#include        <gl/device.h>

#include        "std.h"
#include        "cimage.h"
#include        "wavefront.h"

#define         DISP_MODE               0
#define         CP_MODE                 1
#define         EDIT_MODE               2
#define         FILE_MODE               3

double          CPvalueget();

main( argc , argv )
int             argc;
char            *argv[];
{
  char          fname[100] , fnametmp[100] , answer[10];
  int           i , j;
  ImageBuffer   *bufptr;
  WaveFront     wf;
  InflGroup     ig[MAXGRIDNUM];
  InflPoint     ip;

  int           sizex = 512 , sizey = 512 ;
  long          x , y , orgx , orgy ;
  float         wsizex , wsizey , tmpy , tmpx;
  float         zoomx  , zoomy , posizex , posizey;
  Device        mdev[2] , dev ;
  short         mval[2] , val , lasty , lastx;
  int           centx = 255 , centy = 255 , pushflag = 0 , pushflag2 = 0;
  int           pushflag3 = 0 , pmode = 0 , tpo = -1 , rtpo;
  vector2       siten;
  double        piw , infp;

  siten.x = 0 ; siten.y = 0;

  bufptr = (ImageBuffer *)malloc( 512 * 512 * 4 );

  ReadWaveFront( argv[1] , &wf , 10 );
  printf( ".obj read OK.\n" );

  ReadGroup( argv[3] , ig );
  printf( "Weight file read OK.\n" );

  ReadControl( argv[2] , &ip );
  printf( "Control point file read OK.\n\n" );

  foreground();

  winopen( argv[1] );

  gconfig();
  clear();
```



```
  getsize( &x , &y );
  wsizex = (float)x ; wsizey = (float)y;
  getorigin( &orgx , &orgy );

  Color( WHITE );

  WFwrite( &wf , siten , wsizex , wsizey , x , y , DOT_ON_MID );
  AllCPwrite( &ip , siten , wsizex , wsizey , x , y , DOT_ON_HUGE );

  qdevice( MIDDLEMOUSE );
  qdevice( MOUSEY );
  qdevice( MOUSEX );
  qdevice( LEFTMOUSE );
  qdevice( RIGHTMOUSE );

  mdev[0] = MOUSEX;
  mdev[1] = MOUSEY;

  getdev( 2 , mdev , mval );
  lastx = mval[0];
  lasty = mval[1];

  printf( "MODE ---> DISPLAY MODE\n" );

  while( 1 ){
    switch( dev = qread( &val ) ){
    case LEFTMOUSE:
      switch( pmode ){
      case DISP_MODE:
        if( pushflag2 == 0 ){
          pushflag2 = 1;
          getdev( 2 , mdev , mval );
          lastx = mval[0];
          lasty = mval[1];
        }
        else{
          pushflag2 = 0;
          WFwrite( &wf , siten , wsizex , wsizey , x , y , DOT_ON_MID );
          AllCPwrite( &ip , siten , wsizex , wsizey , x , y , DOT_ON_HUGE );
        }
        break;
      case EDIT_MODE:
        getdev( 2 , mdev , mval );
        mval[0] -= orgx;
        mval[1] -= orgy;
        tpo = Pointget( &wf , siten , wsizex , wsizey , x , y , mval );
        rtpo = TextToReal( &wf , tpo );
/*      printf( "tpo( at texture) = %d rtpo(at real) = %d\n" , tpo , rtpo);
        printf( "tgrid(%f %f) mgrid(%f %f %f)\n" , wf.tgrid[tpo].x , wf.tgrid[tpo].y ,
                wf.mgrid[rtpo].x , wf.mgrid[rtpo].y , wf.mgrid[rtpo].z );
*/      WFwriteOP( &wf , siten , wsizex , wsizey , x , y , DOT_ON_MID ,
                   YELLOW , CLEAR_OFF );

        CPwrite( &wf , &ip , ig , tpo , siten , wsizex , wsizey , x , y );
        Pwrite( &wf , tpo , siten , wsizex , wsizey , x , y , DOT_ON_MID );
        break;
      case FILE_MODE:
        printf( "\nGroup file output command\n" );
        input_text( "Are you sure ( y / n )" , answer );
        if( strcmp( answer , "y" ) == 0 ){
          input_text( "\nOutput file name" , fname );
          NormalizeIG( ig , wf.tgridnum );
          WriteGroup( fname , wf.tgridnum , ig );
          printf( "Done..\n" );
        }else     printf( "\ncommand canceled .. \n" );
```

```
132         qreset();
133         break;
134       case CP_MODE:
135         getdev( 2 , mdev , mval );
136         mval[0] -= orgx;
137         mval[1] -= orgy;
138         tpo = CPointget( &ip , siten , wsizex , wsizey , mval );
139         CPInfAreaWrite( &wf,&ip , ig , tpo , siten , wsizex , wsizey , x, y );
140         qreset();
141         break;
142       }
143       break;
144
145     case MIDDLEMOUSE:
146       switch( pmode ){
147       case DISP_MODE:
148         if( pushflag == 0 ){
149           pushflag = 1;
150           getdev( 2 , mdev , mval );
151           lasty = mval[1];
152           lastx = mval[0];
153         }
154         else{
155           pushflag = 0;
156           WFwrite( &wf , siten , wsizex , wsizey , x , y , DOT_ON_MID );
157           AllCPwrite( &ip , siten , wsizex , wsizey , x , y , DOT_ON_HUGE );
158         }
159         break;
160       case EDIT_MODE:
161         if( tpo == -1 )   printf(
162          "this command will excute after selecting point on left mouse\n" );
163         else{
164           getdev( 2 , mdev , mval );
165           mval[0] -= orgx;
166           mval[1] -= orgy;
167           rtpo = CPointget( &ip , siten ,wsizex , wsizey , mval );
168           printf( "rtpo = %d\n" , rtpo );
169           infp = CPvalueget( ig , tpo , rtpo );
170 /*        if( infp > 0.0 ){          */
171             printf( "Information .. \n" );
172             printf( "This control point has influence %f percents\n\n" , infp *
    100.0);
173
174             input_text( "Please input weight or n)o change" , fnametmp  );
175             if( strcmp( fnametmp , "n" ) != 0 ){
176               piw = (double)atof( fnametmp ) / 100.0;
177               Chweight( ig ,tpo , rtpo , piw , infp );
178
179               WFwriteOP( &wf , siten , wsizex , wsizey , x , y , DOT_ON_MID ,
180                   YELLOW , CLEAR_OFF );
181               CPwrite( &wf , &ip , ig , tpo , siten , wsizex , wsizey , x , y );
182               Pwrite( &wf , tpo , siten , wsizex , wsizey , x , y , DOT_ON_MID )
    ;
183             }else  printf( "\ncommand canceled ..\n" );
184 /*        }else printf( "Selected control point gives no influence ..\n" );
185 */      qreset();
186         }
187         break;
188       case FILE_MODE:
189         printf( "\nCaution!\n" );
190         printf( "This program don't save weight file automatically.\n" );
191         input_text( "\nDo you realy want to quit? (y / n)" , answer );
192         if( strcmp( answer , "y" ) == 0 ) exit( 1 );
193         printf( "\ncommand canceled .. \n" );
194         qreset();
195         break;
```



```
196       }
197       break;
198
199     case RIGHTMOUSE:
200       if( pushflag3 == 0 ) pushflag3 = 1;
201       else{
202         pushflag3 = 0;
203         switch( pmode ){
204         case DISP_MODE:
205           pmode = CP_MODE;
206           printf( "MODE ---> CONTROL POINT MODE\n" );
207           break;
208         case CP_MODE:
209           pmode = EDIT_MODE;
210           printf( "MODE ---> EDIT_MODE\n" );
211           break;
212         case EDIT_MODE:
213           pmode = FILE_MODE;
214           tpo = -1;
215           printf( "MODE ---> FILE MODE\n" );
216           break;
217         case FILE_MODE:
218           pmode = DISP_MODE;
219           printf( "MODE ---> DISPLAY MODE\n" );
220           break;
221         }
222       }
223       break;
224     case MOUSEY:
225       if( pushflag == 1 ){
226         tmpy = (float)( val - lasty ) * 10.0;
227         Color( WHITE );
228         wsizex += tmpy ; wsizey += tmpy;
229         WFwriteOP( &wf , siten , wsizex , wsizey , x , y , DOT_ON_MID , YELLOW ,
230                    CLEAR_ON );
231         AllCPwrite( &ip , siten ,  wsizex , wsizey , x , y , DOT_ON_HUGE );
232         lasty = val;
233       }
234       break;
235     case MOUSEX:
236       if( pushflag2 == 1 ){
237         getdev( 2 , mdev , mval );
238         tmpx = (float)( lastx - mval[0]  );
239         tmpy = (float)( lasty - mval[1]  );
240         siten.x += tmpx ; siten.y += tmpy;
241         WFwriteOP( &wf , siten , wsizex , wsizey , x , y , DOT_ON_MID , YELLOW ,
242                    CLEAR_ON );
243         AllCPwrite( &ip , siten ,  wsizex , wsizey , x , y , DOT_ON_HUGE );
244         lastx = mval[0]; lasty = mval[1];
245       }
246       break;
247     default:
248       break;
249     }
250   }
251 }
```

```
/*
  deform.c のサブルーチン集です。
*/

#include                "std.h"
#include                "wavefront.h"

ReadMovement( fname , movect )
char            *fname;
MoVector        *movect;
{
  FILE          *fp;
  float         x , y , z;
  char          name[30];
  int           i , j , jj ;

  if( NULL == ( fp = fopen( fname , "r" ) ) ){
    printf( "Cannot open file .. %s(ReadMovement@deformwf)\n" , fname );
    exit( 1 );
  ]

  fscanf( fp , "%d %d" , &i , &j );
  movect->fnum = i;
  movect->pnum = j;

  for( i = 0 ; i < movect->pnum ; i++ ){
    fscanf( fp , "%s" , name );
    strcpy( movect->pname[i] , name );
    for( j = 0 ; j < movect->fnum ; j++ ){
      fscanf( fp , "%d %f %f %f" , &jj , &x , &y , &z );
      movect->mov[i][j].x = (double)x;
      movect->mov[i][j].y = (double)y;
      movect->mov[i][j].z = (double)z;
/*
      printf( "movector = (%f %f %f)\n" , x , y , z );
*/
    }
  }
  fclose( fp );
}

DeformWF( wf , movect , ig , ip , frame , dwf , clsf )
WaveFront               *wf , *dwf;
MoVector                *movect;
InflGroup               *ig;
int                     frame , clsf;
InflPoint               *ip;
{
  int           i , j , mgnum;
  double        x , y , z ;

  for( i = 0 ; i < wf->tgridnum ; i++ ){
    x = 0.0 ; y = 0.0 ; z = 0.0;
    for( j = 0 ; j < (ig+i)->totcp ; j++ ){
      x += movect->mov[ip->movnum[(ig+i)->cpnum[j]]][frame].x *
        (ig+i)->cpweight[j] * ip->side[(ig+i)->cpnum[j]];
      y += movect->mov[ip->movnum[(ig+i)->cpnum[j]]][frame].y *
        (ig+i)->cpweight[j];
      z += movect->mov[ip->movnum[(ig+i)->cpnum[j]]][frame].z *
        (ig+i)->cpweight[j];
    ]
/*
    printf( "x,y,z = (%f %f %f)\n" , x , y , z );
*/
    mgnum = TextToReal( wf , i );
    dwf->mgrid[mgnum].x = wf->mgrid[mgnum].x + x;
```



```
    dwf->mgrid[mgnum].y = wf->mgrid[mgnum].y + y;
    dwf->mgrid[mgnum].z = wf->mgrid[mgnum].z + z;
    dwf->tgrid[i].x = wf->tgrid[i].x;
    dwf->tgrid[i].y = wf->tgrid[i].y;
    dwf->tgrid[i].z = wf->tgrid[i].z;
  ]

  if( clsf == CLEAR_ON ){
    dwf->gridnum = wf->gridnum;
    dwf->tgridnum = wf->tgridnum;
    dwf->linknum = wf->linknum;
    strcpy( dwf->colfname , wf->colfname );

    for( i = 0 ; i < dwf->linknum ; i++ ){
      dwf->link[i].model.g1 = wf->link[i].model.g1;
      dwf->link[i].model.g2 = wf->link[i].model.g2;
      dwf->link[i].model.g3 = wf->link[i].model.g3;
      dwf->link[i].tex.g1 = wf->link[i].tex.g1;
      dwf->link[i].tex.g2 = wf->link[i].tex.g2;
      dwf->link[i].tex.g3 = wf->link[i].tex.g3;
    ]
  ]
]

WriteWaveFront( fname , wf , val )
char            *fname;
WaveFront       *wf;
double          val;
{
  FILE          *fp;
  int           i , j;

  if( NULL == ( fp = fopen( fname , "w" ) ) ){
    printf( "Cannot open file .. %s(WriteWaveFront@deformwf)\n" );
    exit( 1 );
  ]

  fprintf( fp , "#   Wavefront data format\n" );

  for( i = 0 ; i < wf->gridnum ; i++ ){
    fprintf( fp , "v %10.6f %10.6f %10.6f\n" , wf->mgrid[i].x * val,
            wf->mgrid[i].y * val , wf->mgrid[i].z * val );
  ]

  for( i = 0 ; i < wf->tgridnum ; i++ ){
    fprintf( fp , "vt %10.6f %10.6f %10.6f\n" , wf->tgrid[i].x ,
            wf->tgrid[i].y , 0.0 );
  ]

  fprintf( fp , "usemtl white\n" );
  fprintf( fp , "usemap %s\n" , wf->colfname );

  for( i = 0 ; i < wf->linknum ; i++ ){
    fprintf( fp , "f %d/%d %d/%d %d/%d\n" , wf->link[i].model.g1 ,
            wf->link[i].tex.g1 , wf->link[i].model.g2 ,
            wf->link[i].tex.g2 , wf->link[i].model.g3 ,
            wf->link[i].tex.g3 );
  ]

  fclose( fp );
]
```

```
 1 /*
 2   計算された基準点の移動ベクトルと
 3   これまた計算された重みとで、モデルを変形させる
 4   プログラムです
 5 */
 6
 7 #include        "std.h"
 8 #include        "wavefront.h"
 9
10 main( argc , argv )
11 int             argc;
12 char            *argv[];
13 {
14   WaveFront             wf , dwf;
15   InflGroup             ig[MAXGRIDNUM];
16   InflPoint             ip;
17   MoVector              movect;
18   int                   frame;
19   char                  fname[100];
20
21   ReadWaveFront( argv[1] , &wf , 10 );
22   printf( "WaveFront file (.obj file) read OK.\n" );
23
24   ReadMovement( argv[2] , &movect );
25   printf( "Movement file read OK.\n" );
26
27   ReadGroup( argv[3] , ig );
28   printf( "InflGroup file read OK.\n" );
29
30   ReadControl( argv[5] , &ip );
31   printf( "InflPoint file read OK.\n" );
32
33   frame = 0;
34   printf( "Calc & Write now .. %d/%d frame\n" , frame + 1 , movect.fnum );
35   sprintf( fname , "%s%03d.obj" , argv[4] , frame + 1 );
36   DeformWF( &wf , &movect , ig , &ip , frame , &dwf , CLEAR_ON );
37   WriteWaveFront( fname , &dwf , 0.1 );
38
39   for( frame = 1 ; frame < movect.fnum ; frame++ ){
40     printf( "Calc & Write now .. %d/%d frame\n" , frame + 1 , movect.fnum );
41     sprintf( fname , "%s%03d.obj" , argv[4] , frame + 1 );
42     DeformWF( &wf , &movect , ig , &ip , frame , &dwf , CLEAR_OFF );
43     WriteWaveFront( fname , &dwf , 0.1 );
44   }
45   printf( "Done..\n" );
46
47 }
48
```

```
#include        "math.h"
#include        "cmath.h"

double  Gaiseki() , Setten();
int     DirectionCheck();

void CoordConvert( av , bv , cv , xv )
vector2         *av , *bv , *cv , *xv;
{
        bv->x -= av->x;
        bv->y -= av->y;

        cv->x -= av->x;
        cv->y -= av->y;

        xv->x -= av->x;
        xv->y -= av->y;
}

void CoordConvert3( av , bv , cv , xv )
vector3         *av , *bv , *cv , *xv;
{
        bv->x -= av->x;
        bv->y -= av->y;
        bv->z = av->z - bv->z;

        cv->x -= av->x;
        cv->y -= av->y;
        cv->z = av->z - cv->z;

        xv->x -= av->x;
        xv->y -= av->y;
        xv->z = av->z - xv->z;
}

void CoordConvertB( av , bv , cv )
vector3         *av , *bv , *cv ;
{
        bv->x -= av->x;
        bv->y -= av->y;
        bv->z = av->z - bv->z;

        cv->x -= av->x;
        cv->y -= av->y;
        cv->z = av->z - cv->z;
}

void CoordConvertX( av , xv )
vector3         *av , *xv;
{
        xv->x -= av->x;
        xv->y -= av->y;
        xv->z = av->z - xv->z;
}

void Normalize3Dvector( fbv , sbv , bv )
vector2         fbv , sbv ;
vector3         *bv;
{
        double          a;

        bv->x = fbv.x;
        bv->y = fbv.y;

        if( fbv.y == 0.0 || sbv.y == 0.0 ) a = 1.0;
        else                a = ( fbv.y / sbv.y );
```



```
        bv->z = a * sbv.x;

]

double  Gaiseki( bv , cv , ov )
vector3         bv , cv , *ov ;
{
        double  value;
        double  valueb;

        ov->x = bv.y * cv.z - cv.y * bv.z ;
        ov->y = bv.z * cv.x - bv.x * cv.z;
        ov->z = bv.x * cv.y - cv.x * bv.y;

        value = sqrt( ov->x * ov->x + ov->y * ov->y + ov->z * ov->z );

        valueb = sqrt( (bv.x - cv.x)* (bv.x -cv.x)+ (bv.y-cv.y) * (bv.y-cv.y) +( bv.z-cv.z) * (bv.z-cv.z) );

        ov->x = ( ov->x / value ) * valueb;
        ov->y = ( ov->y / value ) * valueb;
        ov->z = ( ov->z / value ) * valueb;

        return( valueb );
]


void Plane( ov , value , a , b , c )
vector3         ov;
double          value , *a , *b , *c;
{
        *a = ov.x / value;
        *b = ov.y / value;
        *c = ov.z / value;
]

double Setten( normals , xv , a , b , c , hv )
vector3         xv , *hv;
double          a , b , c , normals;
{
        double          value ;
        double          x , y , z;

/* heimen to suisen no kouten wo motomeru */
        hv->y = (( a * a + c * c ) * xv.y - b * ( a * xv.x + c * xv.z )) /
                        ( a * a + b * b + c * c );

        if( b == 0.0 ) b = 0.000000000001;

        hv->x = ( a * hv->y - a * xv.y + b * xv.x ) / b;
        hv->z = ( c * hv->y - c * xv.y + b * xv.z ) / b;

/* housenhoukou seibun wo motome seikika suru */
        x = xv.x - hv->x  ;
        y = xv.y - hv->y  ;
        z = xv.z - hv->z  ;
        value = sqrt( x * x + y * y + z * z ) / normals;
        if( DirectionCheck( x , y , z , a , b , c  ) == -1 ) value *= -1;

        printf( "nnv = (%f,%f,%f)\n" ,x , y , z );
        return( value );
]

int DirectionCheck( x , y , z , a , b , c )
     double x , y , z , a , b , c;
{
```

```
132   int   val1 , val2 , val3;
133
134   val1 = SignDouble( x ) * SignDouble( a );
135   val2 = SignDouble( y ) * SignDouble( b );
136   val3 = SignDouble( z ) * SignDouble( c );
137
138   if( val1 < 0 || val2 < 0 || val3 < 0 ) return( -1 );
139   if( val1 == val2 == val3 ) return( 0 );
140   return( 1 );
141 ]
142
143 void Normalize( bv , cv , hv , nb , nc )
144 vector3         bv , cv , hv ;
145 double          *nb , *nc ;
146 [
147         vector3         nbv , ncv;
148
149         ncv.y = ( hv.x * bv.y - bv.x * hv.y ) / ( ( cv.x * bv.y / cv.y ) -
150                 bv.x );
151
152         if( bv.x == 0.0 ) ncv.x = hv.x;
153         else              ncv.x = ( bv.z * hv.x - bv.x * hv.z ) /
154                                   ( bv.z - ( bv.x * cv.z / cv.x ) );
155
156         if( bv.z == 0.0 ) ncv.z = hv.z;
157         else               ncv.z = ncv.x * cv.z / cv.x;
158
159         nbv.x = hv.x - ncv.x;
160         nbv.y = hv.y - ncv.y;
161         nbv.z = hv.z - ncv.z;
162 /*
163         printf( "### Normalize ###\n" );
164         printf( "ncv = (%f,%f,%f) \nnbv = (%f,%f,%f)\n" ,
165                 ncv.x,ncv.y,ncv.z,nbv.x,nbv.y,nbv.z );
166     printf( "\nbvとnbvの成す角..%f\n" , ( bv.x * nbv.x + bv.y * nbv.y + bv.z *
167              nbv.z ) / sqrt( bv.x * bv.x + bv.y * bv.y + bv.z * bv.z ) /
168              sqrt( sqrdb( nbv.x ) + sqrdb( nbv.y ) + sqrdb( nbv.z ))));
169     printf( "### Normalize ###\n" );
170  */
171         *nb = sqrt( nbv.x * nbv.x + nbv.y * nbv.y + nbv.z * nbv.z ) /
172                 sqrt( bv.x * bv.x + bv.y * bv.y + bv.z * bv.z );
173         if( DirectionCheck( nbv.x , nbv.y , nbv.z , bv.x , bv.y , bv.z )
174            == -1 ) *nb *= -1;
175
176         *nc = sqrt( ncv.x * ncv.x + ncv.y * ncv.y + ncv.z * ncv.z ) /
177                 sqrt( cv.x * cv.x + cv.y * cv.y + cv.z * cv.z );
178         if( DirectionCheck( ncv.x , ncv.y , ncv.z , cv.x , cv.y , cv.z )
179            == -1 ) *nc *= -1;
180 ]
181
182 /*
183 void Normalize2DVector( av , bv , cv , xv , ncx , nbx )
184 vector2         av , bv , cv , xv;
185 double          *ncx , *nbx;
186 [
187         vector2         cxv , bxv ;
188         int             i , j ;
189         double          a , b , c ;
190
191         printf( "### Normalized 2D Vector ###\n" );
192
193  kaku keisuu wo motomeru
194         if( bv.x == 0.0 ) a = 1000000000;
195         else              a = bv.y / bv.x;
196         if( cv.x == 0.0 ) b = 1000000000;
197         else              b = cv.y / cv.x;
```



```
198
199         c = xv.y - a * xv.x;
200
201         printf( "a = %f , b = %f , c = %f \n" , a , b , c );
202
203  bekutoru no bunkai
204         cxv.x = ( - c ) / ( a - b );
205         cxv.y = cxv.y * b;
206
207         bxv.x = xv.x - cxv.x;
208         bxv.y = xv.y - cxv.y;
209
210         printf( "cxv = (%f,%f) , bxv = (%f,%f)\n",cxv.x,cxv.y,bxv.x,bxv.y);
211
212  bekutoru seikika
213         *ncx = sqrt( cxv.x * cxv.x + cxv.y * cxv.y ) /
214                                 sqrt( cv.x * cv.x + cv.y * cv.y );
215         *nbx = sqrt( bxv.x * bxv.x + bxv.y * bxv.y ) /
216                                 sqrt( bv.x * bv.x + bv.y * bv.y );
217         printf( "ncx = %f , nbx = %f\n" , *ncx , *nbx );
218
219         printf( "### END ###\n\n" );
220
221 ]
222
223 void Normalize3DVector( fcv , fbv , scv , sbv ,
224                                 fncx , fnbx , sncx , snbx , ncx , nbx )
225 vector2         fcv , fbv , scv , sbv ;
226 double          fncx , fnbx , sncx , snbx , *ncx , *nbx;
227 [
228         vector3         cv3 , bv3 , cxv3 , bxv3;
229         double          a , b , c;
230         int             i , j;
231
232  kizyun bekutoru wo 3jigen ka suru
233         cv3.x = fcv.x;
234         cv3.y = fcv.y;
235         if( fcv.y == 0.0 ) a = scv.y / 0.0000000001;
236         else               a = scv.y / fcv.y;
237         cv3.z = a * scv.x;
238
239         bv3.x = fbv.x;
240         bv3.y = fbv.y;
241         if( fbv.y == 0.0 ) b = sbv.y / 0.0000000001;
242         else               b = sbv.y / fbv.y;
243         bv3.z = b * fbv.x;
244
245  taishou bekutoru wo 3jigen ka suru
246         cxv3.x = fncx * cv3.x;
247         cxv3.y = fncx * cv3.y;
248         cxv3.z = sncx * cv3.z;
249
250         bxv3.x = fnbx * bv3.x;
251         bxv3.y = fnbx * bv3.y;
252         bxv3.z = snbx * bv3.z;
253
254  bekutoru seikika
255         *ncx = sqrt( cxv3.x * cxv3.x + cxv3.y * cxv3.y + cxv3.z * cxv3.z ) /
256                         sqrt( cv3.x * cv3.x + cv3.y * cv3.y + cv3.z * cv3.z );
257         *nbx = sqrt( bxv3.x * bxv3.x + bxv3.y * bxv3.y + bxv3.z * bxv3.z ) /
258                         sqrt( bv3.x * bv3.x + bv3.y * bv3.y + bv3.z * bv3.z );
259 ]
260 */
```

```
 1 #include    "stdio.h"
 2 #include    "cmath.h"
 3
 4 int SignDouble( val )
 5      double val;
 6 {
 7   if( val < 0 ) return( -1 );
 8   if( val > 0 ) return( 1 );
 9   return( 0 );
10 }
11
12 double sqrdb( d )
13      double     d;
14 {
15   return( d * d );
16 }
17
```

```
#include                "std.h"
#include                "cmath.h"

chdimension( fv , sv , ganma1 , ganma2 , cent1 , cent2 , v )
     vector2            fv , sv;
     vector3             *v;
     double             ganma1 , ganma2 , cent1 , cent2;
{
  double                fx , fy , fz;
  double                alpha1 , alpha2 , alpha3;

  fx = fv.x - cent1;
  fy = fv.y - 240;
  fz = cent2 - sv.x;

       printf( "fx = %f , fy = %f , fz = %f\n" , fx , fy , fz );

       alpha1 = fx * ganma1 / 320.0;
       alpha2 = fz * ganma2 / 320.0;
       alpha3 = fy * ganma1 / 320.0;

       printf( "alpha1 = %f , alpha2 = %f , alpha3 = %f\n" , alpha1 , alpha2 ,
                       alpha3 );

       v->x = ( tan( alpha1 ) * ( 320.0 * tan( alpha2 ) - fz + 320.0 ) + fx ) /
             ( 1.0 - tan( alpha1 ) * tan( alpha2 ));
       v->z = ( v->x - fx ) / tan( alpha1 );
       v->y = v->z * tan( alpha3 ) + fy;

       v->x *= 0.50201801;
       v->y *= -0.50201801;
       v->z *= -0.50201801;
/*
       v->x *= 0.3709503691;
       v->y *= -0.3709503691;
       v->z *= -0.3709503691;
*/
printf( "v = (%f,%f,%f)\n" , v->x , v->y , v->z );
}



chdimensionP( fv , sv , ganma1 , ganma2 , cent1 , cent2 , v )
     vector2    fv , sv;
     vector3             *v;
          double             ganma1 , ganma2 , cent1 , cent2;
{
    double                fx , fy , fz;
      double                alpha1 , alpha2 , alpha3;

    printf( "fv = (%f,%f)\n" , fv.x , fv.y );

      fx = fv.x - cent1;
      fy = fv.y - 240;
      fz = cent2 - sv.x;

      printf( "fx = %f , fy = %f , fz = %f\n" , fx , fy , fz );

      alpha1 = fx * ganma1 / 320.0;
      alpha2 = fz * ganma2 / 320.0;
      alpha3 = fy * ganma1 / 320.0;

      printf( "alpha1 = %f , alpha2 = %f , alpha3 = %f\n" , alpha1 , alpha2 ,
                      alpha3 );

      v->x = ( tan( alpha1 ) * ( 320.0 * tan( alpha2 ) - fz + 320.0 ) + fx ) /
```



```
            ( 1.0 - tan( alpha1 ) * tan( alpha2 ));
      v->z = ( v->x - fx ) / tan( alpha1 );
      v->y = v->z * tan( alpha3 ) + fy;

      v->x *= 0.50201801;
      v->y *= -0.50201801;
      v->z *= -0.50201801;
}
```

```
 1 #include        <gl/gl.h>
 2
 3 Color( col )
 4 unsigned short  col;
 5 {
 6   switch( col ){
 7   case WHITE:
 8     cpack( 0x00ffffff );
 9     break;
10   case CYAN:
11     cpack( 0x00ffff00 );
12     break;
13   case MAGENTA:
14     cpack( 0x00ff00ff );
15     break;
16   case BLUE:
17     cpack( 0x00ff0000 );
18     break;
19   case YELLOW:
20     cpack( 0x0000ffff );
21     break;
22   case GREEN:
23     cpack( 0x0000ff00 );
24     break;
25   case RED:
26     cpack( 0x000000ff );
27     break;
28   case BLACK:
29     cpack( 0x00000000 );
30     break;
31   default:
32     break;
33   }
34 }
```

```
1 #include        <gl/gl.h>
2
3 Color( col )
4 unsigned short  col;
5 {
6    color( col );
7 }
```

付録.5

MOの内容について

顔表情の分析実験　1
－基本6表情編－
○/mo/tatsu/simage/
側面画の001～226フレーム（各RGB汎用フォーマット）
○/mo/tatsu/fimage/
正面画の001～226フレーム（各RGB汎用フォーマット）

001～225フレーム
無表情＞怒り＞悲しみ＞喜び＞驚き＞嫌悪＞恐怖＞無表情　の表情変化。
１５フレーム／秒で１５秒分のデータ。
226フレーム
正規化用画像

顔表情の分析実験　2
－個別AU編－
○/mo/tatsu/
側面画の001～226フレーム（各RGB汎用フォーマット）
○/mo/tatsu/fimage/
正面画の001～226フレーム（各RGB汎用フォーマット）

001～045フレーム
眉を上げる（AU1,2）＞眉を寄せる（下げる。AU4）
046～090フレーム
顎を落とす（AU26）＞顎を突き出す（AU17）
091～135フレーム
目を細める（AU7）＞頬を上げる（AU6）
136～180フレーム
口をすぼめる（AU18）＞口を横に伸ばす（AU15）
181～225フレーム
口を小さく開く（AU25）＞上唇を開く（AU10）
226フレーム
正規化用画像

各１５フレーム／秒で３秒分のデータ
ただし、040フレームから060フレームまでの正面画のデータが破壊されている。

顔表情の分析実験　3
－個別AU・口形編－
○/mo/tatsu/
側面画の001～226フレーム（各RGB汎用フォーマット）
○/mo/tatsu/fimage/
正面画の001～226フレーム（各RGB汎用フォーマット）

001～045フレーム
目を閉じる（AU??）＞鼻に皺を寄せる（AU9）
１５フレーム／秒で３秒分のデータ
046～105フレーム
「こんにちわ」と発声中の口形及び表情
106～165フレーム
「あいうえお」と発声中の口形及び表情
166～225フレーム
「f」と「th」の発音時の口形及び表情
各１５フレーム／秒で４秒分のデータ
226フレーム
正規化用画像

顔表情の分析実験　4
－結果編－

実験１の結果のwavefront file