〔公　開〕

ＴＲ－Ｃ－０１３８

臨場感通信会議システムにおける実時間表情検出

海老原　一之　Kazuyuki EBIHARA
山田　正紀　Masaki YAMADA
大谷　淳　Jun OHYA
岸野　文郎　Fumio KISHINO

１９９５　３．１５

ＡＴＲ通信システム研究所

1.はじめに

異なる場所にいる人々があたかも一堂に会しているかのような感覚を持ちつつ、会議や協調作業を行える環境を提供する事を目的とした通信をＡＴＲ通信システム研究所では、臨場感通信会議とよび要素技術の検討を進めている。従来のテレビ会議システムでは、人間同士の自然なコミュニケーションのために重要な視線の一致や運動視差の再現が困難であったが、臨場感通信会議では３次元コンピュータグラフィック(3Dimension Computer Graphic)技術を用いて生成された仮想的な共有空間に３次元人物像や物体を配置し、立体ディスプレイに表示する手法を用いることでこれらの問題を解決している[1][2]。

ここで会議参加者については、送信側で表情や動きを検出し、受信側の３次元人物顔モデルにおいて再現する。したがって、参加者の表情検出を実時間で実現し、３次元顔モデル(3D Wire Frame Model:3D WFM)において忠実に再現できれば、会議参加者同士の円滑なコミュニケーションが可能となり、臨場感通信会議の実現に向けて大きな前進となる。また、会議参加者の表情を検出する場合、顔構成要素である目、口等の変形する部分だけではなく、参加者の視線を検出する事も重要となる。

今回、筆者は、会議参加者の顔にマーカを貼付してその軌跡を追跡することにより実現する実時間表情検出と、視線検出の研究を行った。

従来の表情検出プログラムは、高性能のグラフィックワークステーション(ＷＳ）上で動作し、また顔画像中のマーカ検出のために人間の皮膚の色と異なる小球のマーカを顔に添付していた。しかし、検出の精度、速度共に満足できるレベルではなかった。そこで、ビデオレートで処理可能な手法の検討を行った。

本稿では、表情検出のためのカメラを顔に対して相対的に同一の位置に保つために、小型ＣＣＤカメラを固定したヘルメットを会議参加者が被るようにし、得られた顔画像中から顔に前述のマーカの移動量を求め、表情筋の動きを検出している。(図１）カメラをヘルメットに固定したのは、顔画像が常に一定の画角で撮影されるようにし、処理系の負担を軽減するためである。これによってシステムは顔の向きを考慮した相対的なマーカ位置検出を行う必要がなくなった。ただ

し、ヘルメットがずれた場合などを想定し、鼻につけたマーカのずれを他のマーカにも反映している[1]。

マーカ追跡は WS から独立したパーソナルコンピュータ(ＰＣ)とマーカ抽出用のハードウェアにより実現されており、追跡結果は、無表情時の各マーカの位置との差分による２次元移動ベクトルとして得られる。

視線検出法は、前述のヘルメットに取り付けた視線検出用の CCD カメラから入力される目の映像から黒目の重心を算出する手法と、専用のカメラを必要とせず目の輝度情報の変化から白目と黒目の境界を見つけ出す手法の２種類を検討した。

本論文の第２章では、実時間表情検出・再現系の処理概要について、第３章では表情検出系のシステム構成について、第４章ではマーカ追跡部について、第５章では通信プロトコルについて、それぞれ述べる。第６章で、実験結果と今後の課題について述べる。

本稿で提案する手法により、ビデオレートでの実時間表情再現を実現した。

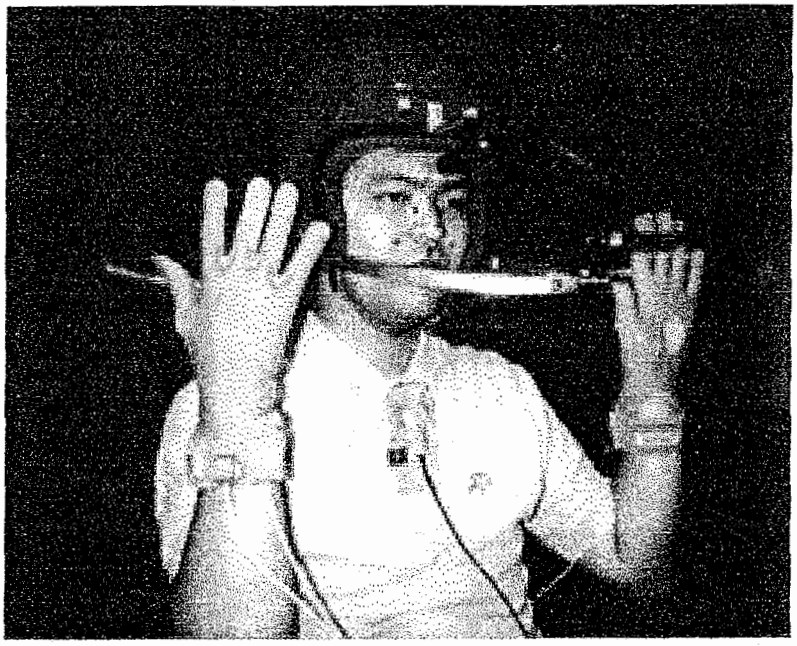


図1:臨場感通信会議参加者

[1] 鼻には表情筋がなく、どのような表情を表出したときも動かないものと仮定している。

2.実時間表情検出・再現系の処理概要

臨場感通信会議システムにおいて人物の表情を再現するためには、１章で述べたように、各会議参加者の表情を送信側で検出し、受信側での３次元顔モデルで実時間再現を実現する必要がある。臨場感通信会議における表情再現のための処理は、図２の（１）～（３）で示される３つのモジュールから構成される。

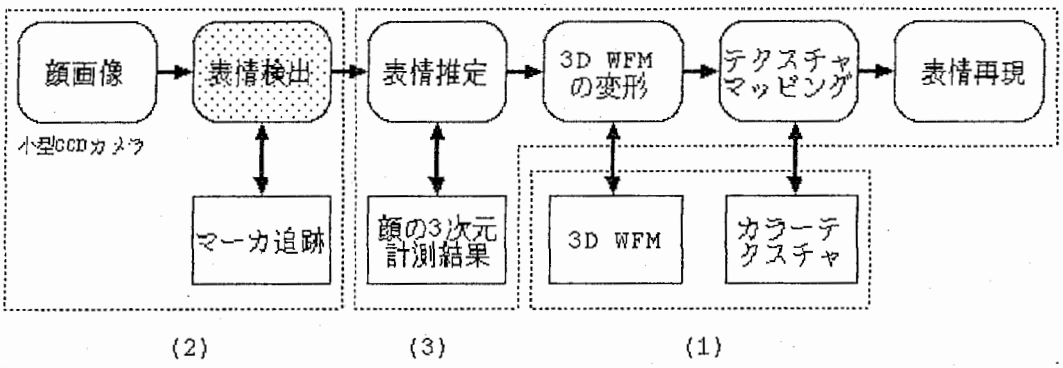


図2:処理の流れ

2.1.顔の３次元モデリング

人物画像の３次元モデリングは、図３のように 3D スキャナ（Cyberware 社製 Color 3D Digitizer）を用いて行われている。3D スキャナを用いると、計測対象の周囲を回転しながら Laser Stripe を照射し、各視線方向の距離情報と計測対象の色彩情報であるカラーテクスチャ情報も併せて入力することが可能である。測定された３次元形状データを 3D モデルの構築が可能なように三角パッチの集合体である３次元顔モデル 3D WFM に変換し、表情の検出にしたがって WFM モデルの頂点を駆動する。そのとき同時に、三角パッチの変形に応じて間引きや補間が行わたテクスチャマッピングが施される。

被測定者は、素顔と図４(ａ)のように、顔料を用いて無数のドットを描画した顔の２種類を測定する。描画されたドットにより図４(ｂ)のように各ドットを頂点とする3D WFMが生成される。これにテクスチャマッピングを施したのが図４(ｃ)である[2]。

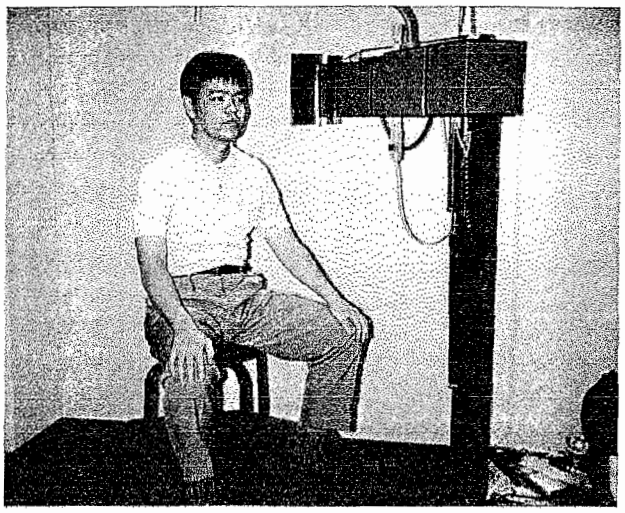


図 3:３Ｄスキャナによる３次元計測

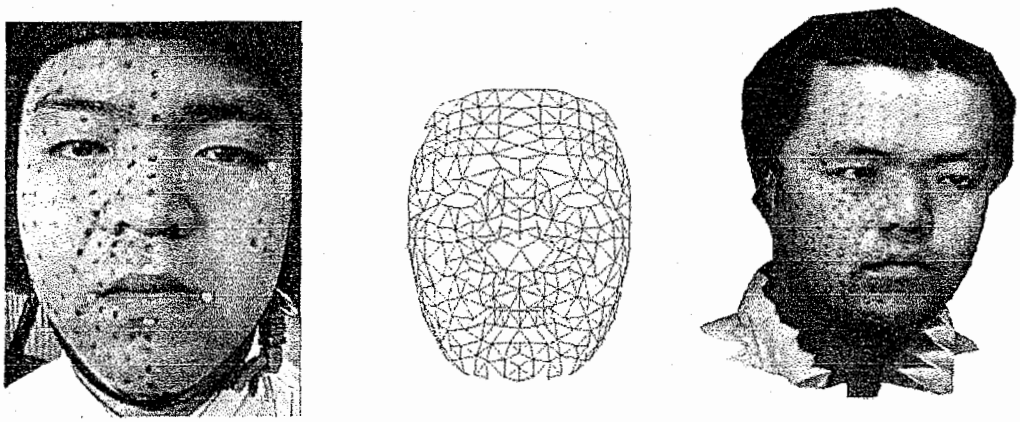


図 4:(ａ)描画されたドット(左)、（ｂ）生成された三角パッチ(中)、（ｃ）テクスチャマッピングを施したＷＦＭ(右)

[2] 会議のときに用いられるのは素顔のテクスチャ情報である。

2.2.実時間表情検出

表情検出系では実時間処理を実現するため図５で示すように、あらかじめ会議参加者の顔に表情検出用のマーカを、顔の表情筋の動きを考慮して添付する。

より忠実度の高い表情を再現するために、表情検出で用いられるマーカは、表情の特徴がよく現れる部分に添付される。顔表面の多くの部分でその動きを追跡するほど、すなわちマーカの数が多いほど、忠実度の高い表情の再現が可能である。しかし、マーカ追跡速度、会議参加者への負担から、図４(ａ)のすべての位置にマーカを添付するのは現実的ではない。そこで顔を額、眼、頬、上唇、下唇、顎の6つの領域に１個以上のマーカを添付する。ここで、領域分割は代表的な表情筋の構造や動作に基づくとともに、例えば眼の近傍のマーカの動きが口の動きの再現結果に影響しないように決定する。[4]

表情検出系は、大きく４つの部分に分けられる。(１)参加者の被るヘルメットに固定した小型 CCD カメラからマーカの添付された顔画像を得る撮影部、(２)専用のマーカ抽出装置を用いて、撮影部で得られた顔画像からマーカのみを抽出するマーカ抽出部、(３)ビデオキャプチャボードを搭載したＰＣにより、マーカ抽出部で抽出されたマーカの動きを追跡する事によって表情変化を検出するマーカ追跡部、(４)ヘルメットに取り付けられた表情検出用のカメラ、もしくは視線検出専用のＣＣＤカメラから入力される目付近の映像から黒目の位置を算出し、視線検出を行う視線検出部の４つである。今回は、マーカ追跡部と視線追跡部の研究を行った。

算出されたマーカと目の移動情報は再現系へと伝送され、表情再現を実現する。

図5:会議参加者に添付されたマーカ

2.3.実時間表情再現

表情再現部では、(2)におけるマーカの追跡結果を用いて、適宜 3D WFM を変形し、これにテクスチャマッピングを施すことにより、表情を再現する。従来法では、マーカの追跡結果を、あらかじめ定められたモーションルールにしたがって、3D WFM の変形情報に変換していたが、モーションルールに定められていない表情の入力に対する再現の忠実性などに問題があった。そこで、従来、困難であった任意の表情の忠実度の高い再現と高速な処理を実現するために、あらかじめ人間の顔に多数のドットを描き、基準となる各種表情の表出時における各ドットの３次元位置計測を行い、そのデータを用いてマーカ追跡結果を３次元顔モデル変形情報に変換することにより実時間表情再現を行う手法を提案し、これらの問題を解決している[3],[6]。

3.表情検出系の構成

2.2.で述べたように、表情再現系は撮影部、マーカ抽出部、マーカ追跡部、視線検出部に分かれている。

今回のシステムでは、1で述べた従来システムにおける精度、速度の問題を解決するために、マーカ抽出を行う専用のハードウェアを構築作成した。また、従来のシステムにおけるバスの転送速度の問題と、ＵＮＩＸを使用することによって発生するリアルタイム性の問題を解決するため、イベント駆動方式の MS-Windows ver3.1 と高速な PCI バスを持つＰＣを用いた。

3.1.顔画像撮影部

表情検出系において、会議参加者は顔に皮膚の色には存在しない色である青色のマーカを貼付し、小型 CCD カメラを取り付けたヘルメットを被る。

従来のシールを用いたマーカでは、口の周りなど変化の激しい箇所において、マーカが唇に隠れてしまい正確な位置検出が困難になることが多々あったため、マーカは球形のプラスチックでなるべく軽いものを採用した。

3.2.マーカ抽出部

マーカ抽出部では、図6に示すようなマーカ抽出装置を作成し、会議参加者の顔画像からマーカのみの画像を抽出している。この装置は、小型 CCD カメラから入力される映像信号中から任意の色のみを抽出し、それ以外の色は除去するようになっている。顔画像は小型 CCD カメラ（NTSC 方式）で入力され、色分離回路（Y/C 分離回路）を経由して RGB 信号に変換される。ここで、バースト信号は映像信号から分離され、最後の出力段階で再び信号に付加され NTSC 信号としてマーカ追跡装置に送られ PLL の GEN lock 信号として用いる。水平同期信号による GEN lock

信号と比較し、fsc（カラーサブキャリア）によるPLLのGEN lockはジッタが少なく、マーカのような円形の形状を抽出する場合には安定性に優れている。

貼付するマーカは、３.１で述べたように青色を用いているが、Blue信号の出力を行うと、マーカ追跡装置側で2値化処理を行わねばならず処理時間が増える。そこで、アナログ方式のコンパレータを用いて、実時間で2値化処理を行っている。また、青色による単純な制御のみでマーカ抽出を行うと、光が当たり反射した白色や唇に含まれる紫色など青の成分を含んでいる色が入力されると、マーカの青色と誤認識して出力を行う場合がある。そこで、一定以上の赤色と緑色を含む場合にはコンパレータの出力を制御して誤認識を防止するホワイトキャンセラ回路を導入しこの問題を解決している。図７はマーカ抽出装置の通過前後の顔画像の例である。

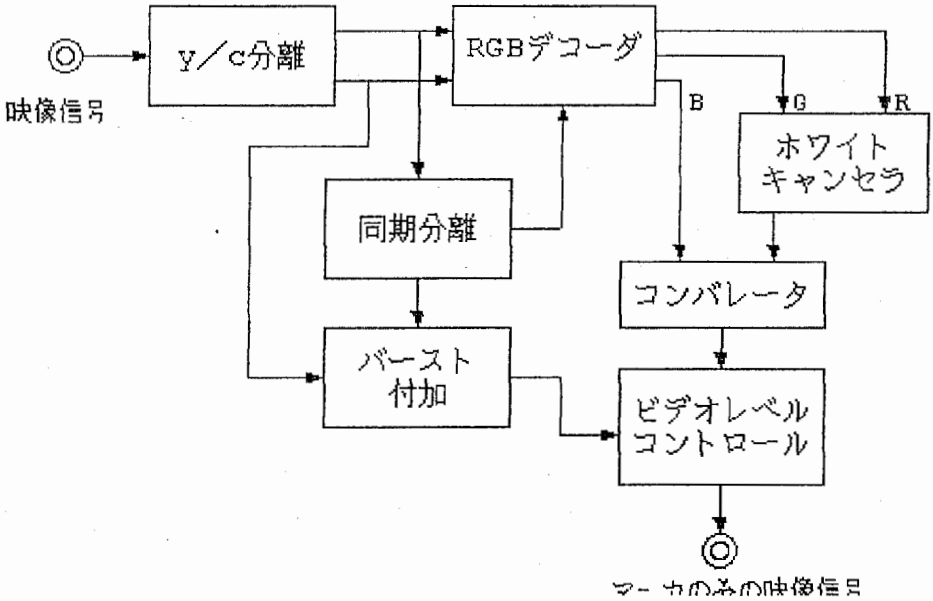


図6:マーカ抽出装置ブロック図

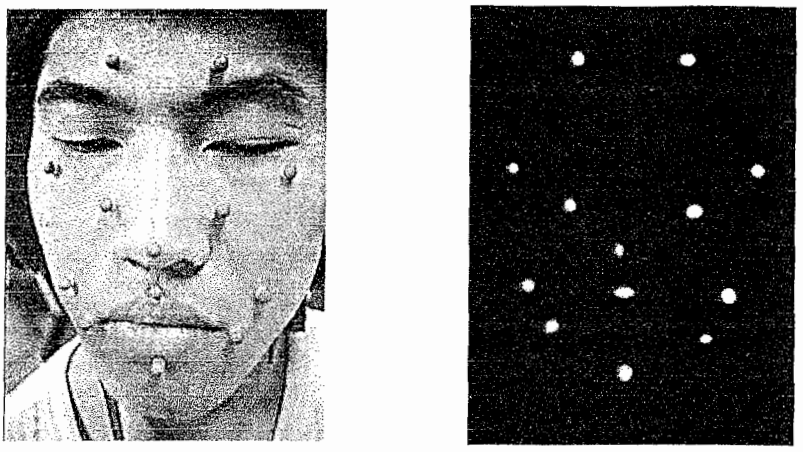


図7:（a）マーカ抽出装置通過前の顔画像（左）（b）同通過後の顔画像（右）

3.3.マーカ追跡部

実時間のマーカ抽出を専用ハードウェア化したため、マーカの追跡はＰＣ上で行えるようになった。ビデオキャプチャボードから入力された顔画像を、高速バス（PCI bus）を介してメモリに転送し、４章で述べるアルゴリズムを用いてマーカ追跡処理を行っている。その後各マーカの２次元ベクトルの移動量を計算し、ネットワークを経由して表情再現系に転送される。

3.4.視線検出部

視線検出部では、ヘルメットに取り付けた小型 CCD カメラから得られる顔画像を用い、視線の検出を実現している。

従来の手法では、視線検出専用のカメラを用いていたため、装着装備が増大し会議参加者への負担が大きく、検出は右目だけで左目は右と同じ動きをするものとしてい(左目の検出も行うにはさらにもう１つカメラをヘルメットに取り付けなければならない)。るため、ウィンク等の左右の目が独立動作を行った場合に対応が困難であった。

現行のシステムでは、マーカ追跡用カメラから得られる顔画像中の目を走査し黒目の位置を検出する手法が用いられている。

4.マーカ追跡部

4.1.マーカ追跡部の構成

マーカ追跡部は、ハードウェアのＰＣは、マーカの顔画像を取り込むためのビデオキャプチャボードと、再現系との通信を行うための、ネットワークカードから構成されている。

4.2.マーカ追跡プログラム

マーカ追跡プログラムは、MS-Windows ver3.1 上で動作するソフトウェアで(付録 C 参照)、顔に添付されたマーカを追跡し表情の変化を検出する。

マーカ追跡開始時には、会議参加者はマーカ初期値を与えるため無表情を表出する。マーカ抽出装置からパソコン内の画像取り込み装置に入力された顔画像中から、同装置によってマーカのみが抽出された 2 値画像をラスタスキャンし、マーカをラベルとしてその重心を求め各マーカの初期位置として保存する。以降図 9 に示すマーカー毎に指定された追跡範囲内でその位置を追跡し、移動した位置と初期位置との差分から 2 次元移動ベクトルを算出し送信している。

マーカ追跡は、マーカ抽出装置により 2 値化された顔画像(青色であればピクセルは ON、それ以外のいろはすべて OFF とする)において、指定された追跡範囲内をラスタスキャンし、マーカを構成しているピクセル群の重心と面積を算出している。

4.3.マーカの重心の検出

ｘ座標の合計をｓｘ、ｙ座標の合計をｓｙ、ON ピクセルの数をｓｄとすると、マーカの重心（ｍｘ，ｍｙ）は、

$$m_x = s_x / s_d,\ m_y = s_y / s_d \qquad (1)$$

で求められる。本稿では、追跡範囲内のONピクセルのｘ座標の合計、ｙ座標の合計を求めONピクセルの数で割り平均値を計算している。

たとえば、図８で示すように、仮に追跡範囲を縦横とも１０ピクセルとして、追跡範囲をラスタスキャンを行うと、(５，２)の座標でピクセルがＯＮなので、このピクセルの座標(５，２)を保存し１つめのピクセルがあったことを記憶しておく。次の座標(６，２)を調べるとこれもＯＮであるので、前に保存した座標(５，２)に子のピクセルの座標を加える。よってｘ座標の合計は１１、ｙ座標の合計は４に、そしてＯＮピクセルの数は２になる。以下同様の作業を座標(９，９)まで行うと、ＯＮピクセルの数は２１個、ｘ座標の合計は１２６、ｙ座標の合計は８４となる。よって重心（ｍｘ，ｍｙ）は、式（１）より

$$m_x = 126 / 21 = 6,\ m_y = 84 / 21 = 4$$

となり、（６，４）が重心位置となる。

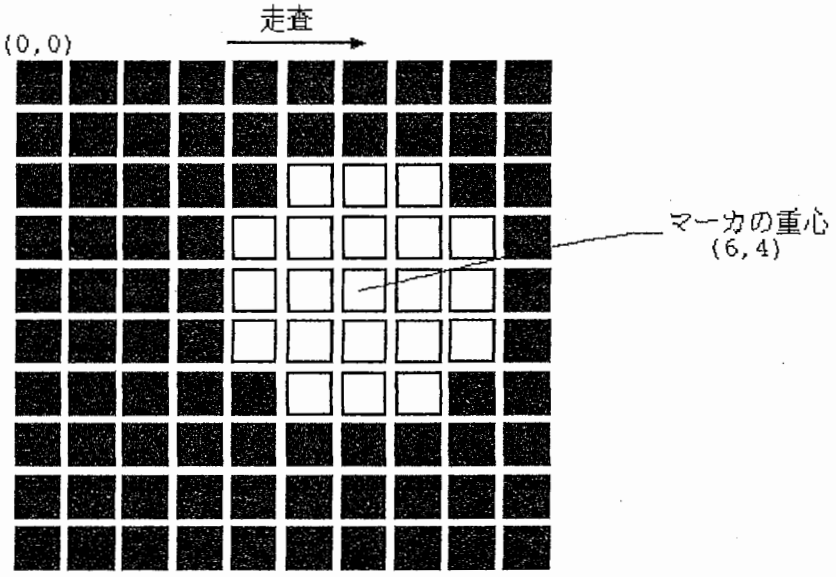


図8:マーカ検出の例

4.4.マーカ追跡範囲

マーカ追跡範囲は図９(ａ)に示すように、予測されるマーカの移動範囲全体をマーカ追跡範囲とするものと、図９(ｂ)マーカの現在位置の重心を中心とする狭い範囲内で移動するものの２種類に分類して処理を行っている。(ａ)は、マーカを見失った場合にも引き続き追跡を行えるが、処理時間が長いという問題を持っている。(ｂ)は、マーカが指定された範囲内を移動している場合には、操作時間が短く実時間処理に優れているが、何らかの原因でマーカ追跡範囲を超えた場合には、マーカを見失ってしまう問題がある。

(ａ)の追跡範囲は、下唇によって隠れてしまうことのある顎のマーカの追跡に用いられ、(ｂ)の追跡範囲はその他のマーカの追跡に用いられ、全体の処理速度を向上させている。

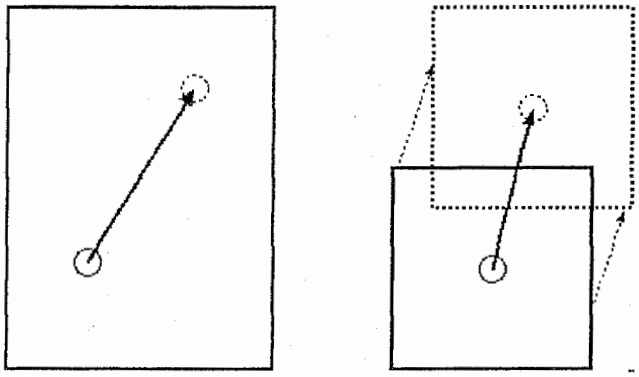


図9:(a)検出範囲固定 (b)マーカ位置に応じて移動

5.再現系への送信

検出された表情、すなわち無表情の状態からの各マーカの２次元移動ベクトルは、ソケットをもちいた通信プロトコルによって表情再現系に送信される。

5.1.ソケット

ソケットは、OSI(Open System Interconnection)参照モデルのトランスポート層とユーザプログラムのAPI(Application Program Interface)である。従って、ソケットを用いると、ユーザプログラム側からトランスポート層のプロトコルモジュールを呼び出すことができる。図 10 のようにプロトコルモジュールとしてTCP(Transmission Control Protocol)と UDP(User Datagram Protocol)の２つがありユーザ適当なものを選択する。

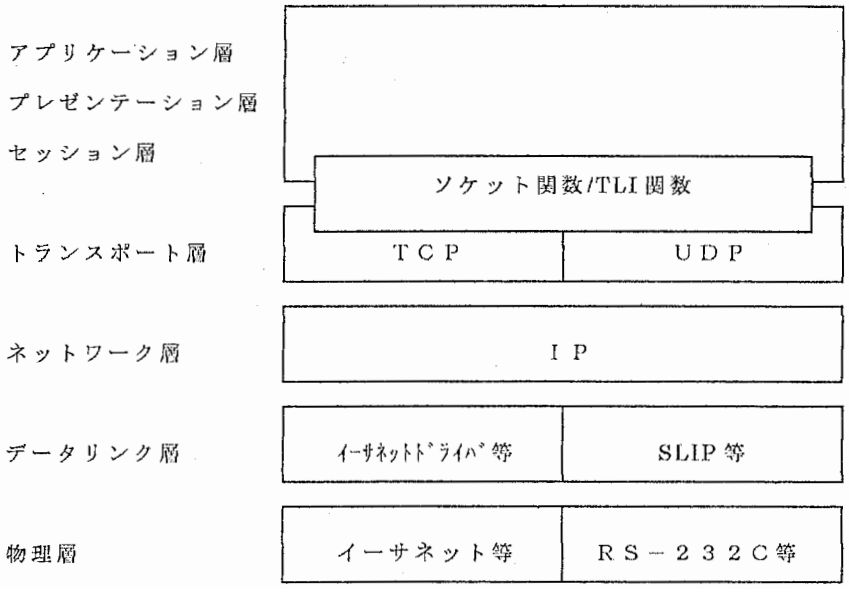


図10:ネットワークプログラムの API

5.2.通信プロトコル

コネクション型サービスは、プロセスとプロセスが通信を行うとき、あらかじめ論理的な通信経路を用意してから通信を行うタイプのサービスである。TCP はコネクション型の信頼性のある全二重バイトストリーム通信[3]を提供するので、6.1.で述べた２つの API のうち TCP を用いる。

図 11 のように、TCP を利用するコネクション型のプログラムでは、listen(接続の受付け準備),connect(接続要求を出す),accept(接続要求の受付け)によってコネクションの確立を行わなければならない。またデータの送受信は、送信側の write、受信側の read によって行われる。

実際の表情検出系と表情再現系の２つのプロセス間通信は、以下に示すような手順によって行われる。

マーカ追跡プログラム(クライアント)を実行する前に、表情再現系(サーバ)のプログラムを立ちあげる。サーバ側、クライアント側ではそれぞれソケットが生成される。クライアント側が接続要求(connect)を出し、サーバ側の接続要求受け付け(accept)によって、コネクションが確立される。

再現系はマーカ追跡プログラムからの送信開始の合図がくるまで待機状態となる。

マーカ追跡プログラムは、すべてのマーカの初期位置が検出されたら、サーバ側に送信開始を知らせるために任意データを送信する。それと同時に、マーカの追跡を開始する。

送信開始の合図を受け取った再現系は、データ送信要求コードとして“１”を送信する。送信要求を受け取ったマーカ追跡プログラムは、送信バッファにある各マーカの２次元移動ベクトルを再現系に送信し、引き続きマーカ追跡を行う。再現系は受け取った移動ベクトルから、３次元空間におけるマーカの移動ベクトルを算出し、各マーカに対応するＷＦＭの頂点を駆動し表情を再現する。

---

[3]全二重:通信を行っているプロセスが同時に送信を行える。/ バイトストリーム:送り手側がデータを送ったときのデータの境界が受け手側で見えなくなるようなタイプのサービス。

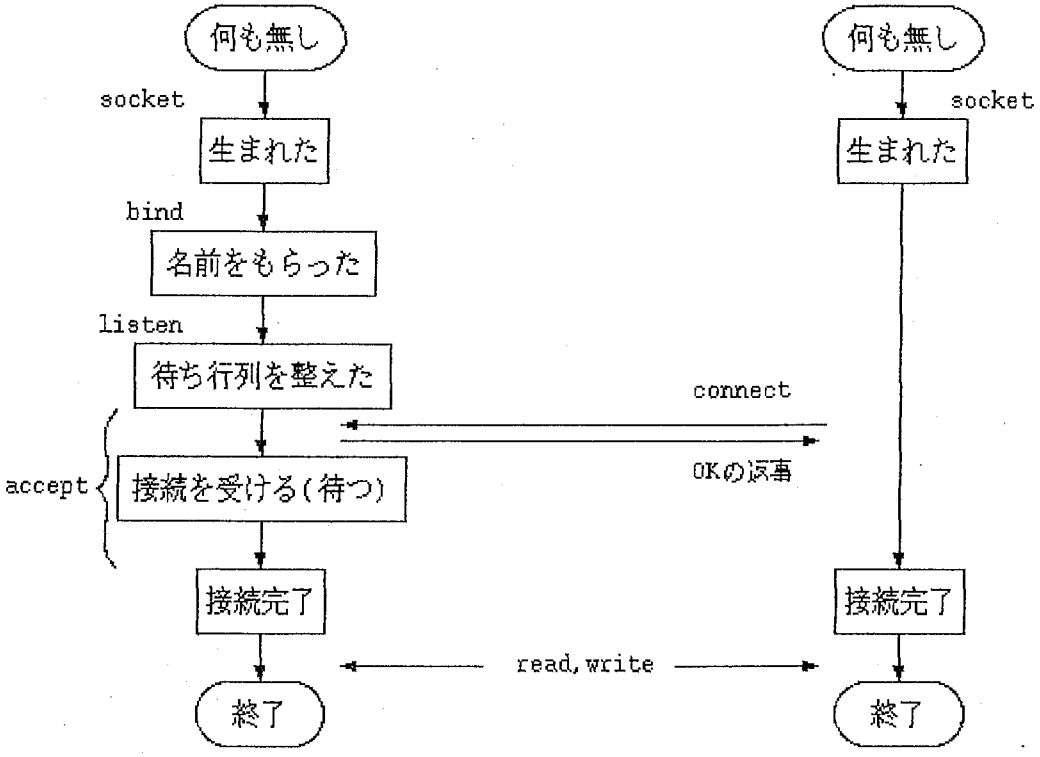


図11:ソケットの生成と接続処理

5.3.送信要求時に発生するWindowsメッセージ

従来の実時間マーカ追跡プログラムは、グラフィックWSのマルチタスクOS上で動作していたので、マーカ検出の処理と通信の処理は並列に実行され、プログラムでは特別な処理は必要なかった。しかし、今回のマーカ追跡プログラムが動作するMS-Windowsは完全なマルチタスクOSではなく、6.2で述べた表情再現系(サーバ)からのデータ送信要求を待つためのループに入るとマーカ追跡の処理が停止してしまった。

このような場合、一定周期のハードウェア割り込みを利用して、常に送信要求を監視するという方法が一般的であるが、Windowsではイベント駆動方式の疑似マルチタスクを実現するために、メッセージなるものが用意されているのでこれを用いて問題を解決した。

メッセージとは、ユーザインタフェイスにおいて生じた変更に関する情報である。すなわちマウスのクリックや、キー入力によりメッセージが発生する。

Windows アプリケーションは、発生したメッセージが必要なものであれば、それに対する処理を行い、関係なければ無視する。プログラム内で、ある特定のメッセージの発生時に呼び出される関数を、メッセージハンドラ関数という。

Windows でソケットを利用するためのソフトフェア、Winsock のダイナミックリンクライブラリには、サーバからの送信があった場合メッセージを発生する WSAAsyncSelect 関数が用意されている。この関数によりサーバからのデータ送信要求時に WM_SOCKET というメッセージが発生する。

送信するデータが少ないため、データ送信処理の時間はマーカ追跡の処理に比べて圧倒的に短い。そこで、送信処理のプログラムを WM_SOCKET メッセージのメッセージハンドラ関数に記述した。プログラムは通常マーカ追跡処理に専念し、WM_SOCKET メッセージが発生したときのみ送信処理が行われる。

これにより、見かけ上マーカ追跡の処理と、データ送信の処理が並列に実行されているようになる。

6.まとめ

6.1.プログラム実行結果

以上の機器、アルゴリズムから作成したＰＣ上で動く実時間マーカ追跡プログラムを実際にプログラムサイズは、134,896バイトである。図12に実行画面を示す、画面中の黒い部分が入力された顔画像で、x,y軸が反転されている。すなわち向かって左側が口、右側が目である。またグラフィックワークステーション上で動作していた旧システムのマーカ追跡プログラムと比較した結果が表１である。

１秒間あたりのマーカ検出回数は、旧システムでは、１５～１８回であったが、今回のプログラムでは１３個のマーカ追跡を 59～60 回行えた。これはＮＴＳＣ(National Television System Committee) 方式のカメラの grab 割り込み[4]の回数に相当（ＮＴＳＣ方式はアスペクト比が４：３で、走査線とフィールド周波数が525本／59.94Hzである）しており、カメラから入力されるすべてのフレームにおいてマーカ追跡が可能である、即ち実時間の表情検出が可能であることを示している。

今回のＰＣ用実時間マーカ追跡プログラムの開発によって、臨場感通信会議システムにおける、表情検出の性能が向上した。

[4] ビデオキャプチャボードが画像を取り込む度に発生するハードウェア割り込み。

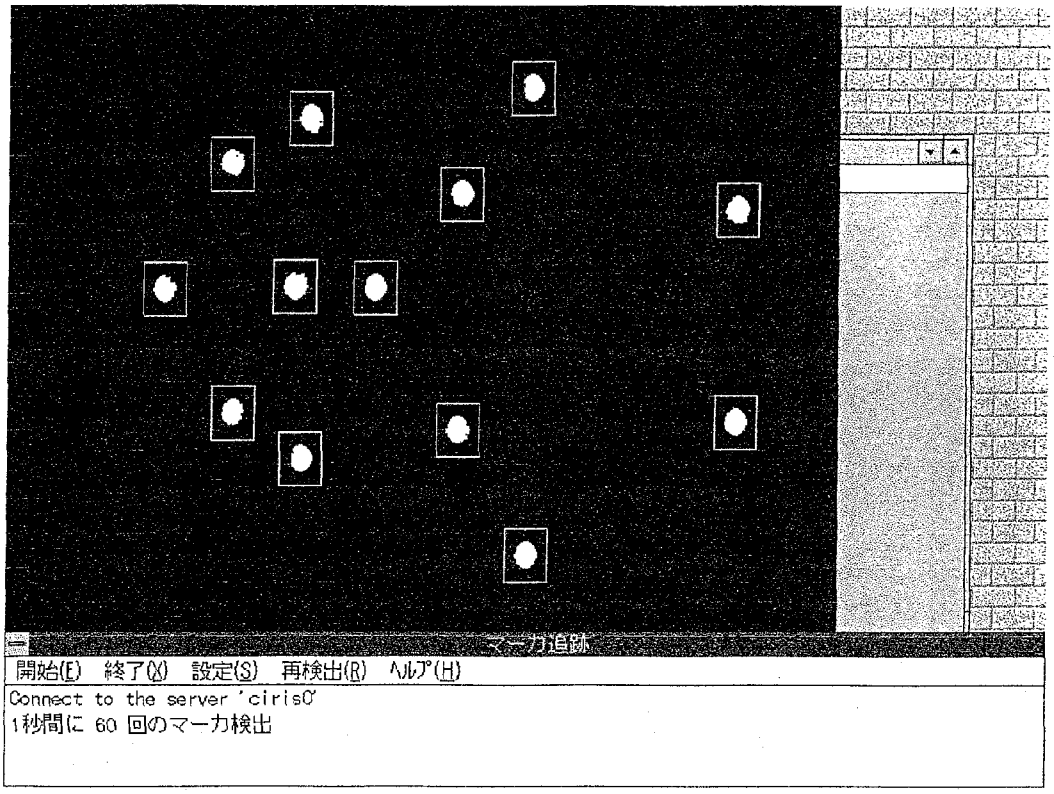


図12:表情検出プログラム実行画面

表 1:新旧システムの比較

	新システム	旧システム
マーカ（視線）検出回数	５９～６０回／秒	１５～１８回／秒
稼動ハードウェア	パーソナルコンピュータ	ｸﾞﾗﾌｨｯｸﾜｰｸｽﾃｰｼｮﾝ
ｵﾍﾟﾚｰﾃｨﾝｸﾞｼｽﾃﾑ	ＭＳ－Ｗｉｎｄｏｗｓ	ＵＮＩＸ

6.2.今後の課題

今回のシステムでは、会議参加者は顔にマーカを添付し、さらにカメラを取り付けたヘルメットを被らなけらばならない。これは、表情検出系にかかる負荷を軽減するための措置であったが、利用者にとって快適な環境であるとはいえない。

現在ＡＴＲ通信システム研究所では、離散コサイン変換（Discrete Cosine Transformation）を用いたマーカ不要の実時間表情検出の開発を行っている。これにより、会議参加者は顔にマーカを添付する不快感から解放される。

今後は、3次元計測の手法を応用し複数台のカメラを用いて、マーカと同じく会議参加者へ負担を与えている、ヘルメットの不要な表情検出の手法の開発を行う。

謝辞

ＡＴＲでの研修の機会を与えてくださった（株）ＡＴＲ通信システム研究所 葉原耕平会長、寺島信義社長に感謝致します。

有益な御助言御助力をいただいた知能処理研究室の岸野文郎室長、主幹研究員の大谷淳氏、楽しく有意義な研究テーマを与えてくださり、研修中様々な面でお世話頂いた研究員の海老原一之氏に感謝致します。

また、質問に親身になって答えてくださった知能処理研究室の皆様、（株）ＣＳＫの皆様に感謝致します。

研修中いろいろ御気遣い頂いた加藤恭子教授、いっしょに研究した期間は短かったけれど、遠くから励ましてくれた加藤研究室のメンバー、その他多くの友人にも感謝致します。

参考文献

[1]岸野文郎『臨場感通信』、テレビ誌、Vol.46、No.6、pp.698-702(Jun. '92)

[2]J.Ohya, Y.Kitamura, F.Kishino, N.Terashima, H.Ishii and H.Takemura.."Virtual Space Teleconferencing: Real-time Reproduction of 3-D Human Image",VCIR Vol. 6, No. 1, pp.1-25(Mar. '95)

[3]大谷淳、北村泰一、竹村治雄、岸野文郎.『臨場感通信会議システムにおける３次元顔画像の実時間表示』、信学技報 HC92-61.(Jan. '93)

[4]鈴木紀子、大谷淳、岸野文郎.『３次元データの重要度に応じた表情再現品質の検討』、信学技報 HC94-83(Feb. '95)

[5]大谷淳、岸野文郎.『遺伝的アルゴリズムを用いたマルチ画像からの人物の姿勢検出の検討』、信学技報 PRU93-122.(Jan. '94)

[6]海老原一之、大谷淳、鈴木紀子、岸野文郎.『３次元計測に基づいた顔画像の実時間表情再現方法』平７信学ソサエティ大会,(A-184)

[7]坂口竜己、森島繁生、大谷淳、岸野文郎.『3次元計測に基づく顔表情変化の分析と合成』信学技報 HC93-74.(Jan.'94)

[8]P.Ekman and W.V.Friesen.(工藤力 訳編)『表情分析入門』誠新書房(Apr. '87)

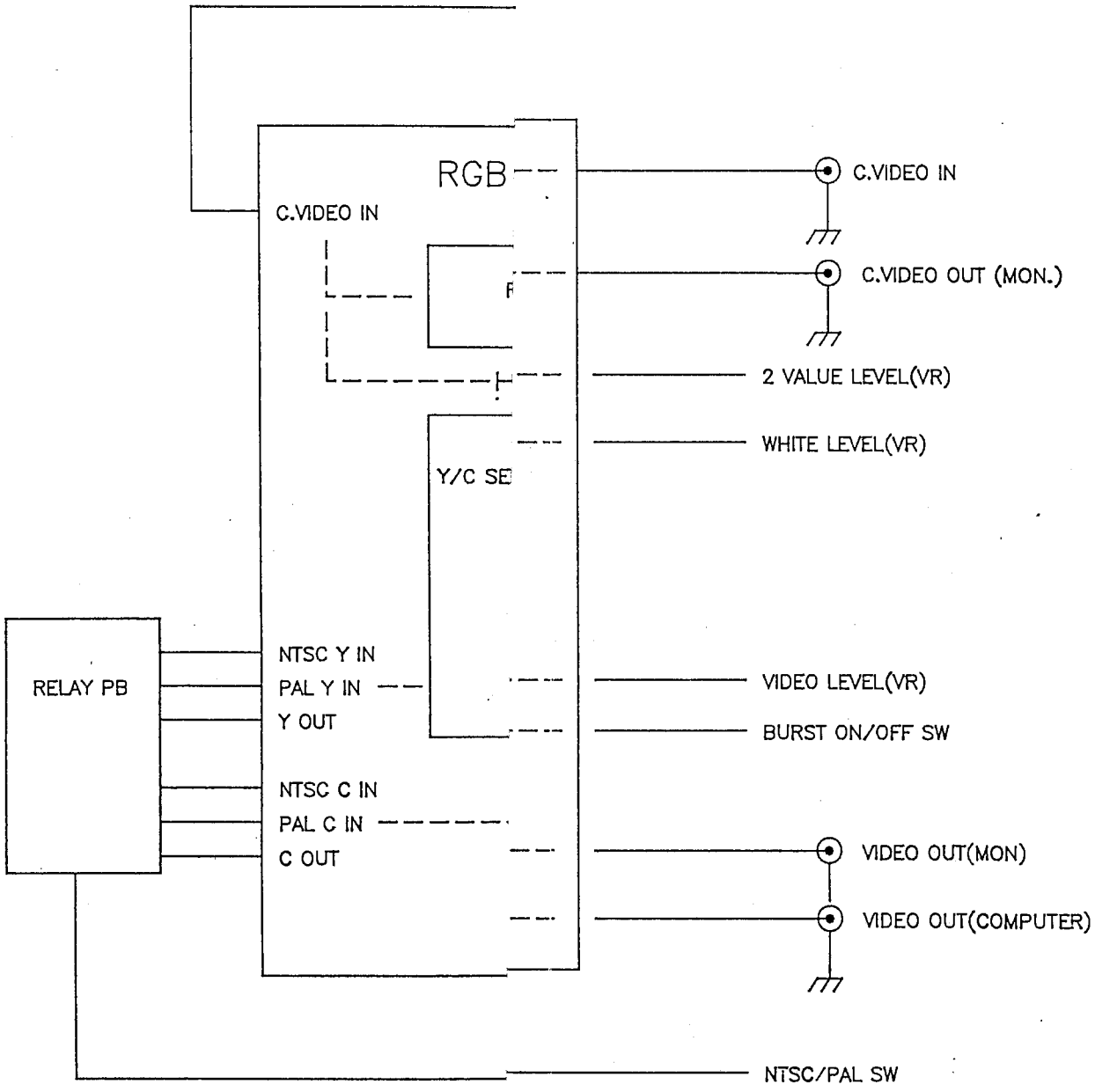
[9]海老原一之、山田正紀、大谷淳、岸野文郎.『臨場感通信会議システムにおける３次元顔画像処理』、信学技報 OFS95-29, IE95-61(Sep. '95)

[10]John Davidson "An Introduction to TCP/IP", Springer-Verlag New York Inc.
[11]金内典充、今安正和（羽山博監修）『プログラミングリファレンス UNIX ネットワークプロトコル』オーム社(Nov. '93)
[12]石坂充弘『情報通信プロトコル』オーム社(Feb. '87)
[13]Microsoft 出版『Windows API－関数』
[14]Microsoft 出版『Windows API－構造体』

付録A　回路図

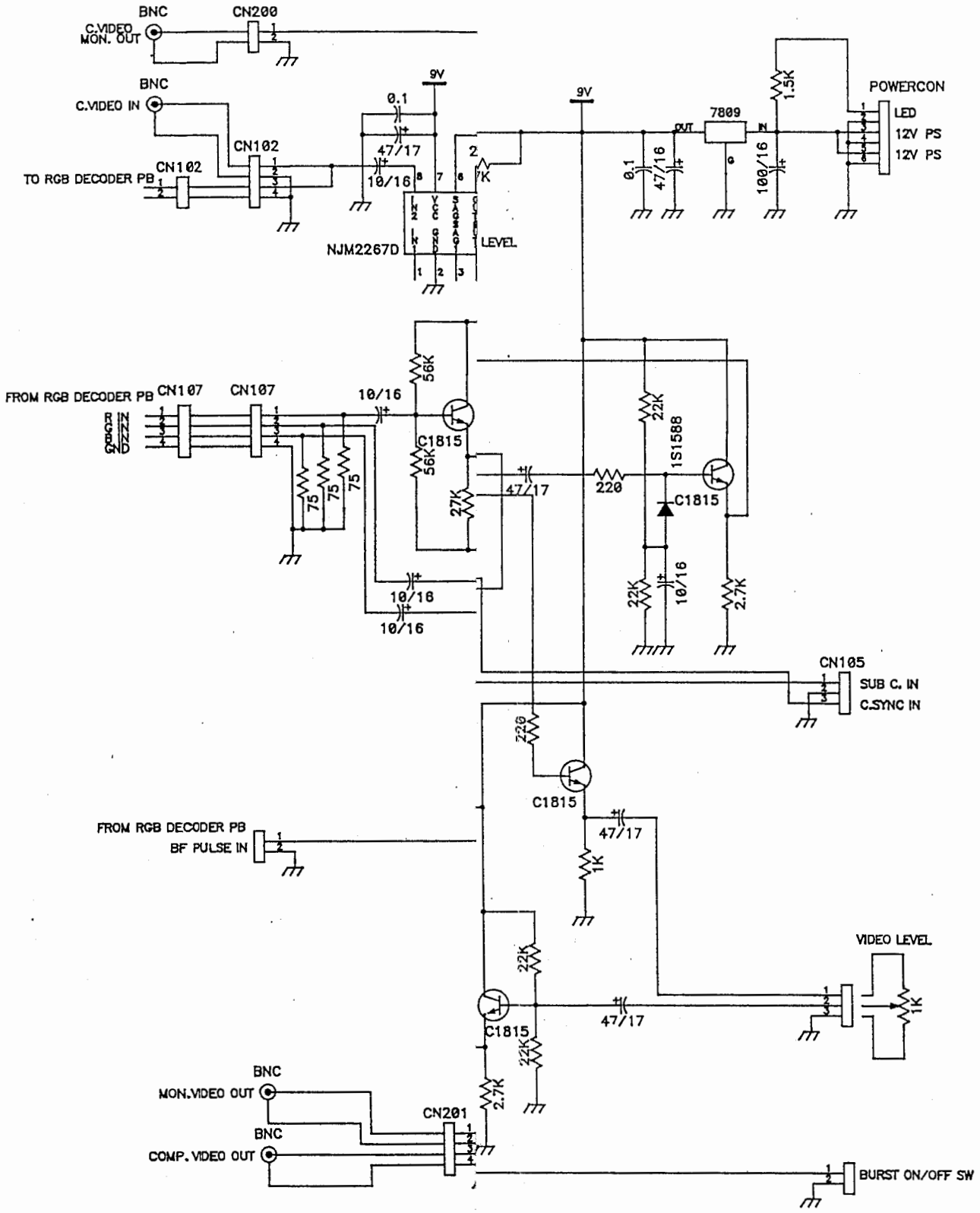
本システムで用いたマーカ抽出装置は、アナログ方式の比較器を用いて実現している。ここで用いられるマーカは、任意に設定可能であるが、グンゼ産業 の 「Mr.カラー」 の80番 (コバルトブルー)を用いた場合が最も効果が得られた。また、マーカはエアガンの弾丸（８ｍｍ）を用いている。

NTSC信号をＲＧＢ信号に変換する手法は、色々と考えられるが、今回は、入手が容易な一般的なＩＣを用いて実現している。もし、小型化を望む場合には、富士通（株）のＮＴＳＣデコータＩＣを用いれば、１チップでこれが実現可能となる。



RGB DECODER BLOCK DIAGRAM

Tue Sep 19 1995
ATR002.S01



I/F CIRCUIT FOR R.G.B DECORDER

Tue Sep 19 1995

ATR001.S01

```
 1:// capboard.h : for matlox Comet
 2://
 3:////////////////////////////////////////////////////////////////////////////
 4:
 5:// Class definition
 6:
 7:class CCapBoard : public CObject
 8:{
 9:public:
10:    CCapBoard();
11:    DECLARE_DYNCREATE( CCapBoard )
12:
13:// Member variables
14:
```

```
 1://///////////////////////////////////////////////////////////////////////////
 2:// Class CCapture : CObject
 3:
 4:
 5:// Class definition.
 6:
 7:class CCapture : public CObject
 8:{
 9:public:
10:    CCapture();
11:    DECLARE_DYNCREATE( CCapture )
12:    CCapture(CView*);
13:
14:// ﾒﾝﾊﾞ変数
15:private:
16:    CView* m_pView;            // ﾋﾞｭｰへのﾎﾟｲﾝﾀ
17:
18:public:
19:    short Buffer[20000];       // 画像取り込用ﾊﾞｯﾌｧ
20:    MarkConf m_Conf;
21:    VideoSize m_VSize;
22:    long m_FrameWin;
23:    int m_SigCaught;
24:    int m_ShmID;
25:    unsigned long m_Counter;
26:    int m_Chan;
27:
28:// ﾒﾝﾊﾞ関数
29:public:
30:
31:    // ｷｬﾌﾟﾁｬﾎﾞｰﾄﾞ関連
32:    int   CaptureInit();
33:    void  CaptureQuit();
34:
35:    long  Addr(long, long);
36:    short Bufpix(int, int);
37:    void  Putpix(int, int, short);
38:
39:    void  CaptureStop();
40:    void  CaptureStart();
41:    void  ChangeChannel();
42:
43:    // ﾏｰｶ追跡関連
44:    int   ReadConfig(char *);
45:    void  FreeMalloc(int x);
46:    void  ScanMarker(int, int, int*, int*, long*, int, int);
47:    int   ScanMarkerPosition(Mark marks[], Mark redMarks[]);
48:    void  ScanMarkMove(Area area, Mark orgPos, Mark *pos);
49:    int   SearchNearlyMarker(int px, int py, Mark *mark, int mn);
50:    int   readCenterLineMarkerConf(FILE *fp, int numb, int *lines);
51:    int   readUpperLineMarkerConf(FILE *fp, int numb, int *lines);
52:    int   readBottomLineMarkerConf(FILE *fp, int numb, int *lines);
53:};
54:
```

```
 1:// GRABDLG.H : header file
 2://
 3:
 4://///////////////////////////////////////////////////////////////////////////
 5:// CGrabDlg dialog
 6:class CCapture;
 7:class CTraceView;
 8:class CGrabDlg : public CDialog
 9:{
10:// Construction
11:public:
12:    CGrabDlg(CWnd* pParent = NULL); // standard constructor
13:    CGrabDlg(CClientDC* pDC);
14:    CGrabDlg(CCapture* pCapture);
15:    BOOL Create();
16:
17:// Member variables
18:public:
19:    CWnd *m_hParent;
20:    CTraceView* m_pView;
21:    CCapture* m_pCapture;
22:
23:// Dialog Data
24:    //{{AFX_DATA(CGrabDlg)
25:    enum { IDD = IDD_DIALOG3 };
26:        // NOTE: the ClassWizard will add data members here
27:    //}}AFX_DATA
28:
29:// Member functions
30:public:
31:    void SetView(CTraceView* pView);
32:
33:// Overrides
34:public:
35:
36:protected:
37:    virtual void DoDataExchange(CDataExchange* pDX);    // DDX/DDV support
38:
39:// Implementation
40:protected:
41:
42:    // Generated message map functions
43:    //{{AFX_MSG(CGrabDlg)
44:    afx_msg void OnLButtonDown(UINT nFlags, CPoint point);
45:    //}}AFX_MSG
46:    DECLARE_MESSAGE_MAP()
47:};
48:
49:#define WM_DLGMSG WM_USER+301
50:
```

```
1:extern "C" {
2:extern int GrabFlag;
3:void interrupt GrabInt(void);
4:}
5:
```

```
 1:// mainfrm.h : interface of the CMainFrame class
 2://
 3://///////////////////////////////////////////////////////////////////////////
 4:
 5:class CMainFrame : public CFrameWnd
 6:{
 7:protected: // create from serialization only
 8:    CMainFrame();
 9:    DECLARE_DYNCREATE(CMainFrame)
10:
11:// Attributes
12:public:
13:
14:// Operations
15:public:
16:
17:// Implementation
18:public:
19:    virtual ~CMainFrame();
20:// Precreate
21:    virtual BOOL PreCreateWindow(CREATESTRUCT &cs);
22:
23:#ifdef _DEBUG
24:    virtual void AssertValid() const;
25:    virtual void Dump(CDumpContext& dc) const;
26:#endif
27:
28:// Generated message map functions
29:protected:
30:    //{{AFX_MSG(CMainFrame)
31:    //}}AFX_MSG
32:    DECLARE_MESSAGE_MAP()
33:};
34:
35://///////////////////////////////////////////////////////////////////////////
36:
```

```
 1:#ifdef __cplusplus
 2:extern "C" {
 3:#endif
 4:
 5:int compMTempLength(const void *v1, const void *v2);
 6:int compMTempY(const void *v1, const void *v2);
 7:int compMTempX(const void *v1, const void *v2);
 8:void lineUPMarker(MarkConf *conf, Mark grn[], Mark red[], Mark marker[], int *mn);
 9:
10:#ifdef __cplusplus
11:}
12:#endif
13:
```

```
 1:// ﾏｰｶ追跡に関する定義ﾍｯﾀﾞﾌｧｲﾙ
 2:
 3:#define RLevel(x) ((x & 0x001f) >>  0)
 4:#define GLevel(x) ((x & 0x02e0) >>  5)
 5:#define BLevel(x) ((x & 0x7c00) >> 10)
 6:
 7:// config. file
 8:#define GRAB_WIN_X  640
 9:#define GRAB_WIN_Y  480
10:
11:// ﾏｰｶ検索範囲のﾏｰｼﾞﾝ
12:#define LEFT_MARGIN          30
13:#define RIGHT_MARGIN         30
14:#define TOP_MARGIN           30
15:#define BOTTOM_MARGIN        30
16:
17:// ﾏｰｶと認識する下限
18:#define MARK_LEVEL           0x7f00
19:#define RED_MARK_LEVEL       0
20:
21:// ﾏｰｶ検索範囲
22:#define MARK_SCAN_WIDTH      40
23:#define MARK_SCAN_HEIGHT     40
24:
25:#define RED_MARK_SCAN_WIDTH      60
26:#define RED_MARK_SCAN_HEIGHT     200
27:
28:// 眼球追跡に関する定義
29:#define EYE_CLOSE_RATE       40
30:#define EYE_AREA_MOVE_X      100
31:#define EYE_AREA_MOVE_Y      30
32:
33:// その他
34:#define REGION_RATIO         0.035
35:#define PORT_NUM             7000
36:#define PKT_SIZE             128
37:
38:// Type definition.
39:
40:typedef unsigned char uchar;
41:typedef struct {
42:     int x, y;
43:     long area;
44:} Mark;
45:
46:typedef struct {
47:     int xl, xr;
48:     int yb, yt;
49:     int attr;
50:} Area;
51:
52:typedef struct {
53:     int id;
54:     Area area;
55:} MarkID;
56:
57:typedef struct {
58:     int width;
59:     int height;
60:} Size;
61:
62:typedef struct {
63:     int x, y;
64:} VideoSize;
65:
66:typedef struct {
67:     int nOfGreen;
68:     int nOfRed;
```

```
    int nOfCenter;
    MarkID *center;
    int nOfUpper;
    MarkID *upper;
    int nOfBottom;
    MarkID *bottom;
    MarkID eye;
    int eyeOpenID;
    Size eyeMove;
    Size grnScan;
    Size redScan;
} MarkConf;

```

```
 1://{{NO_DEPENDENCIES}}
 2:// App Studio generated include file.
 3:// Used by TRACE.RC
 4://
 5:#define IDR_MAINFRAME                    2
 6:#define IDD_ABOUTBOX                     100
 7:#define IDD_DIALOG1                      102
 8:#define IDD_DIALOG2                      104
 9:#define IDD_DIALOG3                      105
10:#define IDB_BITMAP1                      106
11:#define IDC_SERVER                       1000
12:#define IDC_RADIO2                       1003
13:#define IDC_RADIO3                       1004
14:#define IDC_RADIO4                       1005
15:#define IDC_RADIO5                       1006
16:#define IDC_RADIO6                       1007
17:#define IDC_EDIT1                        1008
18:#define IDC_EDIT2                        1009
19:#define IDC_EDIT3                        1010
20:#define IDC_EDIT4                        1011
21:#define IDC_EDIT5                        1012
22:#define IDC_EDIT6                        1013
23:#define ID_END                           32771
24:#define ID_MENUITEM32772                 32772
25:#define ID_MENUSTART                     32773
26:#define ID_MENUITEM32774                 32774
27:#define ID_RETRY                         32778
28:
29:// Next default values for new objects
30://
31:#ifdef APSTUDIO_INVOKED
32:#ifndef APSTUDIO_READONLY_SYMBOLS
33:
34:#define _APS_NEXT_RESOURCE_VALUE         107
35:#define _APS_NEXT_COMMAND_VALUE          32780
36:#define _APS_NEXT_CONTROL_VALUE          1009
37:#define _APS_NEXT_SYMED_VALUE            101
38:#endif
39:#endif
40:
```

```
 1://///////////////////////////////////////////////////////////////////////////
 2:// Class CSerial : public CObject
 3:
 4:// Class definition
 5:
 6:class CSerial : public CObject
 7:{
 8:public:
 9:    CSerial();
10:    CSerial(HWND);
11:    DECLARE_DYNCREATE( CSerial )
12:
13:// Member variables
14:private:
15:    HWND m_hSWnd;
16:
17:    DCB m_dcb;
18:    int m_idComDev;
19:    int m_Port;
20:    int m_Baud;
21:    int m_Parity;
22:    int m_Length;
23:    int m_Stop;
24:    int m_Qr;
25:    int m_Qs;
26:
27:// Member functions
28:public:
29:    int InitSerial(int, int, int, int, int, int, int);
30:    int ChangeStatus(int, int, int, int);
31:    int SetBreak();
32:    int ClearBreak();
33:    int CloseSerial();
34:    int FlushQueue(int);
35:    int EnableNotifyMsg(int, int);
36:    int GetLastError();
37:    int GetSerialStatus(int);
38:    int Read(unsigned char*, int);
39:    int Write(unsigned char*, int);
40:};
41:
42:#define CS_PORT_COM1     0
43:#define CS_PORT_COM2     1
44:#define CS_PORT_LPT1     2
45:#define CS_PORT_LPT2     3
46:
47:#define CS_BOUD_1200     0
48:#define CS_BOUD_2400     1
49:#define CS_BOUD_4800     2
50:#define CS_BOUD_9600     3
51:#define CS_BOUD_19200    4
52:
53:#define CS_PARITY_NONE   0
54:#define CS_PARITY_ODD    1
55:#define CS_PARITY_EVEN   2
56:
57:#define CS_LENGTH_4      0
58:#define CS_LENGTH_5      1
59:#define CS_LENGTH_6      2
60:#define CS_LENGTH_7      3
61:#define CS_LENGTH_8      4
62:
63:#define CS_STOP_1        0
64:#define CS_STOP_1H       1
65:#define CS_STOP_2        2
66:
67:#define CS_NOW           255
68:
```

```
69:#define CS_QUE_SEND     0
70:#define CS_QUE_RECEIVE  1
71:#define CS_QUE_BOTH     2
72:
```

```
 1:// SOCKDLG.H : header file
 2://
 3:
 4://///////////////////////////////////////////////////////////////////////////
 5:// CSockDlg dialog
 6:
 7:class CSockDlg : public CDialog
 8:{
 9:// Construction
10:public:
11:    CSockDlg(CWnd* pParent = NULL); // standard constructor
12:
13:// Dialog Data
14:    //{{AFX_DATA(CSockDlg)
15:    enum { IDD = IDD_DIALOG1 };
16:    CButton m_Ok;
17:    CButton m_Cancel;
18:    CString m_Name;
19:    //}}AFX_DATA
20:
21:
22:// Overrides
23:public:
24:
25:protected:
26:    virtual void DoDataExchange(CDataExchange* pDX);    // DDX/DDV support
27:
28:// Implementation
29:protected:
30:
31:    // Generated message map functions
32:    //{{AFX_MSG(CSockDlg)
33:    //}}AFX_MSG
34:    DECLARE_MESSAGE_MAP()
35:};
36:
```

```
 1://///////////////////////////////////////////////////////////////////////////
 2:// Class CSock : CObject
 3:
 4:class CSocket : public CObject
 5:{
 6:public:
 7:    CSocket();
 8:    DECLARE_DYNCREATE( CSocket )
 9:
10:// ﾒﾝﾊﾞ変数
11:private:
12:    int     m_Socket;                   // ｿｹｯﾄﾃﾞｨｽｸﾘﾌﾟﾀ
13:    CString m_SrvName;                  // ｻｰﾊﾞ名
14:
15:// ﾒﾝﾊﾞ関数
16:public:
17:    int     Connect();                  // ｻｰﾊﾞとの接続
18:    void    SShutdown();                // ｼｬｯﾄﾀﾞｳﾝ
19:    void    SetSrvName(CString buf);    // ｻｰﾊﾞ名の変更
20:    CString GetSrvName();               // 現在のｻｰﾊﾞ名の取得
21:    void    SetSocket(int s);           // ｿｹｯﾄﾃﾞｨｽｸﾘﾌﾟﾀの変更
22:    int     GetSocket();                // 現在のｿｹｯﾄﾃﾞｨｽｸﾘﾌﾟﾀを取得
23:    void    Send(short Pkt_Tbl[]);      // 送信
24:    short   Receive(int* len);          // 受信
25:    int     StartInterrupt(HWND hWnd);  // ｳｨﾝﾄﾞｳｽﾞﾒｯｾｰｼﾞの発生
26:};
27:
28:#define WM_WINSOCK  WM_USER + 50        // ｻｰﾊﾞの送信要求時に発生するｳｨﾝﾄﾞｳｽﾞﾒｯｾｰｼﾞ
29:
```

```
1:// stdafx.h : include file for standard system include files,
2://  or project specific include files that are used frequently, but
3://      are changed infrequently
4://
5:
6:#include <afxwin.h>         // MFC core and standard components
7:#include <afxext.h>         // MFC extensions (including VB)
8:
```

```
 1:// trace.h : main header file for the TRACE application
 2://
 3:
 4:#ifndef __AFXWIN_H__
 5:    #error include 'stdafx.h' before including this file for PCH
 6:#endif
 7:
 8:#include "resource.h"       // main symbols
 9:
10://///////////////////////////////////////////////////////////////////////////
11:// CTraceApp:
12:// See trace.cpp for the implementation of this class
13://
14:
15:class CTraceApp : public CWinApp
16:{
17:public:
18:    CTraceApp();
19:
20:// Overrides
21:    virtual BOOL InitInstance();
22:
23:// Implementation
24:
25:    //{{AFX_MSG(CTraceApp)
26:    afx_msg void OnAppAbout();
27:        // NOTE - the ClassWizard will add and remove member functions here.
28:        //    DO NOT EDIT what you see in these blocks of generated code !
29:    //}}AFX_MSG
30:    DECLARE_MESSAGE_MAP()
31:};
32:
33:
34://///////////////////////////////////////////////////////////////////////////
35:
```

```
xperiment/PSfiles  5950  ./gakkai/Experiment  1517        ./gakkai/HC_kenkyu
kai_94/PSfiles  44       ./gakkai/HC_kenkyukai_94/DrawFiles  1740 ./gakkai/H
C_kenkyukai_94  72486    ./gakkai  431    ./homeWK  6      ./DataCom/DrawFile
s  540   ./DataCom  55    ./papers  10     ./D2_rinko/PSfiles  40   ./D2_rinko
/DrawFiles  196  ./D2_rinko  49   ./review  9062   ./job  13        ./D3_rinko
/PSfiles  5      ./D3_rinko/DrawFiles  2649       ./D3_rinko/ScannedImages  2
925     ./D3_rinko  44   ./References  1151       ./YobiShinsa/DrawFiles  981
        ./YobiShinsa/PSfiles  2397       ./YobiShinsa  126        ./hat_sens
ei  96093        .
```

```
 1:// tracedoc.h : interface of the CTraceDoc class
 2://
 3://///////////////////////////////////////////////////////////////////////////
 4:
 5:class CTraceDoc : public CDocument
 6:{
 7:protected: // create from serialization only
 8:    CTraceDoc();
 9:    DECLARE_DYNCREATE(CTraceDoc)
10:
11:// Attributes
12:public:
13:// Operations
14:public:
15:
16:// Implementation
17:public:
18:    virtual ~CTraceDoc();
19:    virtual void Serialize(CArchive& ar);   // overridden for document i/o
20:#ifdef _DEBUG
21:    virtual void AssertValid() const;
22:    virtual void Dump(CDumpContext& dc) const;
23:#endif
24:
25:protected:
26:    virtual BOOL OnNewDocument();
27:
28:// Generated message map functions
29:protected:
30:    //{{AFX_MSG(CTraceDoc)
31:        // NOTE - the ClassWizard will add and remove member functions here.
32:        //    DO NOT EDIT what you see in these blocks of generated code !
33:    //}}AFX_MSG
34:    DECLARE_MESSAGE_MAP()
35:};
36:
37://///////////////////////////////////////////////////////////////////////////
38:
```

```
 1:// tracevw.h : interface of the CTraceView class
 2://
 3://///////////////////////////////////////////////////////////////////////////
 4:
 5:class CSocket;
 6:class CCapture;
 7:class CGrabDlg;
 8:class CTraceView : public CView
 9:{
10:protected: // create from serialization only
11:     CTraceView();
12:     DECLARE_DYNCREATE(CTraceView)
13:
14:// Attributes
15:public:
16:     CTraceDoc* GetDocument();
17:
18:// Operations
19:public:
20:
21:// Member variables
22:private:
23:     BOOL m_FFlag;
24:     BOOL m_RFlag;
25:     BOOL m_oflag;
26:     BOOL m_CFlag;
27:     CClientDC* m_pDC;
28:     char *m_Mode[2];
29:
30:     // for Socket
31:     int m_Socket;
32:     CSocket* m_pSocket;
33:     CString m_SrvName;
34:     short m_SockBuf[256];
35:
36:     // for Capture
37:     int m_Counter;
38:     CGrabDlg* m_pDlg;
39:     CCapture* m_pCapture;
40:     int m_Condition;
41:     int m_s1;
42:     int m_s2;
43:
44:     short *xyp;
45:     Mark dmy;
46:     Area *maskArea;
47:     Mark *grnMarker;
48:     Mark *redMarker;
49:     Mark *marker;
50:     Mark *orgMark;
51:     Mark *orgVec;
52:
53:// Member functions
54:public:
55:     // Create error message box
56:     void PutErrorMsg(int err, int Param1, CString Param2);
57:     // Abnormal termination
58:     void Terminate(int x);
59:
60:// Implementation
61:public:
62:     virtual ~CTraceView();
63:     virtual void OnDraw(CDC* pDC);  // overridden to draw this view
64:#ifdef _DEBUG
65:     virtual void AssertValid() const;
66:     virtual void Dump(CDumpContext& dc) const;
67:#endif
68:
```

```
69:protected:
70:
71:// Generated message map functions
72:protected:
73:     //{{AFX_MSG(CTraceView)
74:     afx_msg void OnMenustart();
75:     afx_msg void OnEnd();
76:     afx_msg void OnTimer(UINT nIDEvent);
77:     afx_msg LONG OnWinsock(UINT, LONG);
78:     afx_msg LONG OnDlgMsg(UINT, LONG);
79:     afx_msg void OnRetry();
80:     afx_msg void OnRButtonDown(UINT nFlags, CPoint point);
81:     afx_msg void OnSetSock();
82:     //}}AFX_MSG
83:     DECLARE_MESSAGE_MAP()
84:};
85:
86:#ifndef _DEBUG  // debug version in tracevw.cpp
87:inline CTraceDoc* CTraceView::GetDocument()
88:    { return (CTraceDoc*)m_pDocument; }
89:#endif
90:
91://///////////////////////////////////////////////////////////////////////////
92:
93:// Definition of error codes
94:#define ERR_CONFIG  0
95:#define ERR_SOCKET  1
96:#define ERR_MARKER  2
97:#define ERR_MEMORY  3
98:
99:
```

```
 1:// VINDLG.H : header file
 2://
 3:
 4://///////////////////////////////////////////////////////////////////////////
 5:// CVinDlg dialog
 6:
 7:class CVinDlg : public CDialog
 8:{
 9:// Construction
10:public:
11:    CVinDlg(CWnd* pParent = NULL);  // standard constructor
12:
13:// Dialog Data
14:    //{{AFX_DATA(CVinDlg)
15:    enum { IDD = IDD_DIALOG2 };
16:        // NOTE: the ClassWizard will add data members here
17:    //}}AFX_DATA
18:
19:
20:// Overrides
21:public:
22:
23:protected:
24:    virtual void DoDataExchange(CDataExchange* pDX);    // DDX/DDV support
25:
26:// Implementation
27:protected:
28:
29:    // Generated message map functions
30:    //{{AFX_MSG(CVinDlg)
31:        // NOTE: the ClassWizard will add member functions here
32:    //}}AFX_MSG
33:    DECLARE_MESSAGE_MAP()
34:};
35:
```

```
1:#include "cmtlib.h"
2://#include "intrn¥cmtlib.h"
3:
4:int GrabFlag;
5:
6:void interrupt GrabInt(void)
7:{
8:    GrabFlag = TRUE;
9:}
```

```
 1:#include <stdlib.h>
 2:#include <malloc.h>
 3:
 4:#include "mtrace.h"
 5:
 6:typedef struct {
 7:     int id;
 8:     int length;
 9:     int x;
10:     int y;
11:} MTemp;
12:
13:int compMTempLength(const void *v1, const void *v2)
14:{
15:     MTemp *t1, *t2;
16:
17:     t1 = (MTemp *)v1;
18:     t2 = (MTemp *)v2;
19:
20:     return (t1->length - t2->length);
21:}
22:
23:int compMTempY(const void *v1, const void *v2)
24:{
25:     MTemp *t1, *t2;
26:
27:     t1 = (MTemp *)v1;
28:     t2 = (MTemp *)v2;
29:
30:     return (t1->y - t2->y);
31:}
32:
33:int compMTempX(const void *v1, const void *v2)
34:{
35:     MTemp *t1, *t2;
36:
37:     t1 = (MTemp *)v1;
38:     t2 = (MTemp *)v2;
39:
40:     return (t2->x - t1->x);
41:}
42:
43:void lineUPMarker(MarkConf *conf, Mark grn[], Mark red[], Mark marker[], int *mn)
44:{
45:     int i, n;
46:     MTemp *index;
47://   MTemp dind[20];
48:
49:     index = calloc(conf->nOfGreen, sizeof(MTemp));
50:     n = conf->nOfGreen;
51:     // 顔中央の線上のﾏｰｶを抽出
52:     for (i = 0; i < conf->nOfGreen; i++) {
53:         index[i].id = i;
54:         index[i].length = abs(grn[i].y - GRAB_WIN_Y/2);
55:         index[i].x = grn[i].x;
56:         index[i].y = grn[i].y;
57:     }
58:     // y座標が中央値に近いﾏｰｶをconf->nOfCenter個抽出
59:     qsort(index, n, sizeof(MTemp), compMTempLength);
60:     // x座標でｿｰﾄして右から順番に整列
61:     qsort(index, conf->nOfCenter, sizeof(MTemp), compMTempX);
62:     for (i=0; i<conf->nOfCenter; i++) {
63:         marker[ conf->center[i].id - 1] = grn[index[i].id];
64:     }
65:     index += conf->nOfCenter;
66:     n -= conf->nOfCenter;
67:     // y座標が小さいﾏｰｶをconf->nOfBottom個抽出
68:     qsort(index, n, sizeof(MTemp), compMTempY);
```

```
69:     // x座標でソートして右から順番に整列
70:     qsort(index, conf->nOfBottom, sizeof(MTemp), compMTempX);
71:     for (i = 0; i < conf->nOfBottom; i++) {
72:         marker[conf->bottom[i].id - 1] = grn[index[i].id];
73:     }
74:     index += conf->nOfBottom;
75:     n -= conf->nOfBottom;
76:     // x座標でソートして右から順番に整列
77:     qsort(index, conf->nOfBottom, sizeof(MTemp), compMTempX);
78:     for (i = 0; i < conf->nOfUpper; i++) {
79:         marker[conf->upper[i].id - 1] = grn[index[i].id];
80:     }
81:     *mn = conf->nOfGreen;
82://   free(index);
83:}
```

```
 1:// capture.cpp : implementation of the CCapture class
 2://
 3:#include "stdafx.h"
 4:#include "mtrace.h"
 5:#include "capture.h"
 6:
 7:#include "cmtlib.h"
 8:#include "grabint.h"
 9:
10:#include "math.h"
11:
12://///////////////////////////////////////////////////////////////////////////
13:// CCapture
14:
15:IMPLEMENT_DYNCREATE(CCapture, CObject)
16:
17://///////////////////////////////////////////////////////////////////////////
18:// CCapture construction
19:
20:CCapture::CCapture()
21:{
22:        m_pView = NULL;
23:        m_Counter = 0;
24:        m_Chan = 0;
25:}
26:
27:CCapture::CCapture(CView* pView)
28:{
29:        m_pView = pView;
30:}
31:
32://///////////////////////////////////////////////////////////////////////////
33:// CCapture member functions
34:
35:// ｷｬﾌﾟﾁｬﾎﾞｰﾄﾞ関連
36:
37:// 初期化、ｷｬﾌﾟﾁｬﾘﾝｸﾞ開始、grab割込み開始
38:int CCapture::CaptureInit()
39:{
40:        VG_WORD lwError;
41:
42:        lwError = CmtGenLibInit();                                      // Cometﾗｲﾌﾞﾗﾘ初期化
43:        if (lwError < 0) return FALSE;
44:        CmtGenBoardReset();                                             // ﾎﾞｰﾄﾞﾘｾｯﾄ
45:        CmtVinInSelect(CMT_VIN_CHAN_A);                                 // ﾁｬﾝﾈﾙAからの入力
46:        CmtVinInSetStandard(CMT_VIN_NTSC);                              // NTSC形式
47:        CmtVipGrabSetMode(CMT_VIP_GRAB_555);                            // 15ﾋﾞｯﾄﾓｰﾄﾞ
48:        CmtVipGrabSetSourcePos( 0, 0, 640 ,480);                        // 640x480の範囲を入力
49:        CmtVipGrabSetDestPos(0, 0, GRAB_WIN_X, GRAB_WIN_Y);     // 出力位置、範囲
50:        CmtVipGrabSetRate(CMT_VIP_GRAB_CONTINUOUS);                     // ｷｬﾌﾟﾁｬﾘﾝｸﾞ開始
51:
52:        CmtVipGrabRegisterIntFunction(&GrabInt);                        // ｲﾝﾀﾗﾌﾟﾄ関数のｱﾄﾞﾚｽを通知
53:        CmtVipGrabStartInterrupts();                                    // grab割込み開始
54:        return TRUE;
55:}
56:
57:void CCapture::CaptureQuit()
58:{
59:        CmtVipGrabStopInterrupts();                                     // grab割込み停止
60:        CmtVipGrabRegisterIntFunction(NULL);                            // ｲﾝﾀﾗﾌﾟﾄ関数のｱﾄﾞﾚｽをNULLに
61:        CmtVipGrabSetRate(CMT_VIP_GRAB_OFF);                            // ｷｬﾌﾟﾁｬﾘﾝｸﾞ停止
62:        CmtVipDisplaySetSt(0);
63:        CmtGenLibQuit();                                                // ﾗｲﾌﾞﾗﾘ終了
64:        FreeMalloc(2);                                                  // ﾒﾓﾘ解放
65:}
66:
67:// 指定された座標をﾌﾚｰﾑﾊﾞｯﾌｧ上のｱﾄﾞﾚｽに変換
68:
```

```
 69:long CCapture::Addr(long x, long y)
 70:{
 71:        return ((x + y * 800)*2);
 72:}
 73:
 74:// 任意座標のﾋﾟｸｾﾙﾃﾞｰﾀを読み込む
 75:
 76:short CCapture::Bufpix(int x, int y)
 77:{
 78:        short buf[2];
 79:        CmtGenBlockRead(Addr(x, y), 1, 1, (unsigned char*)buf, sizeof(short)*2);
 80://      buf[0] = Frame[x + y * GRAB_WIN_Y];
 81:        buf[0] &= 0x7fff;
 82:        return buf[0];
 83:}
 84:
 85:// 任意座標にﾋﾟｸｾﾙﾃﾞｰﾀを書き込む
 86:
 87:void CCapture::Putpix(int x, int y, short src)
 88:{
 89:        short buf[2];
 90:        buf[0] = src;
 91:        CmtGenBlockWrite(Addr(x, y), 1, 1, (unsigned char*)buf, 2);
 92:}
 93:
 94:// ｷｬﾌﾟﾁｬﾘﾝｸﾞ停止
 95:
 96:void CCapture::CaptureStop()
 97:{
 98:        CmtVipGrabSetRate(CMT_VIP_GRAB_OFF);
 99:}
100:
101:// ｷｬﾌﾟﾁｬﾘﾝｸﾞ開始
102:
103:void CCapture::CaptureStart()
104:{
105:        CmtVipGrabSetRate(CMT_VIP_GRAB_CONTINUOUS);
106:}
107:
108:void CCapture::ChangeChannel()
109:{
110:        if (m_Chan == 0)
111:                CmtVinInSelect(CMT_VIN_CHAN_A);
112:        else
113:                CmtVinInSelect(CMT_VIN_CHAN_B);
114:        m_Chan = 1 - m_Chan;
115:}
116:
117:// ﾏｰｶ追跡関連
118:
119:// ﾒﾓﾘ解放
120:
121:void CCapture::FreeMalloc(int x)
122:{
123:        switch(x) {
124:                case 2:
125:                        delete m_Conf.bottom;
126:                case 1:
127:                        delete m_Conf.upper;
128:                case 0:
129:                        delete m_Conf.center;
130:                default:
131:                break;
132:        }
133:}
134:
135:// Configﾌｧｲﾙの読み込み
136:
```

```
137:int CCapture::ReadConfig(char *name)
138:{
139:        FILE *fp;
140:        char buff[128];
141:        char key[128];
142:        int cnt;
143:        int lines = 0;
144:
145:        m_Conf.grnScan.width = MARK_SCAN_WIDTH;
146:        m_Conf.grnScan.height = MARK_SCAN_HEIGHT;
147:        m_Conf.redScan.width = RED_MARK_SCAN_WIDTH;
148:        m_Conf.redScan.height = RED_MARK_SCAN_HEIGHT;
149:        m_Conf.nOfGreen = 0;
150:        m_Conf.nOfCenter = 0;
151:        m_Conf.nOfUpper = 0;
152:        m_Conf.nOfBottom = 0;
153:        m_Conf.eye.id = -1;
154:        m_Conf.eyeMove.width = EYE_AREA_MOVE_X;
155:        m_Conf.eyeMove.height = EYE_AREA_MOVE_Y;
156:
157:        if ((fp = fopen(name, "r")) == NULL) {
158:                return -1;
159:        }
160:        while (fgets(buff, 128, fp) != NULL) {
161:                lines++;
162:                if (*buff == '#') continue;
163:
164:                sscanf(buff, "%s", key);
165:                if (strcmp(key, "number") == 0) {
166:                        if (sscanf(buff, "%s %d %d", key, &m_Conf.nOfGreen, &m_Conf.nOfRed) != 3) {
167:                                fclose(fp);
168:                                return lines;
169:                        }
170:                } else
171:                if (strcmp(key, "scansize") == 0) {
172:                        if (sscanf(buff, "%s %d %d %d %d",
173:                                key,
174:                                &m_Conf.grnScan.width,
175:                                &m_Conf.grnScan.height,
176:                                &m_Conf.redScan.width,
177:                                &m_Conf.redScan.height ) != 5) { fclose(fp); return lines; }
178:                } else
179:                if (strcmp(key, "eye") == 0) {
180:                        if (sscanf(buff, "%s %d %d %d %d %d %d",
181:                                key,
182:                                &m_Conf.eye.id,
183:                                &m_Conf.eyeOpenID,
184:                                &m_Conf.eye.area.xl,
185:                                &m_Conf.eye.area.xr,
186:                                &m_Conf.eye.area.yb,
187:                                &m_Conf.eye.area.yt ) != 7) {fclose(fp); return lines;}
188:                        m_Conf.eye.area.attr = -1;
189:                        m_Conf.nOfRed = 1;
190:                } else
191:                if (strcmp(key, "eyeMove") == 0)  {
192:                        if (sscanf(buff, "%s %d %d",
193:                                key,
194:                                &m_Conf.eyeMove.width,
195:                                &m_Conf.eyeMove.height) != 3) {fclose(fp); return lines;}
196:                } else
197:                if (strcmp(key, "center") == 0) {
198:                        if (sscanf(buff, "%s %d", key, &cnt) != 2) {fclose(fp); return lines;}
199:                        if (readCenterLineMarkerConf(fp, cnt, &lines) != 0) {
200:                                FreeMalloc(0);
201:                                fclose(fp);
202:                                return lines;
203:                        }
204:                } else
```

```
205:                if (strcmp(key, "upper") == 0) {
206:                        if (sscanf(buff, "%s %d", key, &cnt) != 2) {
207:                                FreeMalloc(0);
208:                                fclose(fp);
209:                                return lines;
210:                        }
211:                        if (readBottomLineMarkerConf(fp, cnt, &lines) != 0) {
212:                                FreeMalloc(1);
213:                                fclose(fp);
214:                                return lines;
215:                        }
216:                } else
217:                if (strcmp(key, "bottom") == 0) {
218:                        if (sscanf(buff, "%s %d", key, &cnt) != 2) {
219:                                FreeMalloc(1);
220:                                fclose(fp);
221:                                return lines;
222:                        }
223:                        if (readUpperLineMarkerConf(fp, cnt, &lines)) {
224:                                FreeMalloc(2);
225:                                fclose(fp);
226:                                return lines;
227:                        }
228:                } else {fclose(fp); return lines;}
229:        }
230:        fclose(fp);
231:        return 0;
232:}
233:
234:int CCapture::readCenterLineMarkerConf(FILE* fp, int numb, int *lines)
235:{
236:        char buff[128];
237:        int result = -1;
238:        int n;
239:
240:        if (m_Conf.nOfCenter != 0 || numb == 0) return result;
241:        m_Conf.nOfCenter = numb;
242:        m_Conf.center = new MarkID[numb];
243:        n = 0;
244:        while (fgets(buff, 128, fp) != NULL && result != 0) {
245:                (*lines)++;
246:                if (*buff == '#') continue;
247:                if (sscanf(buff, "%d %d %d %d %d %d",
248:                        &m_Conf.center[n].id,
249:                        &m_Conf.center[n].area.xl,
250:                        &m_Conf.center[n].area.xr,
251:                        &m_Conf.center[n].area.yb,
252:                        &m_Conf.center[n].area.yt,
253:                        &m_Conf.center[n].area.attr) != 6) return result;
254:                n++;
255:                if (n >= numb) {
256:                        result = 0;
257:                }
258:        }
259:        return result;
260:}
261:
262:int CCapture::readUpperLineMarkerConf(FILE *fp, int numb, int *lines)
263:{
264:        char buff[128];
265:        int result = -1;
266:        int n;
267:
268:        if (m_Conf.nOfUpper != 0 || numb == 0) return result;
269:        m_Conf.nOfUpper = numb;
270:        m_Conf.upper = new MarkID[numb];
271:        n = 0;
272:        while (fgets(buff, 128, fp) != NULL && result != 0) {
```

```
273:                (*lines)++;
274:                if (*buff == '#') continue;
275:                if (sscanf(buff, "%d %d %d %d %d %d",
276:                        &m_Conf.upper[n].id,
277:                        &m_Conf.upper[n].area.xl,
278:                        &m_Conf.upper[n].area.xr,
279:                        &m_Conf.upper[n].area.yb,
280:                        &m_Conf.upper[n].area.yt,
281:                        &m_Conf.upper[n].area.attr) != 6) return result;
282:                n++;
283:                if (n >= numb) {
284:                        result = 0;
285:                }
286:        }
287:        return result;
288:}
289:
290:int CCapture::readBottomLineMarkerConf(FILE *fp, int numb, int *lines)
291:{
292:        char buff[128];
293:        int result = -1;
294:        int n;
295:
296:        if (m_Conf.nOfBottom != 0 || numb == 0) return result;
297:        m_Conf.nOfBottom = numb;
298:        m_Conf.bottom = new MarkID[numb];
299:        n = 0;
300:        while (fgets(buff, 128, fp) != NULL && result != 0) {
301:                (*lines)++;
302:                if (*buff == '#') continue;
303:                if (sscanf(buff, "%d %d %d %d %d %d",
304:                        &m_Conf.bottom[n].id,
305:                        &m_Conf.bottom[n].area.xl,
306:                        &m_Conf.bottom[n].area.xr,
307:                        &m_Conf.bottom[n].area.yb,
308:                        &m_Conf.bottom[n].area.yt,
309:                        &m_Conf.bottom[n].area.attr) != 6) return result;
310:                n++;
311:                if (n >= numb) {
312:                        result = 0;
313:                }
314:        }
315:        return result;
316:}
317:
318:// 探索範囲のﾄﾞｯﾄ群をﾗﾍﾞﾙとして、その重心と面積を返す
319:
320:void CCapture::ScanMarker(int sx, int sy, int *rx, int *ry, long *rsize, int sW, int sH)
321:{
322:        long m00, m01, m10;
323:        int x, y;
324:        int xt, yt;
325:
326:        m00 = m01 = m10 = 0;
327:        for (x = 0; x < sW; x++) {
328:                xt = sx - x;
329:                if (xt > GRAB_WIN_X) break;
330:                for (y = -sH/2; y <= sH/2; y++) {
331:                        yt = sy + y;
332:                        if (yt >= 0 && yt < GRAB_WIN_Y) {
333:                                if (Bufpix(xt, yt) > MARK_LEVEL) {
334:                                        m01 -= x;
335:                                        m10 += y;
336:                                        m00++;
337:                                }
338:                                Putpix(xt, yt, 0x0000);
339:                        }
340:                }
```

```
341:        }
342:        *rx = (int) (m01/m00 + sx);
343:        *ry = (int) (m10/m00 + sy);
344:        *rsize = m00;
345:}
346:
347:// 画面上のﾏｰｶの座標を返す（順不同）
348:// redMarksは視線追跡拡張時のためのﾀﾞﾐｰﾊﾟﾗﾒｰﾀ
349:
350:int CCapture::ScanMarkerPosition(Mark marks[], Mark redMarks[])
351:{
352:        int x, y;
353:        int gx, gy;
354:        long gsize;
355:        int result = -1;
356:        int cnt, redCnt;
357:
358:        CmtVipGrabSetRate(CMT_VIP_GRAB_OFF);
359:
360:        cnt = 0;
361:        redCnt = 0;
362:
363:        for (x = GRAB_WIN_X - RIGHT_MARGIN; x >= LEFT_MARGIN; x--) {
364:                for (y = TOP_MARGIN; y < GRAB_WIN_Y - BOTTOM_MARGIN; y++) {
365:                        if (Bufpix(x, y) > MARK_LEVEL && cnt < m_Conf.nOfGreen) {
366:                                //Putpix(x, y, 0x7c00);
367:                                ScanMarker(x, y, &gx, &gy, &gsize, m_Conf.grnScan.width, m_Conf.grnScan.height);
368:                                if ((gsize >= REGION_RATIO * m_Conf.grnScan.width * m_Conf.grnScan.height) &&
369:                                        (gsize <= 0.80 * m_Conf.grnScan.width * m_Conf.grnScan.height)) {
370:                                        marks[cnt].x = gx;
371:                                        marks[cnt].y = gy;
372:                                        marks[cnt].area = gsize;
373:                                        cnt++;
374:                                }
375:                        }
376:                }
377:        }
378:
379:        CmtVipGrabSetRate(CMT_VIP_GRAB_CONTINUOUS);
380:        return cnt;
381:}
382:
383:// 指定のﾏｰｶの移動先を調べる
384:
385:void CCapture::ScanMarkMove(Area area, Mark orgPos, Mark * pos)
386:{
387:        int xs, ys;
388:        int x, y;
389:        long m00, m01, m10;
390:        int orgx, orgy;
391:        short tmp;
392:        int tmp_x, tmp_y;
393:
394:        xs = area.xr - area.xl + 1;
395:        ys = area.yt - area.yb + 1;
396:
397:        tmp_x = pos->x;
398:        tmp_y = pos->y;
399:        if (area.attr != 1) {
400:                orgx = pos->x;
401:                orgy = pos->y;
402:        } else {
403:                orgx = orgPos.x;
404:                orgy = orgPos.y;
405:        }
406:        CmtGenBlockRead(Addr(orgx+area.xl, orgy+area.yb), xs, ys, (unsigned char*)Buffer, sizeof(short)*xs*ys);
407:
408:        m00 = m01 = m10 = 0;
```

```
409:        if (area.attr >= 0) {
410:                for (y = 0; y < ys; y++) {
411:                        for (x = 0; x < xs; x++) {
412:                                tmp = Buffer[x + y * xs] & 0x7fff;
413:                                if (tmp > MARK_LEVEL) {
414:                                        m01 += x;
415:                                        m10 += y;
416:                                        m00++;
417:                                }
418:                        }
419:                }
420:        }
421:        pos->area = m00;
422:        if (m00 > 0) {
423:                pos->x = (int)((m01/m00) + (orgx + area.xl));
424://              if (abs(pos->x - tmp_x) < 3) pos->x = tmp_x;
425:                pos->y = (int)((m10/m00) + (orgy + area.yb));
426://              if (abs(pos->y - tmp_y) < 3) pos->y = tmp_y;
427:        }
428:}
429:
430:int CCapture::SearchNearlyMarker(int px, int py, Mark *mark, int mn)
431:{
432:        int i, t = -1;
433:        double xl, yl, l, len = HUGE;
434:
435:        for (i=0;i < mn; i++) {
436:                xl = px - mark[i].x;
437:                yl = py - mark[i].y;
438:                l = sqrt(xl*xl + yl*yl);
439:                if (len>l) {
440:                        len = l;
441:                        t = i;
442:                }
443:        }
444:
445:        return t;
446:}
447:
448:
```

```
 1:// capture.cpp : implementation of the CCapture class
 2://
 3:#include "stdafx.h"
 4:#include "mtrace.h"
 5:#include "capture.h"
 6:
 7:#include "cmtlib.h"
 8:#include "grabint.h"
 9:
10:#include "math.h"
11:
12://///////////////////////////////////////////////////////////////////////////
13:// CCapture
14:
15:IMPLEMENT_DYNCREATE(CCapture, CObject)
16:
17://///////////////////////////////////////////////////////////////////////////
18:// CCapture construction
19:
20:CCapture::CCapture()
21:{
22:        m_pView = NULL;
23:        m_Standard = CMT_VIN_PAL;
24:        m_Counter = 0;
25:}
26:
27:CCapture::CCapture(CView* pView)
28:{
29:        m_pView = pView;
30:}
31:
32://///////////////////////////////////////////////////////////////////////////
33:// CCapture member functions
34:
35:void CCapture::SetStandard(uchar st)
36:{
37:        if (st > 2) st = 2;
38:        if (st < 0) st = 0;
39:        m_Standard = st;
40:        CmtVinInSetStandard(st);
41:}
42:
43:uchar CCapture::GetStandard()
44:{
45:        return m_Standard;
46:}
47:
48:void CCapture::SetType(uchar ty)
49:{
50:        if (ty > 1) ty = 1;
51:        if (ty < 0) ty = 0;
52:        m_Type = ty;
53:        CmtVinInSetType(ty);
54:}
55:
56:uchar CCapture::GetType()
57:{
58:        return m_Type;
59:}
60:
61:void CCapture::SetBrightness(uchar br)
62:{
63:        if (br > 255) br = 255;
64:        if (br <   0) br = 0;
65:        m_Brightness = br;
66:        CmtVinLumSetBrightness(br);
67:}
68:
```

```
 69:uchar CCapture::GetBrightness()
 70:{
 71:        return m_Brightness;
 72:}
 73:
 74:int CCapture::CaptureInit()
 75:{
 76:        VG_WORD lwError;
 77:
 78:        lwError = CmtGenLibInit();
 79:        if (lwError < 0) return FALSE;
 80:        CmtGenBoardReset();
 81:        CmtVinInSelect(CMT_VIN_CHAN_A);
 82:        CmtVinInSetStandard(CMT_VIN_NTSC);
 83:        CmtVipGrabSetMode(CMT_VIP_GRAB_555);
 84:        CmtVipGrabSetSourcePos( 0, 0, 640 ,480);
 85:        CmtVipGrabSetDestPos(0, 0, GRAB_WIN_X, GRAB_WIN_Y);
 86:        CmtVipGrabSetRate(CMT_VIP_GRAB_CONTINUOUS);
 87:
 88:        CmtVipGrabRegisterIntFunction(&GrabInt);
 89:        CmtVipGrabStartInterrupts();
 90:        return TRUE;
 91:}
 92:
 93:void CCapture::CaptureQuit()
 94:{
 95:        CmtVipGrabStopInterrupts();
 96:        CmtVipGrabRegisterIntFunction(NULL);
 97:        CmtVipGrabSetRate(CMT_VIP_GRAB_OFF);
 98:        CmtVipDisplaySetSt(0);
 99:        CmtGenLibQuit();
100:        FreeMalloc(2);
101:}
102:
103:long CCapture::Addr(long x, long y)
104:{
105:        return ((x + y * 800)*2);
106:}
107:
108:int CCapture::ReadConfig(char *name)
109:{
110:        FILE *fp;
111:        char buff[128];
112:        char key[128];
113://      int sts;
114:        int cnt;
115:        int lines = 0;
116:
117:        m_Conf.grnScan.width = MARK_SCAN_WIDTH;
118:        m_Conf.grnScan.height = MARK_SCAN_HEIGHT;
119:        m_Conf.redScan.width = RED_MARK_SCAN_WIDTH;
120:        m_Conf.redScan.height = RED_MARK_SCAN_HEIGHT;
121:        m_Conf.nOfGreen = 0;
122:        m_Conf.nOfCenter = 0;
123:        m_Conf.nOfUpper = 0;
124:        m_Conf.nOfBottom = 0;
125:        m_Conf.eye.id = -1;
126:        m_Conf.eyeMove.width = EYE_AREA_MOVE_X;
127:        m_Conf.eyeMove.height = EYE_AREA_MOVE_Y;
128:
129:        if ((fp = fopen(name, "r")) == NULL) {
130:                return -1;
131:        }
132:        while (fgets(buff, 128, fp) != NULL) {
133:                lines++;
134:                if (*buff == '#') continue;
135:
136:                sscanf(buff, "%s", key);
```

```
137:                if (strcmp(key, "number") == 0) {
138:                        if (sscanf(buff, "%s %d %d", key, &m_Conf.nOfGreen, &m_Conf.nOfRed) != 3) {
139:                                fclose(fp);
140:                                return lines;
141:                        }
142:                } else
143:                if (strcmp(key, "scansize") == 0) {
144:                        if (sscanf(buff, "%s %d %d %d %d",
145:                                key,
146:                                &m_Conf.grnScan.width,
147:                                &m_Conf.grnScan.height,
148:                                &m_Conf.redScan.width,
149:                                &m_Conf.redScan.height ) != 5) { fclose(fp); return lines; }
150:                } else
151:                if (strcmp(key, "eye") == 0) {
152:                        if (sscanf(buff, "%s %d %d %d %d %d %d",
153:                                key,
154:                                &m_Conf.eye.id,
155:                                &m_Conf.eyeOpenID,
156:                                &m_Conf.eye.area.xl,
157:                                &m_Conf.eye.area.xr,
158:                                &m_Conf.eye.area.yb,
159:                                &m_Conf.eye.area.yt ) != 7) {fclose(fp); return lines;}
160:                        m_Conf.eye.area.attr = -1;
161:                        m_Conf.nOfRed = 1;
162:                } else
163:                if (strcmp(key, "eyeMove") == 0)  {
164:                        if (sscanf(buff, "%s %d %d",
165:                                key,
166:                                &m_Conf.eyeMove.width,
167:                                &m_Conf.eyeMove.height) != 3) {fclose(fp); return lines;}
168:                } else
169:                if (strcmp(key, "center") == 0) {
170:                        if (sscanf(buff, "%s %d", key, &cnt) != 2) {fclose(fp); return lines;}
171:                        if (readCenterLineMarkerConf(fp, cnt, &lines) != 0) {
172:                                FreeMalloc(0);
173:                                fclose(fp);
174:                                return lines;
175:                        }
176:                } else
177:                if (strcmp(key, "upper") == 0) {
178:                        if (sscanf(buff, "%s %d", key, &cnt) != 2) {
179:                                FreeMalloc(0);
180:                                fclose(fp);
181:                                return lines;
182:                        }
183:                        if (readBottomLineMarkerConf(fp, cnt, &lines) != 0) {
184:                                FreeMalloc(1);
185:                                fclose(fp);
186:                                return lines;
187:                        }
188:                } else
189:                if (strcmp(key, "bottom") == 0) {
190:                        if (sscanf(buff, "%s %d", key, &cnt) != 2) {
191:                                FreeMalloc(1);
192:                                fclose(fp);
193:                                return lines;
194:                        }
195:                        if (readUpperLineMarkerConf(fp, cnt, &lines)) {
196:                                FreeMalloc(2);
197:                                fclose(fp);
198:                                return lines;
199:                        }
200:                } else {fclose(fp); return lines;}
201:        }
202:        fclose(fp);
203:        return 0;
204:}
```

```
205:
206:int CCapture::readCenterLineMarkerConf(FILE* fp, int numb, int *lines)
207:{
208:        char buff[128];
209:        int result = -1;
210:        int n;
211:
212:        if (m_Conf.nOfCenter != 0 || numb == 0) return result;
213:        m_Conf.nOfCenter = numb;
214://      m_Conf.center = (MarkID *) malloc(sizeof(MarkID) * numb);
215:        m_Conf.center = new MarkID[numb];
216:        n = 0;
217:        while (fgets(buff, 128, fp) != NULL && result != 0) {
218:                (*lines)++;
219:                if (*buff == '#') continue;
220:                if (sscanf(buff, "%d %d %d %d %d %d",
221:                        &m_Conf.center[n].id,
222:                        &m_Conf.center[n].area.xl,
223:                        &m_Conf.center[n].area.xr,
224:                        &m_Conf.center[n].area.yb,
225:                        &m_Conf.center[n].area.yt,
226:                        &m_Conf.center[n].area.attr) != 6) return result;
227:                n++;
228:                if (n >= numb) {
229:                        result = 0;
230:                }
231:        }
232:        return result;
233:}
234:
235:int CCapture::readUpperLineMarkerConf(FILE *fp, int numb, int *lines)
236:{
237:        char buff[128];
238:        int result = -1;
239:        int n;
240:
241:        if (m_Conf.nOfUpper != 0 || numb == 0) return result;
242:        m_Conf.nOfUpper = numb;
243://      m_Conf.upper = (MarkID *)malloc(sizeof(MarkID) * numb);
244:        m_Conf.upper = new MarkID[numb];
245:        n = 0;
246:        while (fgets(buff, 128, fp) != NULL && result != 0) {
247:                (*lines)++;
248:                if (*buff == '#') continue;
249:                if (sscanf(buff, "%d %d %d %d %d %d",
250:                        &m_Conf.upper[n].id,
251:                        &m_Conf.upper[n].area.xl,
252:                        &m_Conf.upper[n].area.xr,
253:                        &m_Conf.upper[n].area.yb,
254:                        &m_Conf.upper[n].area.yt,
255:                        &m_Conf.upper[n].area.attr) != 6) return result;
256:                n++;
257:                if (n >= numb) {
258:                        result = 0;
259:                }
260:        }
261:        return result;
262:}
263:
264:int CCapture::readBottomLineMarkerConf(FILE *fp, int numb, int *lines)
265:{
266:        char buff[128];
267:        int result = -1;
268:        int n;
269:
270:        if (m_Conf.nOfBottom != 0 || numb == 0) return result;
271:        m_Conf.nOfBottom = numb;
272://      m_Conf.bottom = (MarkID *) malloc(sizeof(MarkID) * numb);
```

```
273:        m_Conf.bottom = new MarkID[numb];
274:        n = 0;
275:        while (fgets(buff, 128, fp) != NULL && result != 0) {
276:                (*lines)++;
277:                if (*buff == '#') continue;
278:                if (sscanf(buff, "%d %d %d %d %d %d",
279:                        &m_Conf.bottom[n].id,
280:                        &m_Conf.bottom[n].area.xl,
281:                        &m_Conf.bottom[n].area.xr,
282:                        &m_Conf.bottom[n].area.yb,
283:                        &m_Conf.bottom[n].area.yt,
284:                        &m_Conf.bottom[n].area.attr) != 6) return result;
285:                n++;
286:                if (n >= numb) {
287:                        result = 0;
288:                }
289:        }
290:        return result;
291:}
292:
293:void CCapture::FreeMalloc(int x)
294:{
295:        switch(x) {
296:                case 2:
297://                      free(m_Conf.bottom);
298:                        delete m_Conf.bottom;
299:                case 1:
300://                      free(m_Conf.upper);
301:                        delete m_Conf.upper;
302:                case 0:
303://                      free(m_Conf.center);
304:                        delete m_Conf.center;
305:                default:
306:                break;
307:        }
308:}
309:
310:// 探索範囲の緑ﾌﾟﾚｰﾝがONのﾄﾞｯﾄ群をﾗﾍﾞﾙとして、その重心と面積を返す
311:
312:void CCapture::ScanMarker(int sx, int sy, int *rx, int *ry, long *rsize, int sW, int sH)
313:{
314:        long m00, m01, m10;
315:        int x, y;
316:        int xt, yt;
317:
318:        m00 = m01 = m10 = 0;
319:        for (x = 0; x < sW; x++) {
320:                xt = sx - x;
321:                if (xt > GRAB_WIN_X) break;
322:                for (y = -sH/2; y <= sH/2; y++) {
323:                        yt = sy + y;
324:                        if (yt >= 0 && yt < GRAB_WIN_Y) {
325:                                if (Bufpix(xt, yt) > MARK_LEVEL) {
326:                                        m01 -= x;
327:                                        m10 += y;
328:                                        m00++;
329:                                }
330:                                Putpix(xt, yt, 0x0000);
331:                        }
332:                }
333:        }
334:        *rx = (int) (m01/m00 + sx);
335:        *ry = (int) (m10/m00 + sy);
336:        *rsize = m00;
337:}
338:
339:// 画面上のﾏｰｶの座標を返す（順不同）
340:// redMarksは視線追跡拡張時のためのﾀﾞﾐｰﾊﾟﾗﾒｰﾀ
```

```
341:
342:int CCapture::ScanMarkerPosition(Mark marks[], Mark redMarks[])
343:{
344:        int x, y;
345:        int gx, gy;
346:        long gsize;
347:        int result = -1;
348:        int cnt, redCnt;
349:
350:        CmtVipGrabSetRate(CMT_VIP_GRAB_OFF);
351://      GetRect();
352://      CmtVipGrabSingle();
353:        cnt = 0;
354:        redCnt = 0;
355:
356:        for (x = GRAB_WIN_X - RIGHT_MARGIN; x >= LEFT_MARGIN; x--) {
357:                for (y = TOP_MARGIN; y < GRAB_WIN_Y - BOTTOM_MARGIN; y++) {
358:                        if (Bufpix(x, y) > MARK_LEVEL && cnt < m_Conf.nOfGreen) {
359:                                //Putpix(x, y, 0x7c00);
360:                                ScanMarker(x, y, &gx, &gy, &gsize, m_Conf.grnScan.width, m_Conf.grnScan.height);
361:                                if ((gsize >= REGION_RATIO * m_Conf.grnScan.width * m_Conf.grnScan.height) &&
362:                                        (gsize <= 0.80 * m_Conf.grnScan.width * m_Conf.grnScan.height)) {
363:                                        marks[cnt].x = gx;
364:                                        marks[cnt].y = gy;
365:                                        marks[cnt].area = gsize;
366:                                        cnt++;
367:                                }
368:                        }
369:                }
370:        }
371://      SetRect();
372://      if (cnt == m_Conf.nOfGreen) result = 0;
373://      return result;
374:        CmtVipGrabSetRate(CMT_VIP_GRAB_CONTINUOUS);
375:        return cnt;
376:}
377:
378:// 指定のﾏｰｶの移動先を調べる
379:
380:void CCapture::ScanMarkMove(Area area, Mark orgPos, Mark * pos)
381:{
382:        int xs, ys;
383:        int x, y;
384:        long m00, m01, m10;
385:        int orgx, orgy;
386:        short tmp;
387:        int tmp_x, tmp_y;
388:
389:        xs = area.xr - area.xl + 1;
390:        ys = area.yt - area.yb + 1;
391:
392:        tmp_x = pos->x;
393:        tmp_y = pos->y;
394:        if (area.attr != 1) {
395:                orgx = pos->x;
396:                orgy = pos->y;
397:        } else {
398:                orgx = orgPos.x;
399:                orgy = orgPos.y;
400:        }
401:        CmtGenBlockRead(Addr(orgx+area.xl, orgy+area.yb), xs, ys, (unsigned char*)Buffer, sizeof(short)*xs*ys);
402:
403:        m00 = m01 = m10 = 0;
404:        if (area.attr >= 0) {
405:                for (y = 0; y < ys; y++) {
406:                        for (x = 0; x < xs; x++) {
407:                                tmp = Buffer[x + y * xs] & 0x7fff;
408:                                if (tmp > MARK_LEVEL) {
```

```
409:                                    m01 += x;
410:                                    m10 += y;
411:                                    m00++;
412:                            }
413:                    }
414:            }
415:    }
416:    pos->area = m00;
417:    if (m00 > 0) {
418:            pos->x = (int)((m01/m00) + (orgx + area.xl));
419://          if (abs(pos->x - tmp_x) < 3) pos->x = tmp_x;
420:            pos->y = (int)((m10/m00) + (orgy + area.yb));
421://          if (abs(pos->y - tmp_y) < 3) pos->y = tmp_y;
422:    }
423:}
424:
425:int CCapture::SearchNearlyMarker(int px, int py, Mark *mark, int mn)
426:{
427:    int i, t = -1;
428:    double xl, yl, l, len = HUGE;
429:
430:    for (i=0;i < mn; i++) {
431:            xl = px - mark[i].x;
432:            yl = py - mark[i].y;
433:            l = sqrt(xl*xl + yl*yl);
434:            if (len>l) {
435:                    len = l;
436:                    t = i;
437:            }
438:    }
439:
440:    return t;
441:}
442:
443:short CCapture::Bufpix(int x, int y)
444:{
445:    short buf[2];
446:    CmtGenBlockRead(Addr(x, y), 1, 1, (unsigned char*)buf, sizeof(short)*2);
447://  buf[0] = Frame[x + y * GRAB_WIN_Y];
448:    buf[0] &= 0x7fff;
449:    return buf[0];
450:}
451:
452:void CCapture::Putpix(int x, int y, short src)
453:{
454:    short buf[2];
455:    buf[0] = src;
456:    CmtGenBlockWrite(Addr(x, y), 1, 1, (unsigned char*)buf, 2);
457:}
458:
459:void CCapture::CaptureStop()
460:{
461:    CmtVipGrabSetRate(CMT_VIP_GRAB_OFF);
462:}
463:
464:void CCapture::CaptureStart()
465:{
466:    CmtVipGrabSetRate(CMT_VIP_GRAB_CONTINUOUS);
467:}
```

```
 1:// GRABDLG.CPP : implementation file
 2://
 3:
 4:#include "stdafx.h"
 5:#include "mtrace.h"
 6:#include "trace.h"
 7:#include "GRABDLG.H"
 8:
 9:#include "capture.h"
10:
11:#ifdef _DEBUG
12:#undef THIS_FILE
13:static char BASED_CODE THIS_FILE[] = __FILE__;
14:#endif
15:
16:#define WM_DLGMSG WM_USER+301
17:
18://///////////////////////////////////////////////////////////////////////////
19:// CGrabDlg dialog
20:
21:
22:CGrabDlg::CGrabDlg(CWnd* pParent /*=NULL*/)
23:        : CDialog(CGrabDlg::IDD, pParent)
24:{
25:        m_hParent = pParent;
26:        //{{AFX_DATA_INIT(CGrabDlg)
27:                // NOTE: the ClassWizard will add member initialization here
28:        //}}AFX_DATA_INIT
29:}
30:
31:CGrabDlg::CGrabDlg(CCapture* pCapture) : CDialog()
32:{
33:        m_pCapture = pCapture;
34:}
35:
36:BOOL CGrabDlg::Create()
37:{
38:        return CDialog::Create(CGrabDlg::IDD);
39:}
40:
41:void CGrabDlg::DoDataExchange(CDataExchange* pDX)
42:{
43:        CDialog::DoDataExchange(pDX);
44:        //{{AFX_DATA_MAP(CGrabDlg)
45:                // NOTE: the ClassWizard will add DDX and DDV calls here
46:        //}}AFX_DATA_MAP
47:}
48:
49:BEGIN_MESSAGE_MAP(CGrabDlg, CDialog)
50:        //{{AFX_MSG_MAP(CGrabDlg)
51:        ON_WM_LBUTTONDOWN()
52:        //}}AFX_MSG_MAP
53:END_MESSAGE_MAP()
54:
55://
56:void CGrabDlg::SetView(CTraceView* pView)
57:{
58:        m_pView = pView;
59:}
60://///////////////////////////////////////////////////////////////////////////
61:// CGrabDlg message handlers
62:
63:void CGrabDlg::OnLButtonDown(UINT nFlags, CPoint point)
64:{
65:        if (point.x < 640 && point.y < 480)
66:                ::SendMessage(m_hParent->m_hWnd, WM_DLGMSG, (UINT)point.x, (LONG)point.y);
67:}
68:
```

```
 1:// mainfrm.cpp : implementation of the CMainFrame class
 2://
 3:
 4:#include "stdafx.h"
 5:#include "trace.h"
 6:
 7:#include "mainfrm.h"
 8:
 9:#ifdef _DEBUG
10:#undef THIS_FILE
11:static char BASED_CODE THIS_FILE[] = __FILE__;
12:#endif
13:
14://///////////////////////////////////////////////////////////////////////////
15:// CMainFrame
16:
17:IMPLEMENT_DYNCREATE(CMainFrame, CFrameWnd)
18:
19:BEGIN_MESSAGE_MAP(CMainFrame, CFrameWnd)
20:         //{{AFX_MSG_MAP(CMainFrame)
21:         //}}AFX_MSG_MAP
22:END_MESSAGE_MAP()
23:
24://///////////////////////////////////////////////////////////////////////////
25:// CMainFrame construction/destruction
26:
27:CMainFrame::CMainFrame()
28:{
29:         // TODO: add member initialization code here
30:}
31:
32:CMainFrame::~CMainFrame()
33:{
34:}
35:
36://///////////////////////////////////////////////////////////////////////////
37:// CMainFrame diagnostics
38:
39:#ifdef _DEBUG
40:void CMainFrame::AssertValid() const
41:{
42:         CFrameWnd::AssertValid();
43:}
44:
45:void CMainFrame::Dump(CDumpContext& dc) const
46:{
47:         CFrameWnd::Dump(dc);
48:}
49:
50:#endif //_DEBUG
51://///////////////////////////////////////////////////////////////////////////
52:// CMainFrame member functions
53:
54:BOOL CMainFrame::PreCreateWindow(CREATESTRUCT &cs)
55:{
56:         CFrameWnd::PreCreateWindow(cs);
57:
58:         TEXTMETRIC tm;
59:         CDC* pDC = GetDC();
60:         pDC->GetTextMetrics(&tm);
61:         ReleaseDC(pDC);
62:
63:         cs.lpszName = "マーカ追跡";
64:         cs.style = WS_OVERLAPPED | WS_SYSMENU;
65:         cs.cx = 800;
66:         cs.cy = 120;
67:         cs.x = 0;
68:         cs.y = 480;
```

```
69:         return TRUE;
70:}
71:
72://///////////////////////////////////////////////////////////////////////////
73:// CMainFrame message handlers
74:
```

```
 1:// serial.cpp : implementation of the CSerial class
 2:
 3:#include "stdafx.h"
 4:#include "serial.h"
 5:
 6://///////////////////////////////////////////////////////////////////////////
 7:// CSerial
 8:
 9:IMPLEMENT_DYNCREATE(CSerial, CObject)
10:
11://///////////////////////////////////////////////////////////////////////////
12:// CCerial construction
13:
14:CSerial::CSerial()
15:{
16:}
17:
18:CSerial::CSerial(HWND hwnd)
19:{
20:        m_hSWnd = hwnd;
21:}
22:
23://///////////////////////////////////////////////////////////////////////////
24:// CSerial member functions
25:
26:// 通信ﾃﾞﾊﾞｲｽの初期化
27:
28:int CSerial::InitSerial(int port, int baud, int parity, int length, int stop, int qs, int qr)
29:{
30:        char *npt[] = {"COM1", "COM2", "LPT1", "LPT2"};
31:        char *bud[] = {"1200", "2400", "4800", "9600", "19200"};
32:        char *lpt[] = {"n", "o", "e"};
33:        char *len[] = {"4", "5", "6", "7", "8"};
34:        char *stb[] = {"1", "1.5", "2"};
35:        char buf[128];
36:
37:        if (port   < 0 || port   > 3) port   = 0;
38:        if (baud   < 0 || baud   > 4) baud   = 3;
39:        if (parity < 0 || parity > 2) parity = 0;
40:        if (length < 0 || length > 4) length = 4;
41:        if (stop   < 0 || stop   > 2) stop   = 0;
42:
43:        if (qr < 128) qr = 128;
44:        if (qs < 128) qs = 128;
45:
46:        // 通信ﾃﾞﾊﾞｲｽｵｰﾌﾟﾝ
47:        if ((m_idComDev = OpenComm(npt[port], qr, qs)) < 0) return FALSE;
48:
49:        // 文字列をｼﾘｱﾙDCBに変換
50:        sprintf(buf, "%s:%s, %s, %s, %s", npt[port], bud[baud], lpt[parity], len[length], stb[stop]);
51:        if (BuildCommDCB(buf, &m_dcb) < 0) return FALSE;
52:
53:        // DCBの状態に通信ﾃﾞﾊﾞｲｽを設定
54:        if (SetCommState(&m_dcb) < 0) return FALSE;
55:
56:        // 初期値を保存する
57:        m_Port = port;
58:        m_Baud = baud;
59:        m_Parity = parity;
60:        m_Length = length;
61:        m_Stop = stop;
62:        m_Qr = qr;
63:        m_Qs = qs;
64:
65:        return TRUE;
66:}
67:
68:// 通信ﾃﾞﾊﾞｲｽの設定を変更する
```

```
 69:
 70:int CSerial::ChangeStatus(int baud, int parity, int length, int stop)
 71:{
 72:        char *npt[] = {"COM1", "COM2", "LPT1", "LPT2"};
 73:        char *bud[] = {"1200", "2400", "4800", "9600", "19200"};
 74:        char *lpt[] = {"n", "o", "e"};
 75:        char *len[] = {"4", "5", "6", "7", "8"};
 76:        char *stb[] = {"1", "1.5", "2"};
 77:        char buf[128];
 78:
 79:        if (baud   == CS_NOW) baud   = m_Baud;
 80:        if (parity == CS_NOW) parity = m_Parity;
 81:        if (length == CS_NOW) length = m_Length;
 82:        if (stop   == CS_NOW) stop   = m_Stop;
 83:
 84:        if (baud   < 0 || baud   > 4) baud   = 3;
 85:        if (parity < 0 || parity > 2) parity = 0;
 86:        if (length < 0 || length > 4) length = 4;
 87:        if (stop   < 0 || stop   > 2) stop   = 0;
 88:
 89:        // 文字列をｼﾘｱﾙDCBに変換
 90:        sprintf(buf, "%s:%s, %s, %s, %s", npt[m_Port], bud[baud], lpt[parity], len[length], stb[stop]);
 91:        if (BuildCommDCB(buf, &m_dcb) < 0) return FALSE;
 92:
 93:        // DCBの状態に通信ﾃﾞﾊﾞｲｽを設定
 94:        if (SetCommState(&m_dcb) < 0) return FALSE;
 95:
 96:        // 変更を保存する
 97:        m_Baud = baud;
 98:        m_Parity = parity;
 99:        m_Length = length;
100:        m_Stop = stop;
101:
102:        return TRUE;
103:}
104:
105:// 通信ﾃﾞﾊﾞｲｽをﾌﾞﾚｰｸ状態にする
106:
107:int CSerial::SetBreak()
108:{
109:        if (SetCommBreak(m_idComDev) == 0) {
110:                return TRUE;
111:        } else {
112:                return FALSE;
113:        }
114:}
115:
116:// 通信ﾃﾞﾊﾞｲｽのﾌﾞﾚｰｸ状態を解除する
117:
118:int CSerial::ClearBreak()
119:{
120:        if (ClearCommBreak(m_idComDev) == 0) {
121:                return TRUE;
122:        } else {
123:                return FALSE;
124:        }
125:}
126:
127:// 通信ﾃﾞﾊﾞｲｽをｸﾛｰｽﾞする
128:
129:int CSerial::CloseSerial()
130:{
131:        if (CloseComm(m_idComDev) == 0) {
132:                return TRUE;
133:        } else {
134:                return FALSE;
135:        }
136:}
```

```
137:
138:// 通信ﾃﾞﾊﾞｲｽの送信/受信ｷｭｰをﾌﾗｯｼｭする
139:
140:int CSerial::FlushQueue(int queue)
141:{
142:        if (queue < 0 || queue > 2) queue = 2;
143:
144:        // queue が2のときは送受信ともにﾌﾗｯｼｭする
145:        if (queue == 0 || queue == 2) {
146:                if (FlushComm(m_idComDev, 0) != 0) return FALSE;
147:        }
148:        if (queue == 1 || queue == 2) {
149:                if (FlushComm(m_idComDev, 1) != 0) return FALSE;
150:        }
151:        return TRUE;
152:}
153:
154:// WM_COMMNOTIFYﾒｯｾｰｼﾞのﾎﾟｽﾄの許可、不許可
155:
156:int CSerial::EnableNotifyMsg(int wNot, int oNot)
157:{
158:        if (!(EnableCommNotification(m_idComDev, m_hSWnd, wNot, oNot))) {
159:                return TRUE;
160:        } else {
161:                return FALSE;
162:        }
163:}
164:
165:// 通信ﾃﾞﾊﾞｲｽの最近のｴﾗｰ値を取得する
166:
167:int CSerial::GetLastError()
168:{
169:        return GetCommError(m_idComDev, NULL);
170:}
171:
172:// 通信ﾃﾞﾊﾞｲｽの状態を調べる
173:
174:int CSerial::GetSerialStatus(int queue)
175:{
176:        COMSTAT stat;
177:
178:        GetCommError(m_idComDev, &stat);
179:        switch (queue) {
180:                case CS_QUE_BOTH:
181:                        return (int) stat.status;
182:                case CS_QUE_RECEIVE: // 受信ｷｭｰの文字数を調べる
183:                        return (int) stat.cbInQue;
184:                case CS_QUE_SEND: // 送信ｷｭｰの文字数を調べる
185:                        return (int) stat.cbOutQue;
186:                default:
187:                        return -1;
188:                        break;
189:        }
190:}
191:
192:// ﾃﾞｰﾀ読み取り
193:
194:int CSerial::Read(unsigned char *buf, int num)
195:{
196:        if (num < 1) num = 1;
197:        return ReadComm(m_idComDev,(unsigned char*) buf, num);
198:}
199:
200:// ﾃﾞｰﾀ書き込み
201:
202:int CSerial::Write(unsigned char *buf, int num)
203:{
204:        int mem;
```

```
205:
206:        if (num < 1) num = 1;
207:        mem = GetSerialStatus(CS_QUE_SEND);     // 送信ｷｭｰの文字数を得る
208:        if ((m_Qs - mem) < num) return -1; // 送信ｷｭｰに空き容量はあるか
209:        return WriteComm(m_idComDev, buf, num);
210:}
211:
```

```
 1:// SOCKDLG.CPP : implementation file
 2://
 3:
 4:#include "stdafx.h"
 5:#include "trace.h"
 6:#include "SOCKDLG.H"
 7:
 8:#ifdef _DEBUG
 9:#undef THIS_FILE
10:static char BASED_CODE THIS_FILE[] = __FILE__;
11:#endif
12:
13://///////////////////////////////////////////////////////////////////////////
14:// CSockDlg dialog
15:
16:
17:CSockDlg::CSockDlg(CWnd* pParent /*=NULL*/)
18:        : CDialog(CSockDlg::IDD, pParent)
19:{
20:        //{{AFX_DATA_INIT(CSockDlg)
21:        m_Name = _T("ciris7");
22:        //}}AFX_DATA_INIT
23:}
24:
25:
26:void CSockDlg::DoDataExchange(CDataExchange* pDX)
27:{
28:        CDialog::DoDataExchange(pDX);
29:        //{{AFX_DATA_MAP(CSockDlg)
30:        DDX_Control(pDX, IDOK, m_Ok);
31:        DDX_Control(pDX, IDCANCEL, m_Cancel);
32:        DDX_CBString(pDX, IDC_SERVER, m_Name);
33:        //}}AFX_DATA_MAP
34:}
35:
36:BEGIN_MESSAGE_MAP(CSockDlg, CDialog)
37:        //{{AFX_MSG_MAP(CSockDlg)
38:        //}}AFX_MSG_MAP
39:END_MESSAGE_MAP()
40:
41:
42://///////////////////////////////////////////////////////////////////////////
43:// CSockDlg message handlers
44:
45:
```

```
 1:// socket.cpp : implementation of the CSocket class
 2://
 3:
 4:#include "stdafx.h"
 5:
 6:#include "socket.h"
 7:#include "c:¥winsock¥winsock.h"
 8:
 9:#define SEND_SIZE 128 // 送受信ﾃｰﾌﾞﾙの大きさ
10:
11://///////////////////////////////////////////////////////////////////////////
12:// CSocket
13:
14:IMPLEMENT_DYNCREATE(CSocket, CObject)
15:
16://///////////////////////////////////////////////////////////////////////////
17:// CSocket construction
18:
19:CSocket::CSocket()
20:{
21:        m_Socket = 0;
22:        m_SrvName = "";
23:}
24:
25://///////////////////////////////////////////////////////////////////////////
26:// CSocket member functions
27:
28:// ｻｰﾊﾞと接続する
29:
30:int CSocket::Connect()
31:{
32:        WORD    wVersionRequested;
33:        WSADATA wsaData;
34:        struct protoent FAR *pp;
35:        struct hostent FAR *host;
36:        struct sockaddr_in sin;
37:        int err;
38:
39:        wVersionRequested = (unsigned short)MAKEWORD(1, 1);
40:        if (WSAStartup(wVersionRequested, &wsaData) != 0) return -1;
41:
42:        // ｿｹｯﾄを生成する
43:        pp = getprotobyname("tcp");
44:        if ((m_Socket = socket(AF_INET, SOCK_STREAM, pp->p_proto)) < 0) {
45:                return -1;
46:        }
47:
48:        if ((host = gethostbyname(m_SrvName)) == NULL) {
49:                closesocket(m_Socket);
50:                WSACleanup();
51:                return -1;
52:        }
53:        _fmemset((char *)&sin, 0, sizeof(sin));
54:        sin.sin_family = AF_INET;
55:        sin.sin_port = htons(7000);
56:        _fmemcpy(&sin.sin_addr, host->h_addr, host->h_length);
57:
58:        // ｻｰﾊﾞと接続する
59:        if ((err = connect(m_Socket, (SOCKADDR *)&sin, sizeof(sin))) != 0) {
60:                shutdown(m_Socket, 2);
61:                closesocket(m_Socket);
62:                WSACleanup();
63:                return -1;
64:        }
65:        return m_Socket;
66:}
67:
68:// ｼｬｯﾄﾀﾞｳﾝを行う
```

```
 69:
 70:void CSocket::SShutdown()
 71:{
 72:        shutdown(m_Socket, 2); // ｻｰﾊﾞとの接続を切る
 73:        closesocket(m_Socket); // ｿｹｯﾄを閉じる
 74:        WSACleanup();
 75:}
 76:
 77:// ｻｰﾊﾞ名を変更する
 78:
 79:void CSocket::SetSrvName(CString buf)
 80:{
 81:        if (strcmp(buf, "") != 0)
 82:                m_SrvName = buf;
 83:}
 84:
 85:// 現在のｻｰﾊﾞ名を得る
 86:
 87:CString CSocket::GetSrvName()
 88:{
 89:        return m_SrvName;
 90:}
 91:
 92:// ｿｹｯﾄﾃﾞｨｽｸﾘﾌﾟﾀの変更
 93:
 94:void CSocket::SetSocket(int s)
 95:{
 96:        if (s >= 0)
 97:                m_Socket = s;
 98:}
 99:
100:// 現在のｿｹｯﾄﾃﾞｨｽｸﾘﾌﾟﾀを得る
101:
102:int CSocket::GetSocket()
103:{
104:        return m_Socket;
105:}
106:
107:// ﾊﾞｯﾌｧの内容を送信する
108:
109:void CSocket::Send(short Pkt_Tbl[])
110:{
111:        unsigned short pk[SEND_SIZE];
112:        int     psize = sizeof(short)*SEND_SIZE;
113:        int     len,i;
114:
115:        for (i=0;i<SEND_SIZE;i++) pk[i] = htons((unsigned short)Pkt_Tbl[i]);
116:
117:        len = send(m_Socket, (char *)pk, psize, 0);
118:}
119:
120:// 受信ﾃﾞｰﾀの先頭を返す
121:
122:short CSocket::Receive(int *len)
123:{
124:        static unsigned short tmp[SEND_SIZE];
125:
126:        *len = recv(m_Socket, (char *)tmp, sizeof(short)*SEND_SIZE, 0);
127:        return ntohs(tmp[0]);
128:}
129:
130:// ｻｰﾊﾞからの送信要求時にｳｨﾝﾄﾞｳｽﾞﾒｯｾｰｼﾞを発生させる
131:
132:int CSocket::StartInterrupt(HWND hWnd)
133:{
134:        if (WSAAsyncSelect(m_Socket, hWnd, WM_WINSOCK, FD_READ) != 0) {
135:                return WSAGetLastError();
136:        }
```

```
137:        return 0;
138:}
139:
```

```
1:// stdafx.cpp : source file that includes just the standard includes
2://  stdafx.pch will be the pre-compiled header
3://  stdafx.obj will contain the pre-compiled type information
4:
5:#include "stdafx.h"
6:
```

```
 1:// trace.cpp : Defines the class behaviors for the application.
 2://
 3:
 4:#include "stdafx.h"
 5:#include "mtrace.h"
 6:#include "trace.h"
 7:
 8:#include "mainfrm.h"
 9:#include "tracedoc.h"
10:#include "tracevw.h"
11:
12:#ifdef _DEBUG
13:#undef THIS_FILE
14:static char BASED_CODE THIS_FILE[] = __FILE__;
15:#endif
16:
17://///////////////////////////////////////////////////////////////////////////
18:// CTraceApp
19:
20:BEGIN_MESSAGE_MAP(CTraceApp, CWinApp)
21:         //{{AFX_MSG_MAP(CTraceApp)
22:         ON_COMMAND(ID_APP_ABOUT, OnAppAbout)
23:                 // NOTE - the ClassWizard will add and remove mapping macros here.
24:                 //    DO NOT EDIT what you see in these blocks of generated code!
25:         //}}AFX_MSG_MAP
26:         // Standard file based document commands
27:         ON_COMMAND(ID_FILE_NEW, CWinApp::OnFileNew)
28:         ON_COMMAND(ID_FILE_OPEN, CWinApp::OnFileOpen)
29:END_MESSAGE_MAP()
30:
31://///////////////////////////////////////////////////////////////////////////
32:// CTraceApp construction
33:
34:CTraceApp::CTraceApp()
35:{
36:         // TODO: add construction code here,
37:         // Place all significant initialization in InitInstance
38:}
39:
40://///////////////////////////////////////////////////////////////////////////
41:// The one and only CTraceApp object
42:
43:CTraceApp NEAR theApp;
44:
45://///////////////////////////////////////////////////////////////////////////
46:// CTraceApp initialization
47:
48:BOOL CTraceApp::InitInstance()
49:{
50:         // Standard initialization
51:         // If you are not using these features and wish to reduce the size
52:         //  of your final executable, you should remove from the following
53:         //  the specific initialization routines you do not need.
54:
55:         SetDialogBkColor();        // Set dialog background color to gray
56:         LoadStdProfileSettings();  // Load standard INI file options (including MRU)
57:
58:         // Register the application's document templates.  Document templates
59:         //  serve as the connection between documents, frame windows and views.
60:
61:         CSingleDocTemplate* pDocTemplate;
62:         pDocTemplate = new CSingleDocTemplate(
63:                 IDR_MAINFRAME,
64:                 RUNTIME_CLASS(CTraceDoc),
65:                 RUNTIME_CLASS(CMainFrame),     // main SDI frame window
66:                 RUNTIME_CLASS(CTraceView));
67:         AddDocTemplate(pDocTemplate);
68:
```

```
 69:        // create a new (empty) document
 70:        OnFileNew();
 71:
 72:        if (m_lpCmdLine[0] != '¥0')
 73:        {
 74:                // TODO: add command line processing here
 75:        }
 76:
 77:
 78:        return TRUE;
 79:}
 80:
 81://///////////////////////////////////////////////////////////////////////////
 82:// CAboutDlg dialog used for App About
 83:
 84:class CAboutDlg : public CDialog
 85:{
 86:public:
 87:        CAboutDlg();
 88:
 89:// Dialog Data
 90:        //{{AFX_DATA(CAboutDlg)
 91:        enum { IDD = IDD_ABOUTBOX };
 92:        //}}AFX_DATA
 93:
 94:// Implementation
 95:protected:
 96:        virtual void DoDataExchange(CDataExchange* pDX);    // DDX/DDV support
 97:        //{{AFX_MSG(CAboutDlg)
 98:                // No message handlers
 99:        //}}AFX_MSG
100:        DECLARE_MESSAGE_MAP()
101:};
102:
103:CAboutDlg::CAboutDlg() : CDialog(CAboutDlg::IDD)
104:{
105:        //{{AFX_DATA_INIT(CAboutDlg)
106:        //}}AFX_DATA_INIT
107:}
108:
109:void CAboutDlg::DoDataExchange(CDataExchange* pDX)
110:{
111:        CDialog::DoDataExchange(pDX);
112:        //{{AFX_DATA_MAP(CAboutDlg)
113:        //}}AFX_DATA_MAP
114:}
115:
116:BEGIN_MESSAGE_MAP(CAboutDlg, CDialog)
117:        //{{AFX_MSG_MAP(CAboutDlg)
118:                // No message handlers
119:        //}}AFX_MSG_MAP
120:END_MESSAGE_MAP()
121:
122:// App command to run the dialog
123:void CTraceApp::OnAppAbout()
124:{
125:        CAboutDlg aboutDlg;
126:        aboutDlg.DoModal();
127:}
128:
129://///////////////////////////////////////////////////////////////////////////
130:// CTraceApp commands
131:
```

```
 1:// tracedoc.cpp : implementation of the CTraceDoc class
 2://
 3:
 4:#include "stdafx.h"
 5:#include "trace.h"
 6:
 7:#include "tracedoc.h"
 8:
 9:#ifdef _DEBUG
10:#undef THIS_FILE
11:static char BASED_CODE THIS_FILE[] = __FILE__;
12:#endif
13:
14://///////////////////////////////////////////////////////////////////////////
15:// CTraceDoc
16:
17:IMPLEMENT_DYNCREATE(CTraceDoc, CDocument)
18:
19:BEGIN_MESSAGE_MAP(CTraceDoc, CDocument)
20:        //{{AFX_MSG_MAP(CTraceDoc)
21:                // NOTE - the ClassWizard will add and remove mapping macros here.
22:                //    DO NOT EDIT what you see in these blocks of generated code!
23:        //}}AFX_MSG_MAP
24:END_MESSAGE_MAP()
25:
26://///////////////////////////////////////////////////////////////////////////
27:// CTraceDoc construction/destruction
28:
29:CTraceDoc::CTraceDoc()
30:{
31:        // TODO: add one-time construction code here
32:}
33:
34:CTraceDoc::~CTraceDoc()
35:{
36:}
37:
38:BOOL CTraceDoc::OnNewDocument()
39:{
40:        if (!CDocument::OnNewDocument())
41:                return FALSE;
42:
43:        // TODO: add reinitialization code here
44:        // (SDI documents will reuse this document)
45:
46:        return TRUE;
47:}
48:
49://///////////////////////////////////////////////////////////////////////////
50:// CTraceDoc serialization
51:
52:void CTraceDoc::Serialize(CArchive& ar)
53:{
54:        if (ar.IsStoring())
55:        {
56:                // TODO: add storing code here
57:        }
58:        else
59:        {
60:                // TODO: add loading code here
61:        }
62:}
63:
64://///////////////////////////////////////////////////////////////////////////
65:// CTraceDoc diagnostics
66:
67:#ifdef _DEBUG
68:void CTraceDoc::AssertValid() const
```

```
69:{
70:         CDocument::AssertValid();
71:}
72:
73:void CTraceDoc::Dump(CDumpContext& dc) const
74:{
75:         CDocument::Dump(dc);
76:}
77:#endif //_DEBUG
78:
79://///////////////////////////////////////////////////////////////////////////
80:// CTraceDoc commands
81:
```

```
 1:// tracevw.cpp : implementation of the CTraceView class
 2://
 3:
 4:#include "stdafx.h"
 5:#include "mtrace.h"
 6:#include "trace.h"
 7:
 8:#include "tracedoc.h"
 9:#include "tracevw.h"
10:
11:#include "socket.h"                          // ｿｹｯﾄ通信用ｸﾗｽ
12:#include "c:¥winsock¥winsock.h" // Winsock ﾗｲﾌﾞﾗﾘ
13:#include "capture.h"                 // ﾏｰｶ追跡に関するｸﾗｽ
14:#include "grabdlg.h"                 // 入力画像表示用ﾀﾞｲｱﾛｸﾞ
15:#include "sockdlg.h"                 // ｿｹｯﾄに関する設定用ﾀﾞｲｱﾛｸﾞ
16:
17:#include "grabint.h"                 // ｸﾞﾗﾌﾞ割込み関数
18:#include "marksort.h"                // ﾏｰｶｿｰﾄ用関数
19:
20:#ifdef _DEBUG
21:#undef THIS_FILE
22:static char BASED_CODE THIS_FILE[] = __FILE__;
23:#endif
24:
25:#define WM_DLGMSG WM_USER+301
26:
27://///////////////////////////////////////////////////////////////////////////
28:// CTraceView
29:
30:IMPLEMENT_DYNCREATE(CTraceView, CView)
31:
32:BEGIN_MESSAGE_MAP(CTraceView, CView)
33:        //{{AFX_MSG_MAP(CTraceView)
34:        ON_COMMAND(ID_MENUSTART, OnMenustart)
35:        ON_COMMAND(ID_END, OnEnd)
36:        ON_WM_TIMER()
37:        ON_MESSAGE(WM_WINSOCK, OnWinsock)
38:        ON_MESSAGE(WM_DLGMSG, OnDlgMsg)
39:        ON_COMMAND(ID_RETRY, OnRetry)
40:        ON_WM_RBUTTONDOWN()
41:        ON_COMMAND(ID_MENUITEM32774, OnSetSock)
42:        //}}AFX_MSG_MAP
43:END_MESSAGE_MAP()
44:
45://///////////////////////////////////////////////////////////////////////////
46:// CTraceView construction/destruction
47:
48:CTraceView::CTraceView()
49:{
50:        m_SrvName = "ciris7";
51:        m_FFlag = TRUE;
52:        m_pDlg = new CGrabDlg(this);
53:        m_Mode[0] = " -検出- ";
54:        m_Mode[1] = " -入替- ";
55:}
56:
57:CTraceView::~CTraceView()
58:{
59:        delete m_pDlg;
60:}
61:
62://///////////////////////////////////////////////////////////////////////////
63:// CTraceView drawing
64:
65:void CTraceView::OnDraw(CDC* pDC)
66:{
67:        CTraceDoc* pDoc = GetDocument();
68:        ASSERT_VALID(pDoc);
```

```
 69:}
 70:
 71://///////////////////////////////////////////////////////////////////////////
 72:// CTraceView diagnostics
 73:
 74:#ifdef _DEBUG
 75:void CTraceView::AssertValid() const
 76:{
 77:        CView::AssertValid();
 78:}
 79:
 80:void CTraceView::Dump(CDumpContext& dc) const
 81:{
 82:        CView::Dump(dc);
 83:}
 84:
 85:CTraceDoc* CTraceView::GetDocument() // non-debug version is inline
 86:{
 87:        ASSERT(m_pDocument->IsKindOf(RUNTIME_CLASS(CTraceDoc)));
 88:        return (CTraceDoc*)m_pDocument;
 89:}
 90:#endif //_DEBUG
 91:
 92://///////////////////////////////////////////////////////////////////////////
 93:// Member functions
 94:
 95:// ｴﾗｰ表示用ﾒｯｾｰｼﾞﾎﾞｯｸｽ
 96:
 97:void CTraceView::PutErrorMsg(int err, int Param1, CString Param2)
 98:{
 99:        char buf[80];
100://      CClientDC *pDC = new CClientDC(this);
101:
102:        switch (err) {
103:                case ERR_CONFIG: // configﾌｧｲﾙｴﾗｰ
104:                        if (Param1 == -1)
105:                                sprintf(buf, "Cannot open config file '%s'", Param2);
106:                        else
107:                                sprintf(buf, "Config file error : line %d", Param1);
108:                        break;
109:                case ERR_SOCKET: // ｿｹｯﾄｺﾈｸﾄｴﾗｰ
110:                        sprintf(buf, "Cannot connect with the server '%s'", Param2);
111:                        break;
112:                case ERR_MARKER: // ﾏｰｶ検出ｴﾗｰ
113:                        sprintf(buf, "Found only %d marker(s)", Param1);
114:                        break;
115:                case ERR_MEMORY: // ﾒﾓﾘ確保ｴﾗｰ
116:                        sprintf(buf, "Not enough memory");
117:                        break;
118:                default:
119:                        return;
120:        }
121:        MessageBox(buf, NULL, MB_OK);
122://      delete pDC;
123:}
124:
125:// 中途終了
126:
127:void CTraceView::Terminate(int x)
128:{
129:        m_FFlag = TRUE;
130://      m_pDC->TextOut(700, 0, "          ");
131:        switch (x) {
132:                case 3:
133://                      m_pSocket->SShutdown();
134:                case 2: // ｸﾞﾗﾌﾞ･ﾏｰｶ追跡の中止、ﾒﾓﾘの解放
135:                        m_pCapture->CaptureQuit();
136:
```

```
137:                    delete xyp;
138:                    delete maskArea;
139:                    delete grnMarker;
140:                    delete redMarker;
141:                    delete marker;
142:                    delete orgMark;
143:                    delete orgVec;
144:            case 1:
145:                    delete m_pDC;
146:                    delete m_pCapture;
147:                    delete m_pSocket;
148:                    m_pDlg->DestroyWindow();
149:            case 0: // ｿｹｯﾄ通信の終了
150:            default:
151:                    break;
152:        }
153:        RedrawWindow(NULL, NULL, RDW_ERASE | RDW_INVALIDATE);
154:}
155:
156:////////////////////////////////////////////////////////////////////////////
157:// CTraceView message handlers
158:
159:void CTraceView::OnMenustart()
160:{
161:        char str[128];                  // 文字列処理用
162://      CClientDC nDC(this);    // ﾒｲﾝﾌﾚｰﾑﾋﾞｭｰへのﾃﾞﾊﾞｲｽｺﾝﾃｷｽﾄ
163:        MSG lpmsg;                              // Windows ﾒｯｾｰｼﾞ処理用
164:        int result, i;
165:        int mNum;
166:        // ﾏｰｶ検出用
167:        int t;
168:        int mn = 0;
169:        // 画面描画用
170:
171:        if (m_FFlag == FALSE) return;
172:
173:        CDC *phdc = new CDC();
174:        CDC *phdcMem = new CDC();
175:        CBitmap *phBitmap = new CBitmap();
176:        CBrush *phBrush = new CBrush();
177:        CPen *phPen = new CPen(PS_SOLID, 1, RGB(255, 0, 0));
178:        RECT rect1;
179:        // Winsock 関連
180:        m_pSocket = new CSocket();                              // ｿｹｯﾄｵﾌﾞｼﾞｪｸﾄ
181:        // ｷｬﾌﾟﾁｬﾎﾞｰﾄﾞ関連
182:        m_pCapture = new CCapture();                    // ｷｬﾌﾟﾁｬﾎﾞｰﾄﾞｵﾌﾞｼﾞｪｸﾄ
183:        m_pDC = new CClientDC(this);
184:
185:        m_Condition = 0;
186:        m_RFlag = FALSE;
187:        m_FFlag = FALSE;
188:        m_pDlg->SetView(this);
189:        m_pDC->TextOut(700, 0, m_Mode[m_RFlag]);
190:
191:        // ｷｬﾌﾟﾁｬ画像表示用ﾓｰﾄﾞﾚｽﾀﾞｲｱﾛｸﾞ(移動不可)
192:        if (m_pDlg->GetSafeHwnd() == 0) {
193:                m_pDlg->Create();
194:        }
195:        m_pDlg->MoveWindow(0, 0, 800, GRAB_WIN_Y, TRUE);
196:
197:        ////////// ｿｹｯﾄ通信 //////////
198:
199:        for (i = 0; i< 256; i++) m_SockBuf[i] = 0; // 送信ﾊﾞｯﾌｧを初期化
200:
201:        ////////// ﾏｰｶ検出 //////////
202:
203:        phdc = m_pDlg->GetDC(); // ﾋﾞﾃﾞｵ入力表示用ﾀﾞｲｱﾛｸﾞのﾃﾞﾊﾞｲｽｺﾝﾃｷｽﾄを取得
204:        if (phdc == NULL) {
```

```
205:                m_pDC->TextOut(0, 0, "ﾃﾞﾊﾞｲｽｺﾝﾃｷｽﾄが取得できません");
206:                return;
207:        }
208:
209:        phBrush->CreateSolidBrush(RGB(255, 0, 0)); // 赤のﾌﾞﾗｼ
210:
211:        // Configﾌｧｲﾙの読み込み
212:        result = m_pCapture->ReadConfig("config.dat");
213:        if (result != 0) {
214://              sprintf(str, "Config ﾌｧｲﾙｴﾗｰ (line:%d)", result);
215:                ReleaseDC(phdc);
216:                Terminate(1);
217:                PutErrorMsg(ERR_CONFIG, result, "config.dat");
218:                return;
219:        }
220:
221:        // 領域確保      ('redMarker'は視線追跡拡張時のためのﾀﾞﾐｰ)
222:        mNum = m_pCapture->m_Conf.nOfGreen; // + m_pCapture->m_Conf.nOfRed;
223:        grnMarker = new Mark[m_pCapture->m_Conf.nOfGreen];
224:        redMarker = new Mark[m_pCapture->m_Conf.nOfRed];
225:        marker = new Mark[mNum];
226:        orgVec = new Mark[mNum];
227:        orgMark = new Mark[mNum];
228:        maskArea = new Area[mNum];
229:        xyp = new short[mNum+mNum];
230:
231:/*
232:        if (xyp == NULL) {
233:                m_pDC->TextOut(0, 0, ">>領域確保ｴﾗｰ");
234:                Terminate(2);
235://              PutErrorMsg(ERR_MEMORY, 0, "");
236:                return;
237:        }
238:*/
239:
240:        // Configﾌｧｲﾙの内容を保存
241:        for (i=0; i < m_pCapture->m_Conf.nOfCenter; i++) {
242:                t = m_pCapture->m_Conf.center[i].id - 1;
243:                maskArea[t] = m_pCapture->m_Conf.center[i].area;
244:        }
245:        for (i=0; i < m_pCapture->m_Conf.nOfUpper; i++) {
246:                t = m_pCapture->m_Conf.upper[i].id - 1;
247:                maskArea[t] = m_pCapture->m_Conf.upper[i].area;
248:        }
249:        for (i=0; i < m_pCapture->m_Conf.nOfBottom; i++) {
250:                t = m_pCapture->m_Conf.bottom[i].id - 1;
251:                maskArea[t] = m_pCapture->m_Conf.bottom[i].area;
252:        }
253:
254:        // ｷｬﾌﾟﾁｬﾎﾞｰﾄﾞ初期化、grab開始、grab割込み開始
255:        m_pCapture->CaptureInit();
256:
257:        // ﾏｰｶ初期位置検出
258:        result = m_pCapture->ScanMarkerPosition(grnMarker, redMarker);
259:                phdc->SetTextColor(RGB(0, 0, 0));
260:                sprintf(str, "%d 個のﾏｰｶを検出", result);
261:                m_pDC->TextOut(0, 0, str);
262:        if (result != m_pCapture->m_Conf.nOfGreen) {
263://              m_pDC->TextOut(0, 20, "ﾏｰｶの数が足りません");
264:                ReleaseDC(phdc);
265:                Terminate(2);
266:                PutErrorMsg(ERR_MARKER, result, "");
267:                return;
268:        }
269:
270:        // ﾏｰｶ番号の整列
271:        lineUPMarker(&m_pCapture->m_Conf, grnMarker, redMarker, marker, &mn);
272:
```

```
273:        // ﾏｰｶ初期位置の保存
274:        for (i = 0; i < mn; i++) {
275:                orgMark[i] = marker[i];
276:                orgVec[i].x = orgMark[0].x - orgMark[i].x;
277:                orgVec[i].y = orgMark[0].y - orgMark[i].y;
278:                xyp[i+i+1] = orgVec[i].x;
279:                xyp[i+i+0] = orgVec[i].y;
280:        }
281:        memcpy(m_SockBuf, xyp, sizeof(short)*mNum*2);
282:
283:        ////////// ｿｹｯﾄ通信 //////////
284:        m_pSocket->SetSrvName(m_SrvName);
285:        if ((m_Socket = m_pSocket->Connect()) != -1) { // ｻｰﾊﾞにｺﾈｸﾄする
286:                m_pDC->TextOut(0, 0, "Connect with the server.");
287:        } else {
288://              sprintf(str, "ｻｰﾊﾞ %s とｺﾈｸﾄできません", m_SrvName);
289://              m_pDC->TextOut(0, 0, str);
290:                ReleaseDC(phdc);
291:                Terminate(3);
292:                PutErrorMsg(ERR_SOCKET, 0, m_SrvName);
293:                return;
294:        }
295:        m_pSocket->Send(m_SockBuf); // ｻｰﾊﾞに初期位置を送信
296:        m_pSocket->StartInterrupt(m_hWnd); // 送信要求割込み開始
297:
298:        phdc->SetTextColor(RGB(255, 0, 0));     // ﾃｷｽﾄの色を設定
299:        phdc->SetBkMode(TRANSPARENT); // ﾃｷｽﾄのﾊﾞｯｸｸﾞﾗｳﾝﾄﾞを描画しない
300:        SetTimer(1, 10000, NULL); // ｲﾝﾀｰﾊﾞﾙﾀｲﾏの設定
301:        m_Counter = 0;
302:
303:        // ﾒｲﾝﾙｰﾌﾟ (ｱﾌﾟﾘｹｰｼｮﾝの終了ﾒｯｾｰｼﾞが来るか「終了」が選択されるまでﾙｰﾌﾟ)
304:        while ((lpmsg.message != WM_QUIT) && (m_FFlag == FALSE)) {
305:                if (PeekMessage(&lpmsg, NULL, 0, 0, PM_REMOVE) != 0) { // ﾒｯｾｰｼﾞｷｭｰをpeek
306:                        TranslateMessage(&lpmsg);
307:                        DispatchMessage(&lpmsg);
308:                } else {
309:                        if (GrabFlag == TRUE) { // grab割込みが発生したかどうか
310:                                GrabFlag = FALSE;
311:                                m_CFlag = FALSE;
312:                                m_Counter++; // ﾊﾟﾌｫｰﾏﾝｽ測定用
313:                                for (i = 0; i < mn; i++) {
314:                                        m_pCapture->ScanMarkMove(maskArea[i], orgMark[i], &marker[i]);
315:                                        xyp[i+i+1] = (orgVec[i].x - (marker[0].x - marker[i].x))*100;
316:                                        xyp[i+i+0] = (orgVec[i].y - (marker[0].y - marker[i].y))*80;
317:                                        rect1.left = marker[i].x - 14;
318:                                        rect1.top = marker[i].y - 14;
319:                                        rect1.right = marker[i].x + 14;
320:                                        rect1.bottom = marker[i].y + 14;
321:                                        phdc->FrameRect(&rect1, phBrush);
322:                                        sprintf(str, "%d", i+1);
323:                                        phdc->TextOut(marker[i].x + 12, marker[i].y + 4, str);
324:                                }
325:                                memcpy(m_SockBuf, xyp, sizeof(short)*mNum*2);
326://                              m_pCapture->ChangeChannel();
327:                        }
328:                }
329:        }
330:        ReleaseDC(phdc);
331:        delete m_pDC;
332:}
333:
334:// 「終了」ﾒﾆｭｰが選択されたとき
335:
336:void CTraceView::OnEnd()
337:{
338:        if (m_FFlag == FALSE) { // 「開始」ﾒﾆｭｰが選択されているか
339://              m_pDC->TextOut(700, 0, "            ");
340:                m_FFlag = TRUE;
```

```
341:                KillTimer(1); // ｲﾝﾀｰﾊﾞﾙﾀｲﾏ
342:                m_pCapture->CaptureQuit(); // ｷｬﾌﾟﾁｬﾎﾞｰﾄﾞ
343:                m_pDlg->DestroyWindow(); // grab画面表示用ﾀﾞｲｱﾛｸﾞ
344:                m_pSocket->SShutdown(); // ｿｹｯﾄ通信
345:                delete xyp;
346:                delete maskArea;
347:                delete grnMarker;
348:                delete redMarker;
349:                delete marker;
350:                delete orgMark;
351:                delete orgVec;
352:                delete m_pCapture;
353:                delete m_pSocket;
354:                RedrawWindow(NULL, NULL, RDW_ERASE | RDW_INVALIDATE);
355:        }
356:}
357:
358:// ｲﾝﾀﾊﾞﾙﾀｲﾏ
359:
360:void CTraceView::OnTimer(UINT nIDEvent)
361:{
362:        char buf[80];
363:        int a;
364:        CClientDC *pDC = new CClientDC(this);
365:
366:        m_pDC->TextOut(0, 20, "                                    ");
367:        m_pDC->TextOut(0, 40, "                                    ");
368:        a = (int)(m_Counter / 10); // ﾊﾟﾌｫｰﾏﾝｽの算出
369:        sprintf(buf, "1秒間に %d 回のﾏｰｶ検出", a);
370:        pDC->TextOut(0, 0, buf);
371:        m_Counter = 0; // ｶｳﾝﾀｸﾘｱ
372:        CView::OnTimer(nIDEvent);
373:}
374:
375:// ｻｰﾊﾞからの送信要求ﾒｯｾｰｼﾞ処理
376:
377:LONG CTraceView::OnWinsock(UINT Param1, LONG Param2)
378:{
379:        int Socket;
380:        int len;
381:        short tmp;
382:        fd_set r_mask;
383:
384:        Socket = m_pSocket->GetSocket();
385:        FD_ZERO(&r_mask);
386:        FD_SET(Socket, &r_mask);
387:
388:        select( 0, &r_mask, NULL, NULL, NULL);
389:
390:        if (FD_ISSET(Socket, &r_mask)) {
391:                tmp = m_pSocket->Receive(&len); // ｻｰﾊﾞからのﾒｯｾｰｼﾞを受信
392:                if (tmp == 1) {
393:                        m_pSocket->Send(m_SockBuf);
394:                }
395:        }
396:
397:        return 0;
398:}
399:
400:// 「再検出」ﾒﾆｭｰが選択されたとき
401:
402:void CTraceView::OnRetry()
403:{
404:        Mark tmp1[20], tmp2[20];
405:        int result, n, mn;
406:
407:        if (m_FFlag == FALSE) {
408:                n = m_pCapture->m_Conf.nOfGreen;
```

```
409:                result = m_pCapture->ScanMarkerPosition(tmp1, tmp2);
410:                if (result == n) {
411:                        lineUPMarker(&m_pCapture->m_Conf, tmp1, tmp2, marker, &mn);
412:                } else {
413:                        m_pDC->TextOut(0, 20, "全てのﾏｰｶが検出できませんでした");
414:                }
415:        }
416:        return;
417:}
418:
419:// ﾋﾞﾃﾞｵ画像表示用ﾀﾞｲｱﾛｸﾞ上で左ｸﾘｯｸされた時
420:
421:LONG CTraceView::OnDlgMsg(UINT mx, LONG my)
422:{
423:        Mark dmy;
424:        char buf[80];
425:        int x, y;
426:
427:        x = (int)mx;
428:        y = (int)my;
429:
430:        if (m_FFlag == TRUE) return 0;
431:
432:        sprintf(buf, "%d, %d", x, y);
433:        m_pDC->TextOut(0, 20, buf);
434:        if (m_Condition == 0) {
435:                m_s1 = m_pCapture->SearchNearlyMarker(x, y, marker, m_pCapture->m_Conf.nOfGreen);
436:                sprintf(buf, "Marker(%2d) Mode(%2d)  ", m_s1, m_Condition);
437:                m_pDC->TextOut(0, 40, buf);
438:                if (m_s1 < 0) m_Condition = 1 - m_Condition;
439:        } else
440:        if (m_Condition == 1) {
441:                if (m_RFlag == TRUE) { // ﾏｰｶを入れ替えるﾓｰﾄﾞ
442:                        m_s2 = m_pCapture->SearchNearlyMarker(x, y, marker, m_pCapture->m_Conf.nOfGreen);
443:                        sprintf(buf, "Marker(%2d) Mode(%2d)  ", m_s2, m_Condition);
444:                        m_pDC->TextOut(0, 40, buf);
445:                        if (m_s2 >= 0) {
446:                                dmy = orgMark[m_s1]; orgMark[m_s1] = orgMark[m_s2]; orgMark[m_s2] = dmy;
447:                                dmy = marker[m_s1];  marker[m_s1]  = marker[m_s2];  marker[m_s2]  = dmy;
448:                                dmy = orgVec[m_s1];  orgVec[m_s1]  = orgVec[m_s2];  orgVec[m_s2]  = dmy;
449:                        } else {
450:                                m_Condition = 1 - m_Condition;
451:                        }
452:                } else { // 検出範囲を移動するﾓｰﾄﾞ
453:                        marker[m_s1].x = x;
454:                        marker[m_s1].y = y;
455:                }
456:        }
457:        m_Condition = 1 - m_Condition;
458:        return 0;
459:}
460:
461:// ﾒｲﾝﾌﾚｰﾑ上で右ｸﾘｯｸされたとき
462:
463:void CTraceView::OnRButtonDown(UINT nFlags, CPoint point)
464:{
465:        if (m_FFlag==FALSE) {
466:                if (m_RFlag==FALSE)     {
467:                        m_RFlag = TRUE;
468:                } else {
469:                        m_RFlag = FALSE;
470:                }
471:                m_pDC->TextOut(700, 0, m_Mode[m_RFlag]);
472:        }
473:        CView::OnRButtonDown(nFlags, point);
474:}
475:
476:// 「通信に関する設定」ﾒﾆｭｰが選択されたとき
```

```
477:
478:void CTraceView::OnSetSock()
479:{
480:        if (m_FFlag == TRUE) {
481:                CSockDlg sDlg; // 通信関係設定用ﾀﾞｲｱﾛｸﾞｵﾌﾞｼﾞｪｸﾄ
482:
483:                sDlg.m_Name = m_SrvName;
484:                if (sDlg.DoModal() == IDOK) { // ﾓｰﾀﾞﾙﾀﾞｲｱﾛｸﾞとして表示
485:                        m_SrvName = sDlg.m_Name;
486:                }
487:        }
488:}
489:
490://///////////////////////////////////////////////////////////////////////////
491:// Winsock Implementation
492:
493:int PASCAL FAR __WSAFDIsSet(SOCKET fd, fd_set FAR *set)
494:{
495:        int i = set->fd_count;
496:
497:        while(i--)
498:                if (set->fd_array[i] == fd)
499:                        return 1;
500:
501:        return 0;
502:}
503:
504:
```

```
 1:// VINDLG.CPP : implementation file
 2://
 3:
 4:#include "stdafx.h"
 5:#include "trace.h"
 6:#include "VINDLG.H"
 7:
 8:#ifdef _DEBUG
 9:#undef THIS_FILE
10:static char BASED_CODE THIS_FILE[] = __FILE__;
11:#endif
12:
13://///////////////////////////////////////////////////////////////////////////
14:// CVinDlg dialog
15:
16:
17:CVinDlg::CVinDlg(CWnd* pParent /*=NULL*/)
18:        : CDialog(CVinDlg::IDD, pParent)
19:{
20:        //{{AFX_DATA_INIT(CVinDlg)
21:                // NOTE: the ClassWizard will add member initialization here
22:        //}}AFX_DATA_INIT
23:}
24:
25:
26:void CVinDlg::DoDataExchange(CDataExchange* pDX)
27:{
28:        CDialog::DoDataExchange(pDX);
29:        //{{AFX_DATA_MAP(CVinDlg)
30:                // NOTE: the ClassWizard will add DDX and DDV calls here
31:        //}}AFX_DATA_MAP
32:}
33:
34:BEGIN_MESSAGE_MAP(CVinDlg, CDialog)
35:        //{{AFX_MSG_MAP(CVinDlg)
36:                // NOTE: the ClassWizard will add message map macros here
37:        //}}AFX_MSG_MAP
38:END_MESSAGE_MAP()
39:
40:
41://///////////////////////////////////////////////////////////////////////////
42:// CVinDlg message handlers
43:
```

```
a=77
Ew=83
l1=24
k1max=51
k2max=33
jmax=195
imax=255
```

```
1:# Microsoft Visual C++ generated build script - Do not modify
2:
3:PROJ = TRACE
4:DEBUG = 0
5:PROGTYPE = 0
6:CALLER =
7:ARGS =
8:DLLS =
9:D_RCDEFINES = /d_DEBUG
10:R_RCDEFINES = /dNDEBUG
11:ORIGIN = MSVC
12:ORIGIN_VER = 1.00
13:PROJPATH = C:¥SOURCE¥TRACE¥
14:USEMFC = 1
15:CC = cl
16:CPP = cl
17:CXX = cl
18:CCREATEPCHFLAG =
19:CPPCREATEPCHFLAG = /YcSTDAFX.H
20:CUSEPCHFLAG =
21:CPPUSEPCHFLAG = /YuSTDAFX.H
22:FIRSTC = GRABINT.C
23:FIRSTCPP = STDAFX.CPP
24:RC = rc
25:CFLAGS_D_WEXE = /nologo /G2 /W3 /Zi /AL /Od /D "_DEBUG" /D "WIN30" /D "CMT" /FR /GA /Fd"TRACE.PDB"
26:CFLAGS_R_WEXE = /nologo /Gs /G2 /W3 /AL /D "NDEBUG" /D "WIN30" /D "CMT" /FR /GA
27:LFLAGS_D_WEXE = /NOLOGO /NOD /PACKC:61440 /STACK:10240 /ALIGN:16 /ONERROR:NOEXE /CO
28:LFLAGS_R_WEXE = /NOLOGO /NOD /PACKC:61440 /STACK:10240 /NOPACKF /ALIGN:16 /ONERROR:NOEXE
29:LIBS_D_WEXE = lafxcwd oldnames libw llibcew commdlg.lib shell.lib
30:LIBS_R_WEXE = lafxcw oldnames libw llibcew commdlg.lib shell.lib
31:RCFLAGS = /nologo /z
32:RESFLAGS = /nologo /t
33:RUNFLAGS =
34:DEFFILE = TRACE.DEF
35:OBJS_EXT = MARKSORT.OBJ
36:LIBS_EXT = ..¥..¥WINSOCK¥WINSOCK.LIB
37:!if "$(DEBUG)" == "1"
38:CFLAGS = $(CFLAGS_D_WEXE)
39:LFLAGS = $(LFLAGS_D_WEXE)
40:LIBS = $(LIBS_D_WEXE)
41:MAPFILE = nul
42:RCDEFINES = $(D_RCDEFINES)
43:!else
44:CFLAGS = $(CFLAGS_R_WEXE)
45:LFLAGS = $(LFLAGS_R_WEXE)
46:LIBS = $(LIBS_R_WEXE)
47:MAPFILE = nul
48:RCDEFINES = $(R_RCDEFINES)
49:!endif
50:!if [if exist MSVC.BND del MSVC.BND]
51:!endif
52:SBRS = STDAFX.SBR ¥
53:                TRACE.SBR ¥
54:                MAINFRM.SBR ¥
55:                TRACEDOC.SBR ¥
56:                TRACEVW.SBR ¥
57:                SOCKET.SBR ¥
58:                GRABDLG.SBR ¥
59:                VINDLG.SBR ¥
60:                SOCKDLG.SBR ¥
61:                CAPTURE.SBR ¥
62:                GRABINT.SBR
63:
64:
65:WINSOCK_DEP =
66:
67:TRACE_RCDEP = c:¥source¥trace¥res¥trace.ico ¥
68:        c:¥source¥trace¥res¥trace.rc2
```

```
 69:
 70:
 71:STDAFX_DEP = c:¥source¥trace¥stdafx.h
 72:
 73:
 74:TRACE_DEP = c:¥source¥trace¥stdafx.h ¥
 75:        c:¥source¥trace¥trace.h ¥
 76:        c:¥source¥trace¥mainfrm.h ¥
 77:        c:¥source¥trace¥tracedoc.h ¥
 78:        c:¥source¥trace¥tracevw.h
 79:
 80:
 81:MAINFRM_DEP = c:¥source¥trace¥stdafx.h ¥
 82:        c:¥source¥trace¥trace.h ¥
 83:        c:¥source¥trace¥mainfrm.h
 84:
 85:
 86:TRACEDOC_DEP = c:¥source¥trace¥stdafx.h ¥
 87:        c:¥source¥trace¥trace.h ¥
 88:        c:¥source¥trace¥tracedoc.h
 89:
 90:
 91:TRACEVW_DEP = c:¥source¥trace¥stdafx.h ¥
 92:        c:¥source¥trace¥mtrace.h ¥
 93:        c:¥source¥trace¥trace.h ¥
 94:        c:¥source¥trace¥tracedoc.h ¥
 95:        c:¥source¥trace¥tracevw.h ¥
 96:        c:¥source¥trace¥socket.h ¥
 97:        c:¥winsock¥winsock.h ¥
 98:        c:¥source¥trace¥capture.h ¥
 99:        c:¥source¥trace¥grabdlg.h ¥
100:        c:¥source¥trace¥grabint.h
101:
102:
103:SOCKET_DEP = c:¥source¥trace¥stdafx.h ¥
104:        c:¥source¥trace¥socket.h ¥
105:        c:¥winsock¥winsock.h
106:
107:
108:GRABDLG_DEP = c:¥source¥trace¥stdafx.h ¥
109:        c:¥source¥trace¥trace.h ¥
110:        c:¥source¥trace¥grabdlg.h
111:
112:
113:VINDLG_DEP = c:¥source¥trace¥stdafx.h ¥
114:        c:¥source¥trace¥trace.h ¥
115:        c:¥source¥trace¥vindlg.h
116:
117:
118:SOCKDLG_DEP = c:¥source¥trace¥stdafx.h ¥
119:        c:¥source¥trace¥trace.h ¥
120:        c:¥source¥trace¥sockdlg.h
121:
122:
123:CAPTURE_DEP = c:¥source¥trace¥stdafx.h ¥
124:        c:¥source¥trace¥mtrace.h ¥
125:        c:¥source¥trace¥capture.h ¥
126:        c:¥cmtlib¥include¥cmtlib.h ¥
127:        c:¥cmtlib¥include¥vmtxdef.h ¥
128:        c:¥cmtlib¥include¥cmtdef.h ¥
129:        c:¥cmtlib¥include¥cmttype.h ¥
130:        c:¥cmtlib¥include¥cmtfnct.h ¥
131:        c:¥cmtlib¥include¥cmtvar.h ¥
132:        c:¥source¥trace¥grabint.h
133:
134:
135:GRABINT_DEP = c:¥cmtlib¥include¥cmtlib.h ¥
136:        c:¥cmtlib¥include¥vmtxdef.h ¥
```

```
137:        c:¥cmtlib¥include¥cmtdef.h ¥
138:        c:¥cmtlib¥include¥cmttype.h ¥
139:        c:¥cmtlib¥include¥cmtfnct.h ¥
140:        c:¥cmtlib¥include¥cmtvar.h
141:
142:
143:MARKSORT_DEP =
144:
145:all:    $(PROJ).EXE $(PROJ).BSC
146:
147:TRACE.RES:      TRACE.RC $(TRACE_RCDEP)
148:        $(RC) $(RCFLAGS) $(RCDEFINES) -r TRACE.RC
149:
150:STDAFX.OBJ:     STDAFX.CPP $(STDAFX_DEP)
151:        $(CPP) $(CFLAGS) $(CPPCREATEPCHFLAG) /c STDAFX.CPP
152:
153:TRACE.OBJ:      TRACE.CPP $(TRACE_DEP)
154:        $(CPP) $(CFLAGS) $(CPPUSEPCHFLAG) /c TRACE.CPP
155:
156:MAINFRM.OBJ:    MAINFRM.CPP $(MAINFRM_DEP)
157:        $(CPP) $(CFLAGS) $(CPPUSEPCHFLAG) /c MAINFRM.CPP
158:
159:TRACEDOC.OBJ:   TRACEDOC.CPP $(TRACEDOC_DEP)
160:        $(CPP) $(CFLAGS) $(CPPUSEPCHFLAG) /c TRACEDOC.CPP
161:
162:TRACEVW.OBJ:    TRACEVW.CPP $(TRACEVW_DEP)
163:        $(CPP) $(CFLAGS) $(CPPUSEPCHFLAG) /c TRACEVW.CPP
164:
165:SOCKET.OBJ:     SOCKET.CPP $(SOCKET_DEP)
166:        $(CPP) $(CFLAGS) $(CPPUSEPCHFLAG) /c SOCKET.CPP
167:
168:GRABDLG.OBJ:    GRABDLG.CPP $(GRABDLG_DEP)
169:        $(CPP) $(CFLAGS) $(CPPUSEPCHFLAG) /c GRABDLG.CPP
170:
171:VINDLG.OBJ:     VINDLG.CPP $(VINDLG_DEP)
172:        $(CPP) $(CFLAGS) $(CPPUSEPCHFLAG) /c VINDLG.CPP
173:
174:SOCKDLG.OBJ:    SOCKDLG.CPP $(SOCKDLG_DEP)
175:        $(CPP) $(CFLAGS) $(CPPUSEPCHFLAG) /c SOCKDLG.CPP
176:
177:CAPTURE.OBJ:    CAPTURE.CPP $(CAPTURE_DEP)
178:        $(CPP) $(CFLAGS) $(CPPUSEPCHFLAG) /c CAPTURE.CPP
179:
180:GRABINT.OBJ:    GRABINT.C $(GRABINT_DEP)
181:        $(CC) $(CFLAGS) $(CCREATEPCHFLAG) /c GRABINT.C
182:
183:
184:$(PROJ).EXE::   TRACE.RES
185:
186:$(PROJ).EXE::   STDAFX.OBJ TRACE.OBJ MAINFRM.OBJ TRACEDOC.OBJ TRACEVW.OBJ SOCKET.OBJ ¥
187:        GRABDLG.OBJ VINDLG.OBJ SOCKDLG.OBJ CAPTURE.OBJ GRABINT.OBJ $(OBJS_EXT) $(DEFFILE)
188:        echo >NUL @<<$(PROJ).CRF
189:STDAFX.OBJ +
190:TRACE.OBJ +
191:MAINFRM.OBJ +
192:TRACEDOC.OBJ +
193:TRACEVW.OBJ +
194:SOCKET.OBJ +
195:GRABDLG.OBJ +
196:VINDLG.OBJ +
197:SOCKDLG.OBJ +
198:CAPTURE.OBJ +
199:GRABINT.OBJ +
200:$(OBJS_EXT)
201:$(PROJ).EXE
202:$(MAPFILE)
203:C:¥MSVC¥CDK16¥LIB¥+
204:c:¥msvc¥lib¥+
```

```
c:¥msvc¥mfc¥lib¥+
c:¥windows¥+
..¥..¥WINSOCK¥WINSOCK.LIB+
$(LIBS)
$(DEFFILE);
<<
        link $(LFLAGS) @$(PROJ).CRF
        $(RC) $(RESFLAGS) TRACE.RES $@
        @copy $(PROJ).CRF MSVC.BND

$(PROJ).EXE::   TRACE.RES
        if not exist MSVC.BND   $(RC) $(RESFLAGS) TRACE.RES $@

run: $(PROJ).EXE
        $(PROJ) $(RUNFLAGS)


$(PROJ).BSC: $(SBRS)
        bscmake @<<
/o$@ $(SBRS)
<<

```

```
 1:; trace.def : Declares the module parameters for the application.
 2:
 3:NAME         TRACE
 4:DESCRIPTION  'TRACE'
 5:EXETYPE      WINDOWS
 6:
 7:CODE         PRELOAD MOVEABLE DISCARDABLE
 8:DATA         PRELOAD MOVEABLE MULTIPLE
 9:
10:HEAPSIZE     1024   ; initial heap size
11:; Stack size is passed as argument to linker's /STACK option
12:IMPORTS
13:;Error function
14:   CMTERR.CMTErrGetErrorString
15:
16:;Gen functions
17:   CMTGEN.CMTGenLibInit
18:   CMTGEN.CMTGenLibQuit
19:   CMTGEN.CMTGenBoardReset
20:   CMTGEN.CMTGenBoardSelect
21:   CMTGEN.CMTGenContextRestore
22:   CMTGEN.CMTGenContextSave
23:   CMTGEN.CMTGenLibGetVersion
24:   CMTGEN.CMTGenEbiBusEnable
25:   CMTGEN.CMTGenEbiSetBusMode
26:   CMTGEN.CMTGenAcsSetVGABuffer
27:   CMTGEN.CMTGenAcsGetVGABuffer
28:   CMTGEN.CMTGenXpressGetDelay
29:   CMTGEN.CMTGenEbiCtrlEnable
30:   CMTGEN.CMTGenParamGetDefault
31:
32:;Gen variables
33:   CMTGEN._gbyCMTGenNumberOfBoards
34:   CMTGEN._gbyCMTGenCurBoardNu
35:   CMTGEN.__gCMTGenBoardType
36:   CMTGEN.__gwCMTGenMajorBoardVersion
37:   CMTGEN.__gwCMTGenMinorBoardVersion
38:
39:;Rgb functions
40:   CMTRGB.CMTRgbInSetReference
41:   CMTRGB.CMTRgbInSetLevel
42:   CMTRGB.CMTRgbInSetSync
43:   CMTRGB.CMTRgbOutSetMode
44:   CMTRGB.CMTRgbOutSetLut
45:   CMTRGB.CMTRgbOutSelectLut
46:   CMTRGB.CMTRgbInSetStandard
47:   CMTRGB.CMTRgbParamGetDefault
48:
49:;Vin functions
50:   CMTVIN.CMTVinChromaSetHue
51:   CMTVIN.CMTVinChromaSetSat
52:   CMTVIN.CMTVinInSelect
53:   CMTVIN.CMTVinInSetStandard
54:   CMTVIN.CMTVinInSetType
55:   CMTVIN.CMTVinInSetGenLockSrc
56:   CMTVIN.CMTVinLumSetAGC
57:   CMTVIN.CMTVinLumSetBrightness
58:   CMTVIN.CMTVinLumSetClamp
59:   CMTVIN.CMTVinLumSetContrast
60:   CMTVIN.CMTVinLumSetCoring
61:   CMTVIN.CMTVinLumSetGain
62:   CMTVIN.CMTVinLumSetSharp
63:   CMTVIN.CMTVinParamGetDefault
64:
65:;Vip functions
66:   CMTVIP.CMTVipGrabSetDestPos
67:   CMTVIP.CMTVipGrabSetSourcePos
68:   CMTVIP.CMTVipGrabSetRate
```

```
 69:    CMTVIP.CMTVipGrabSetMode
 70:    CMTVIP.CMTVipGrabGetDestPos
 71:    CMTVIP.CMTVipGrabSingle
 72:    CMTVIP.CMTVipGrabGetOffScreenSize
 73:    CMTVIP.CMTVipGrabGetOffScreenAddress
 74:    CMTVIP.CMTVipGrabStartInterrupts
 75:    CMTVIP.CMTVipGrabStopInterrupts
 76:    CMTVIP.CMTVipGrabRegisterIntFunction
 77:    CMTVIP.CMTVipKeyFillRect
 78:    CMTVIP.CMTVipKeyFillRow
 79:    CMTVIP.CMTVipKeyFillColumn
 80:    CMTVIP.CMTVipKeySetMode
 81:    CMTVIP.CMTVipOverlayVideo
 82:    CMTVIP.CMTVipPictureLoad
 83:    CMTVIP.CMTVipInSelect
 84:    CMTVIP.CMTVipDisplaySetSt
 85:    CMTVIP.CMTVipGrabSetOffScreenPos
 86:    CMTVIP.CMTVipDisplayFromOffScreen
 87:    CMTVIP.CMTVipParamGetDefault
 88:    CMTVIP.CMTVipOverlayVideoWin
 89:
 90:;Vpe functions
 91:    CMTVPE.CMTVpeSetEnableSt
 92:    CMTVPE.CMTVpeSetAntiFlickerSt
 93:    CMTVPE.CMTVpeSetStandard
 94:    CMTVPE.CMTVpeSetUnderscan
 95:    CMTVPE.CMTVpeDencSetChromBand
 96:    CMTVPE.CMTVpeDencSetStd
 97:    CMTVPE.CMTVpeDencSetGenLockSrc
 98:    CMTVPE.CMTVpeDencLutLoad
 99:    CMTVPE.CMTVpeDencLutSetByPass
100:    CMTVPE.CMTVpeDencSetScPhase
101:    CMTVPE.CMTVpeDencSetHsPhase
102:    CMTVPE.CMTVpeParamGetDefault
103:
104:    CmtVip._Mrv2VipWaitForVBlank
105:    CmtGen.CmtGenBlockRead
106:    CmtGen.CmtGenBlockWrite
107:
108:
109:
```