K
フ-10
A4S

プログラム設計書

SC

プログラム設計書

システムの構造と構成

1. システムの構造および流れ図

本システムの基本的な考え方は、画像処理により樹木の葉より抽出された特徴パラメータを、データベースとしてコンピュータ内に保存された各樹種に関する特徴パラメータを用いて検索し樹種を特定するというものである。従って本システムは大別して、(1)特徴抽出、(2)データベース、(3)検索 の３つの部分から成る。基本的な構成および流れを図 3.1 に示す。

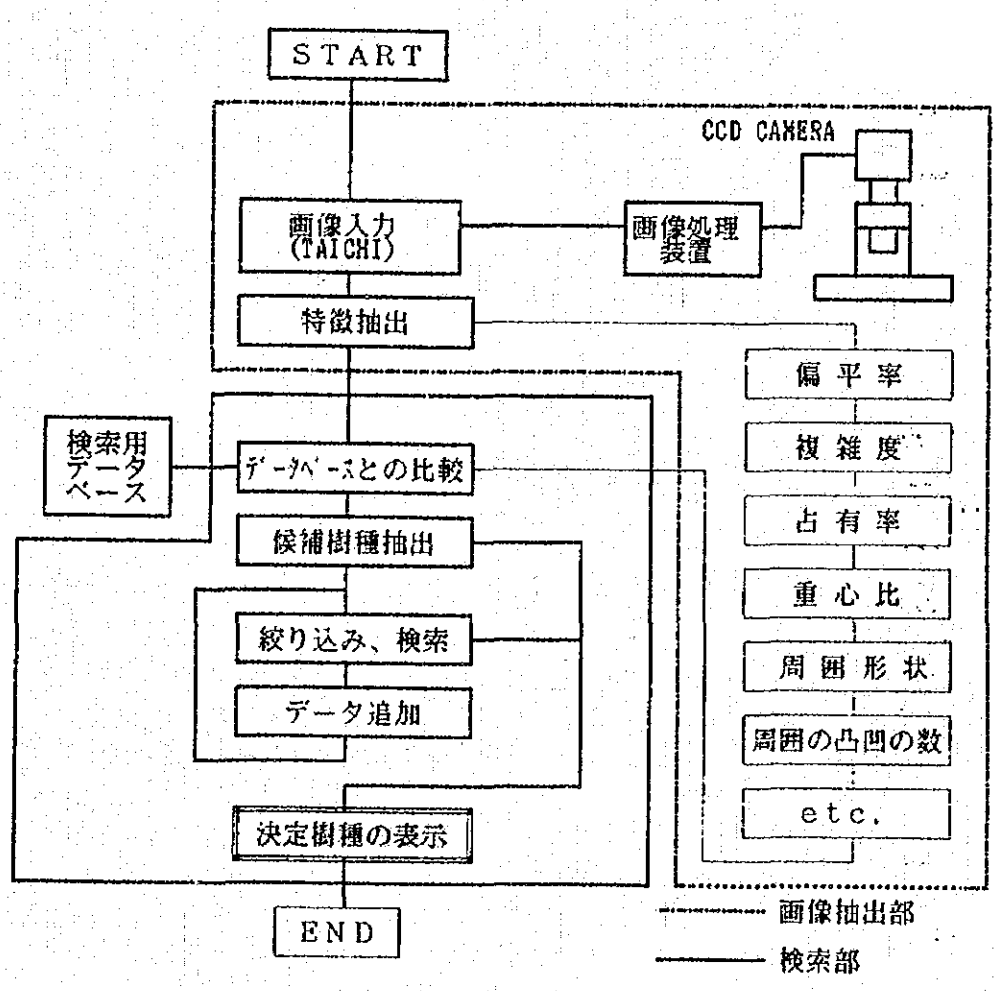


図３．１ 自動検索システムの基本的な流れ

2　システムの構成

2．1　システムの構成図

本システムの構成を図 3.2 に示す。

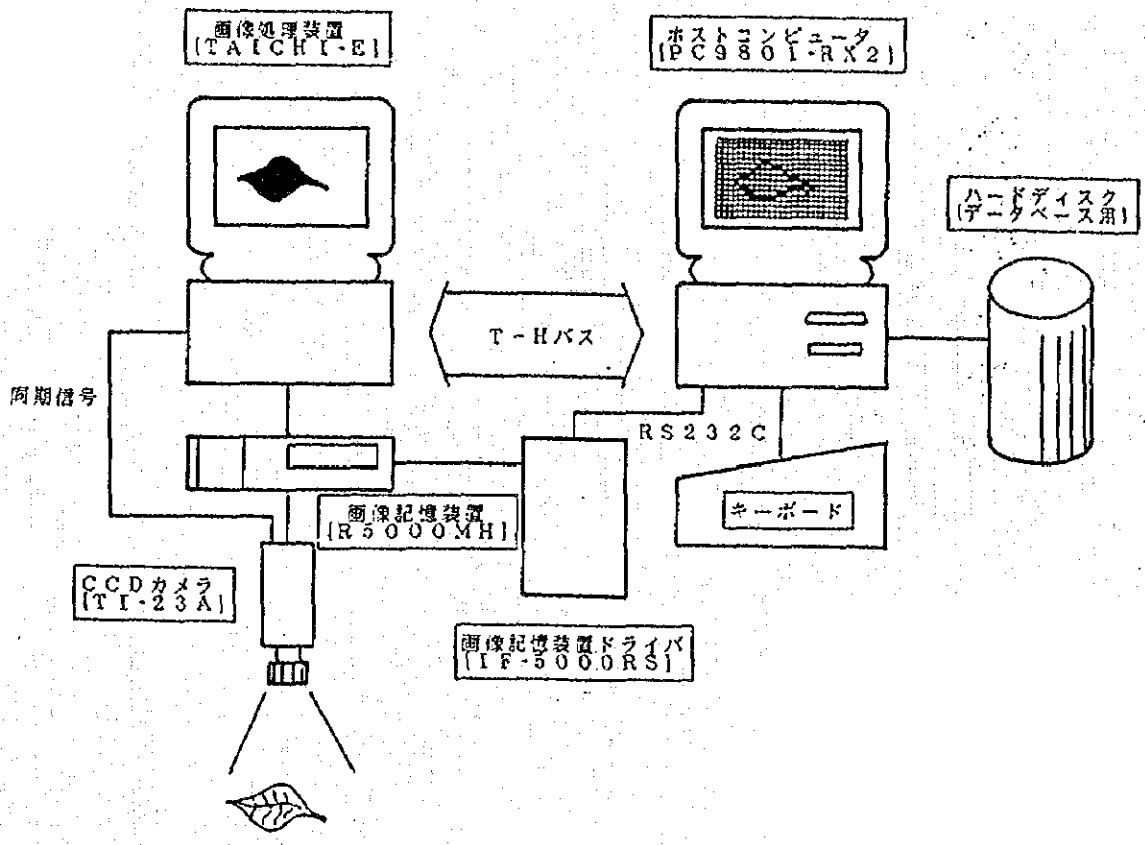


図３．２　実験装置の構成図

２．３　特徴パラメータの抽出手順

本システムでは、図 3.2 に示した自動検索システムを用いて特徴パラメータの抽出を行なった。

樹葉の実画像から特徴パラメータ抽出し、その特徴パラメータをデータベースへ保存する手順を図 4.7 に示す。図中、画像記憶装置は、システムの構築、データベース作成および装置の信頼性を検討する上で必要な対象となる樹葉の実画像データを保存するもので、本システムの開発途中で必要な装置である。したがって、完成された自動検索システムにおいては不要となる。また、図中の「データベースの構築」の部分は、データベースを作成する際に用いられ、検索作業では不要である。

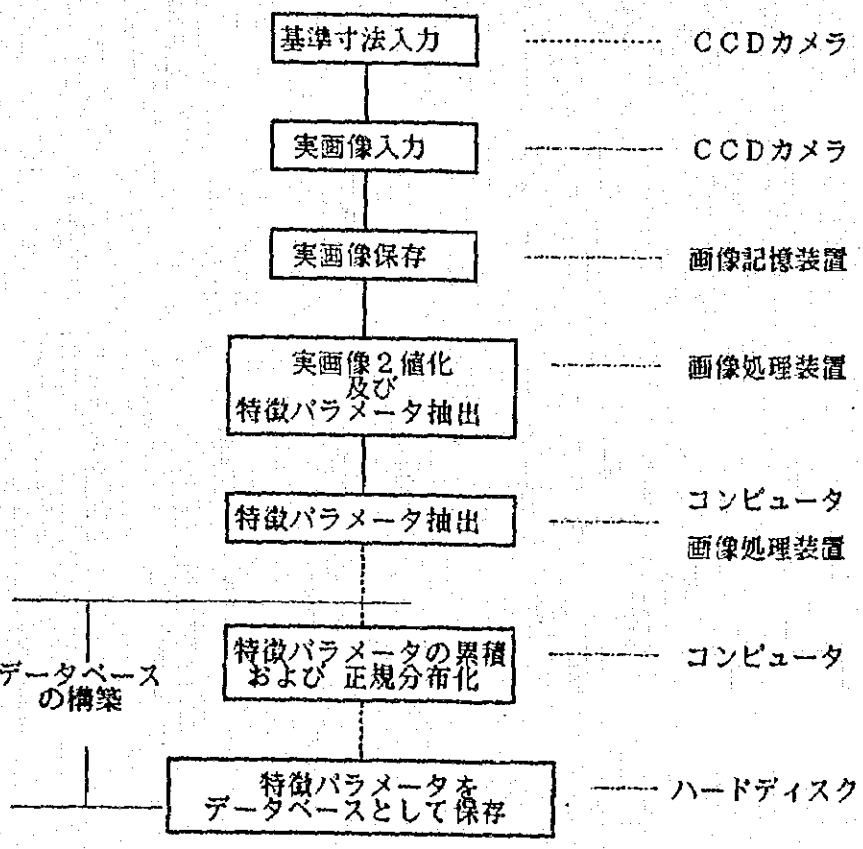


図４．７　特徴パラメータ抽出の手順

３．２　データベースの内容

母集団の各特徴パラメータの分布が正規分布で近似できることから、データベースには平均値と標準偏差を記録すれば良い。表 4.2 にデータベースに記録される各特徴パラメータに関するデータの構造を示す。

ここで G_i ($=G_1 \sim G_m$)は樹種を、P_j ($=P_1 \sim P_n$)は特徴パラメータを示す。

表４．２　比較される分布のデータ構造

	P_1	P_2	…	P_j	…	P_n
G_1	$\bar{X}_{11}, \sigma_{11}$	$\bar{X}_{12}, \sigma_{12}$	…	…	…	…
G_2	…	…	…	…	…	…
	…	…	…	…	…	…
G_i	…	…	…	$\bar{X}_{ij}, \sigma_{ij}$	…	…
	…	…	…	…	…	…
G_m	…	…	…	…	…	$\bar{X}_{mn}, \sigma_{mn}$

４．４　特徴パラメータの分布の評価

本システムの主要部分であるデータベースには、母集団を正確に表現できている平均値$\bar{X}$と標準偏差σが必要である。母集団の平均値と標準偏差は、その母集団を構成する標本の数により変化することは明らかである。図 4.10 に母集団を構成する標本の数の変化に伴う平均値$\bar{X}$および標準偏差σの変化の様子の一例を示す。図から明らかな通り、標本の数が増すと平均値と標準偏差は振動しながらある一定値に近づいていく。したがって、幾つの標本から平均値および標準偏差を求めれば母集団が充分にその樹種を表現していると言えるかを明らかにする必要がある。

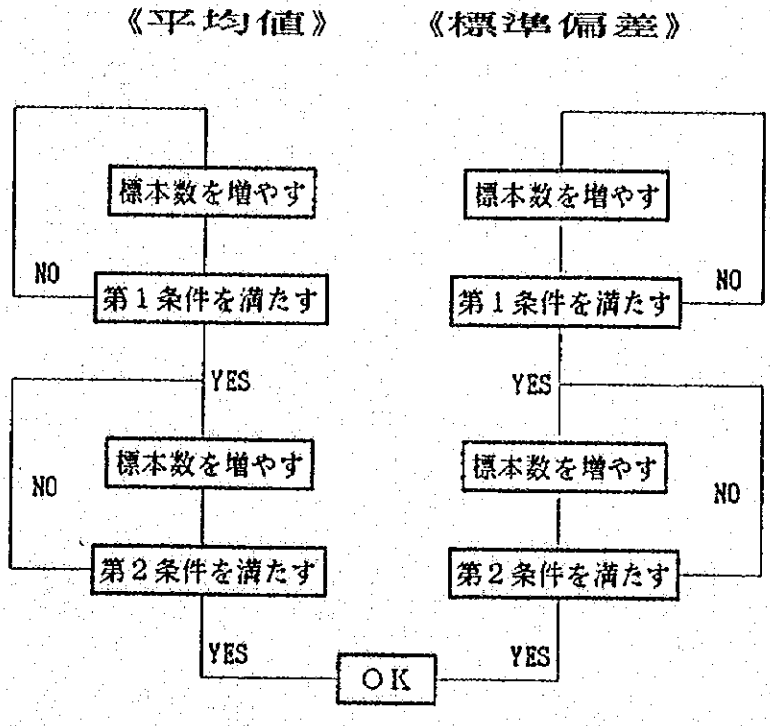


図４．１１　特徴パラメータの分布の収束判断の流れ図

４．５　分布をもつ特徴からの検索方法

本システムにおける検索方法は、標本となる樹葉から得られる各特徴パラメータの値が、データベースに記録された各樹種とどの程度近いかを定量的に算出し、一番近いと算出された樹種をその標本の樹種であると判定するものである。ここで、データベースにある樹種との近似度の定量的計算には、マハラノビス距離を用いる。

４．５．１　マハラノビス距離

マハラノビス距離とは、標本から得られたある特徴パラメータの値がその特徴に対応する母集団の分布どの位置に相対的に相当するかを示す無次元の値である。すなわち、標本のある特徴パラメータの値のマハラノビス距離は、式(4.10)で与えられる。

システムテスト報告書

性能確認試験の方法

1　検索対象樹種

パプア・ニューギニアにある熱帯雨林の伐採現場から採取された約50種類の樹葉の中から10種類の樹葉を選び本システムの性能確認試験を行った。表 5.1 に採用した樹種を示す。また、図 5.1 にそれぞれの代表的と思われる樹葉を示す。

本報告書の範囲では、各樹種を表および図中に示すG～Pのアルファベットで示した。

表５．１　樹種A～Pの樹種名

呼名	樹　種	科　名
G	MANILTOA PLURIJUGA	LEGUMINOSAE
H	TIMONIUS TIMON	RUBIACEAE
I	PTEROCARPUS INDICUS	LEGUMINOSAE
J	CORDIA DICHOTOMA	BORAGINACEAE
K	INTSIA BIJUGA	LEGUMINOSAE
L	MANILTOA PSILOGYNE	LEGUMINOSAE
M	ALBIZIA PROCERA	LEGUMINOSAE
N	MANGIFERA MINOR	ANACARDIACEAE
O	PIMELODENDRON AMBOINICUM	EUPHORBIACEAE
P	ALSTONIA SCHOLARIS	APOCYNACEAE

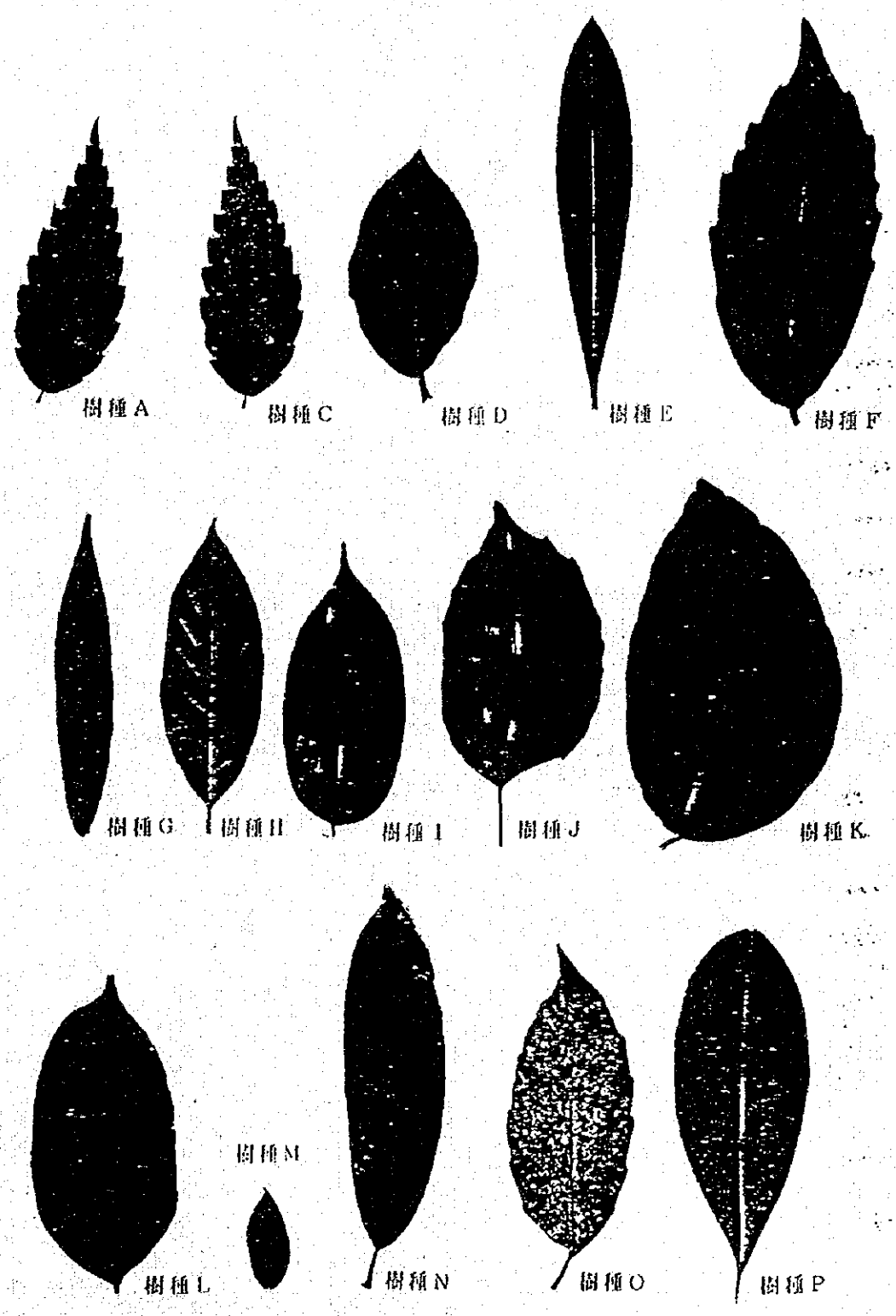


図5．1　検索の対象とした樹葉

2　撮影倍率の影響

標本の実画像を撮影しそれから各特徴パラメータの値を算出する場合、次元をもつパラメータ（長さＬ，幅Ｗ，周囲長ｌ，面積Ｓ・・・）や頂点数のようにしきい値を設定して求めるパラメータでは、撮影倍率が異なることにより得られる値が変化してしまう。したがって、撮影倍率の違いによる測定値の変化を自動的に修正する必要がある。

2．1　基準長さ

特徴パラメータのうち次元をもつ 長さＬ，幅Ｗ，周囲長ｌ，面積Ｓ については、ＣＣＤカメラの位置やレンズ系を変化した場合の実長と画素数の関係を補正するために、必要に応じ撮影前に基準長さを入力し、画素数と実長の関係を決めるようにした。

2．2　頂点数に関する補正

(1) 補正の手順 ····· 頂点数は、報告者らにより開発された偏角差分関数により頂点を抽出法（4.2.2.(3)参照）を用いた。この方法は、画素の配列状況から角度を求め一定の角度以下を頂点と認識する様にしてある。したがって、１画素の長さが撮影倍率により異なると画面上で同一の角度でも実際は異なった角度、あるいは異なった大きさの突起となってしまう。

角度を求める際に幾つの画素間で角度を求めるかをしきい値と呼び、撮影倍率としきい値の関係を予め調べる必要がある。

本システムでは、図 5.2 に示す手順により、撮影倍率（例えば、ＣＣＤカメラの位置）を変化した場合に、実際の目測により測定した頂点数と本システムで計測した頂点数が一致するしきい値を求めた。ただし、角度は 20度に固定した。

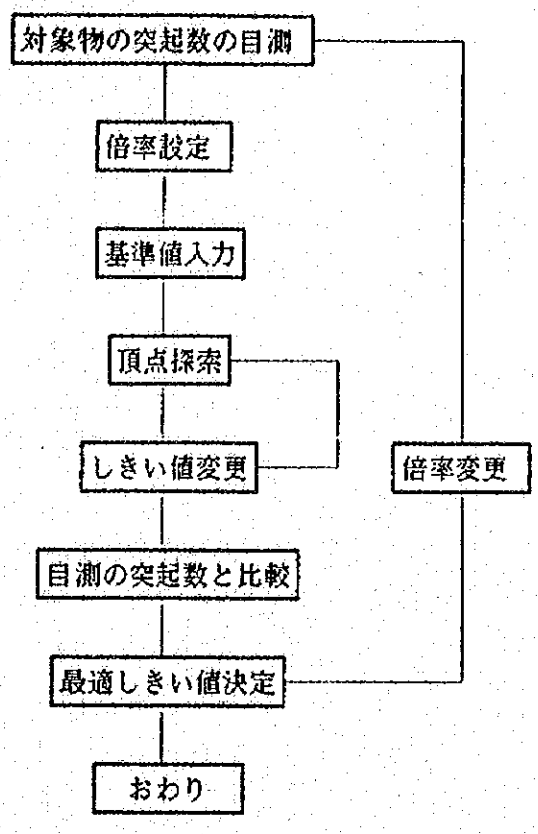


図５．２　撮影倍率の変化にともなうしきい値の決定手順

(2) 頂点数計測の際の撮影倍率としきい値の関係

偏角差分関数により頂点を抽出する際のしきい値（角度、および画素間隔）を変化させ、実際の目測により測定した頂点数と一致する点のしきい値を求めた。すなわち、撮影倍率を変化することは、真の単位長さ当りの画素数（１画素が実寸法で何画素に相当するか）が変化することを意味する。したがって、目測により得られる頂点数と計測により得られる頂点数が一致するためのしきい値を撮影倍率を変化して求めた。但し、角度は20度に固定で行なった。

図 5.3 に検定結果である撮影倍率（単位長さ当りの画素数）としきい値の関係を示す。

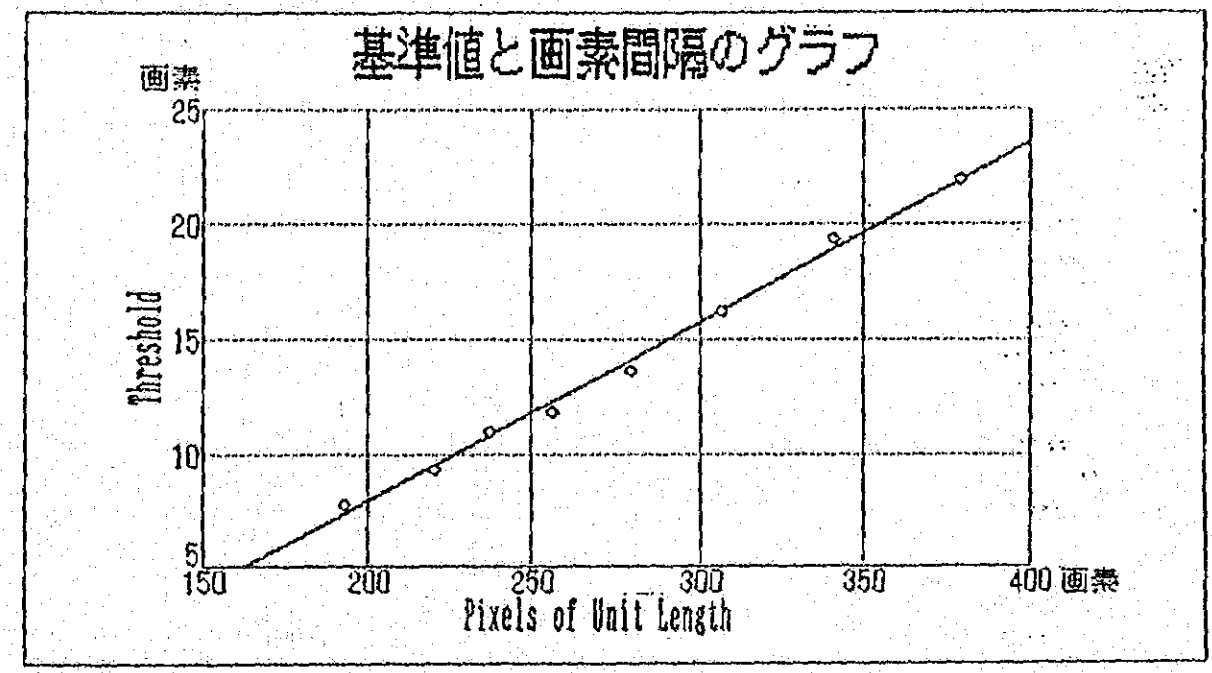


図５．３　基準値としきい値の関係

図からも明らかなように、撮影倍率としきい値は直線関係にある。したがって、本システムでは、カメラの位置やレンズ系を変化した際の撮影倍率を前項で示した基準寸法を用いて算出し、さらに図 5.3 の関係からしきい値の画素数を決定して頂点数を算出するロジックを加えてある。

3　シミュレーションによる性能の確認方法

本システムの性能を確認するために、パプア・ニューギニアで採取された１０種類の樹葉を用いて、繰り返し検索テストを行ない認識率を求めた。

テストに使用した樹種は、表 5.1 で示した樹種Ｇ～樹種Ｐである。

３．１　被検索樹葉

本システムの性能を確認するためには、データベースを構築した樹葉群とは別に同程度の枚数の樹葉群を用いて実際に検索試験を行なうことが望ましい。しかし、ここではデータベースを構成する樹葉の中から樹葉を数枚分のデータを無作為に抽出し、これらを未知の樹葉として検索を行なった。この作業を認識率が一定値に収束するまで繰り返し行なった。本試験でこの方法を採用した理由は以下のとおりである。

(1) 対象とした樹葉は海外で採取されたものであり、充分な数を確保できない。

(2) データベースの各特徴パラメータの分布を作成する際に、充分な枚数の樹葉を用いており、数枚抽出しても分布は変化しない。

３．２　性能の確認方法

本システムの性能を確認するために行なったシミュレーション・テストの方法は、データベースのとして記録されているそれぞれの樹種について、その中から１枚から１０枚のデータをランダムに抽出し、これを被検索樹葉として検索を行なう。この作業を平均認識率が一定値に収束するまで繰り返し行なった。

具体的には、次の２点のテストを行なった。

(1) 認識率の確認 …… データベースに記録されている全ての特徴パラメータを用いて、認識率を求めた。また同時に、判定を行なった際のマハラノビス距離を記録し、その分布図を求めた。

(2) 特徴パラメータの有効性の確認 …… 各特徴パラメータそれぞれ単独で用いて検索を行ない認識率を求めた。

性能確認試験の結果

1　被検索樹葉の枚数の影響

1．1　被検索樹葉が1枚の場合の認識率

標本として樹葉を1枚だけ用いて検索を行なった場合の認識率を図 6.1 に示す。

図から明らかなように、樹種により認識率は異なりその値は５５～１００％の間でばらつきがある。現状で記録されている樹種およびこれらの特徴パラメータでは、樹種Mが最も判別しやすく樹種Ｉが最もしにくい。これは、樹種Ｉの分布が他の何れかの樹種の分布と比較的にており、誤認識するためである。すなわち、本システムの検索方法では、パラメータの分布の位置関係が関与しており、さらに認識率を向上させるためには、誤って認識しやすい樹種を予め調べ、これらを分離することが可能な特徴を中心に再検索するなどのチェック機能などを加えてる必要がある

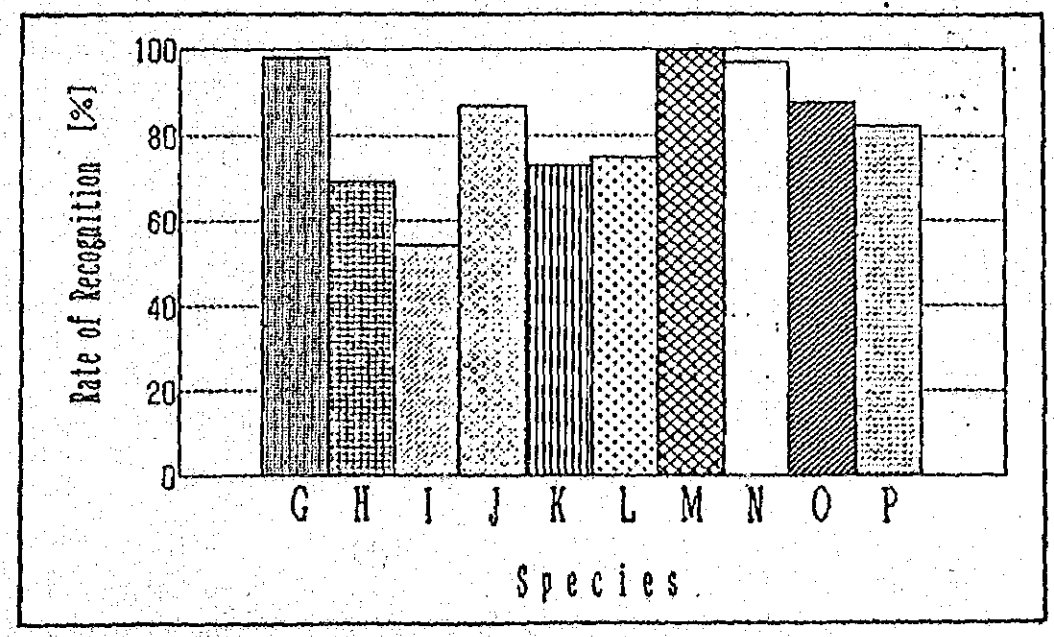


図６．１　標本樹葉１枚の場合の認識率（全特徴パラメータ）

１，２　被検索樹葉の枚数と認識率の関係

被検索樹葉の枚数を１～１０枚に変化させた場合の認識率を図 6.2 に示す。この際、被検索樹葉の樹種は未知であるがそれらは同一の樹種である。

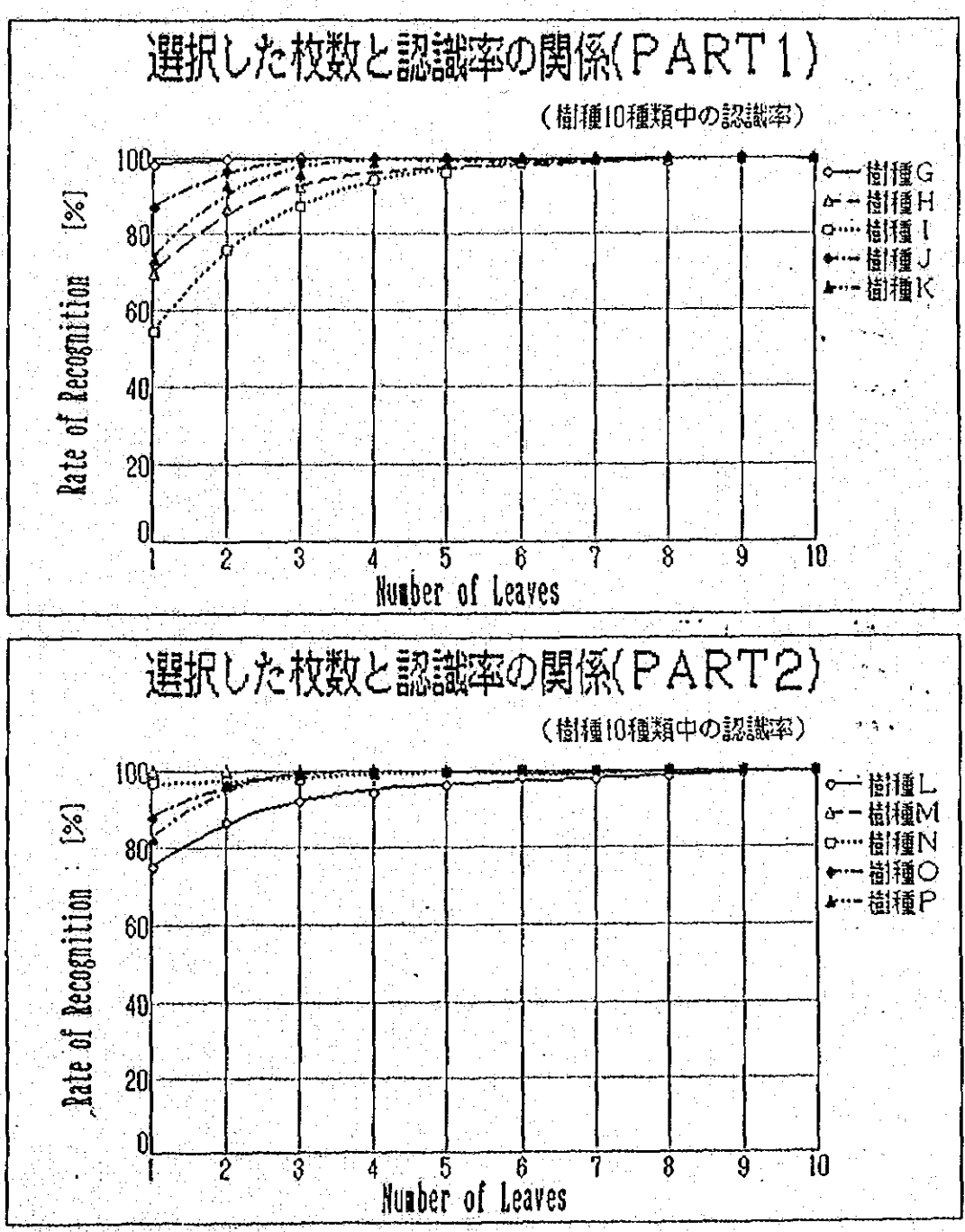


図６，２　標本樹葉枚数と認識率の関係（全特徴パラメータ）

いずれの樹種でも標本の枚数を増やすと認識率は増加する。その増加率は樹種により異なり、１枚で最も認識率の低い樹種Ｉでは９枚でほぼ１００％に達するが、樹種Ｉに比べ１枚ではかなり認識率の高い樹種Ｌは、１０枚の標本を使わないと１００％に達しない。しかし何れの場合でも７枚程度以上でほぼ１００％に近い認識率を得ることができる。

６．１．３　標本数と認識率の関係

前項で示した通り、標本数を増やすと認識率は増加する。この理由は、以下の通りである。

ここで、任意の樹種の任意の特徴パラメータを考える。母集団の特徴パラメータを$N(\mu, \sigma^2)$の正規分布とすると、その確率密度関数は式(6.1)で表わされる。

$$f(X) = \frac{1}{(2\pi)^{0.5}\sigma} \exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right) \quad \cdots\cdots\cdots\cdots\cdots\cdots (6.1)$$

ここで、母集団からｎ個の標本 $X_1, X_2, \cdots, X_i, \cdots, X_n$ をランダムに抽出し、それぞれのある特徴パラメータの値を $X_1, X_2, \cdots, X_i, \cdots, X_n$ とすると、その特徴パラメータの平均値$\overline{X}$は式(6.2)となる。

$$\overline{X} = \sum_{i=1}^{n} \frac{X_i}{n} \quad \cdots\cdots\cdots\cdots\cdots\cdots (6.2)$$

ここで、複数枚の樹葉を繰り返し選択し、ある特徴パラメータを抽出すると、選択された樹葉の特徴パラメータの平均値$\overline{X}$の分布は新たな確率変数となり、その確率密度関数は、式(6.3)で示される。

$$f(\overline{X}) = \frac{n^{0.5}}{(2\pi)^{0.5}\sigma} \exp\left(-\frac{n(\overline{x}-\mu)^2}{2\sigma^2}\right) \quad \cdots\cdots\cdots\cdots\cdots\cdots (6.3)$$

すなわち、特徴パラメータの平均値$\overline{X}$の分布は、$N(\mu, \sigma^2/n)$で表現される正規分布となり、標本数ｎが増加するにしたがい広がりの小さな分布、すなわち

平均値に近い値となる確率が高くなる。したがって、これらの値から求められるマハラノビス距離は、小さくなり認識率が増加する。

図 6.3 に、標本数ｎを変化した場合の正規分布の形の変化を示す。また、図 6.4 に本システムの場合の一例として標本の枚数を変化した場合の頂点率の平均値の分布の違いを示す。

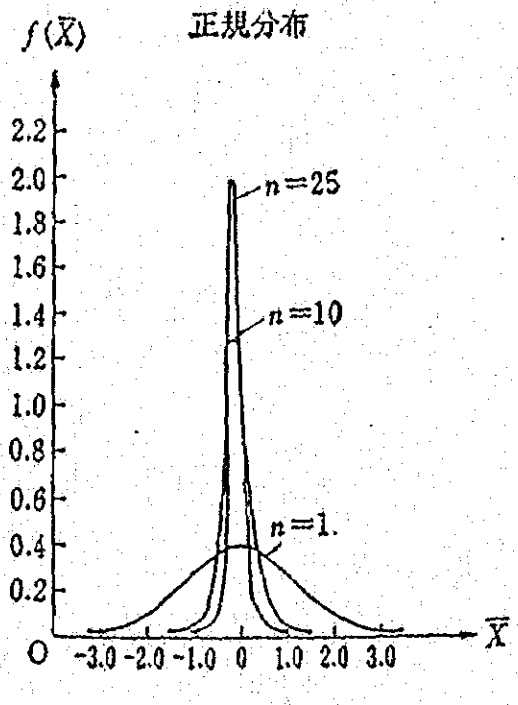


図６．３　正規分布の先鋭化

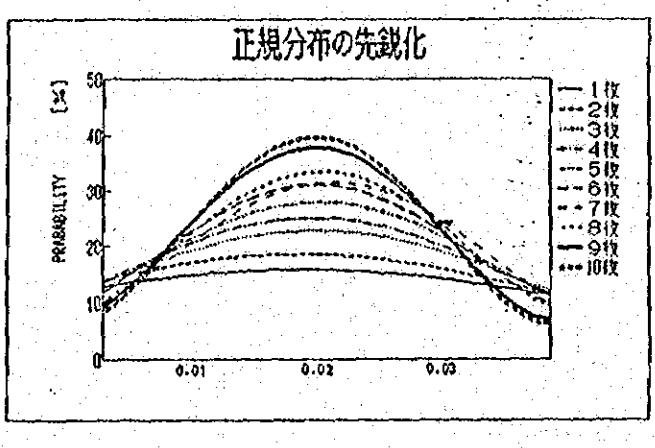
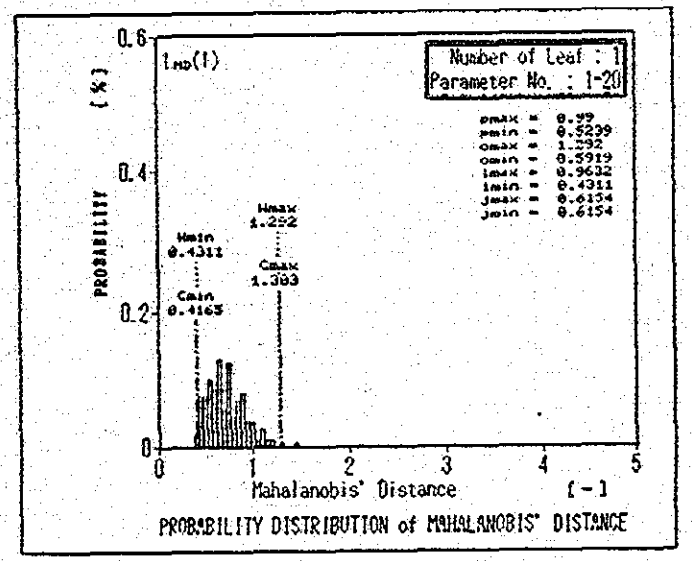
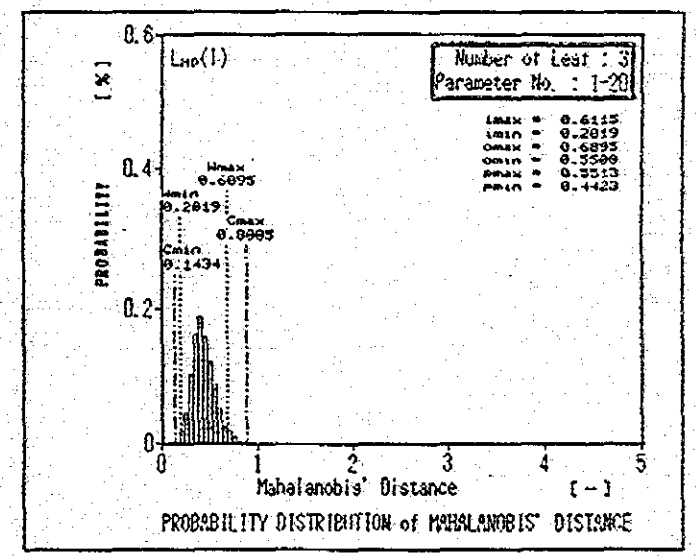


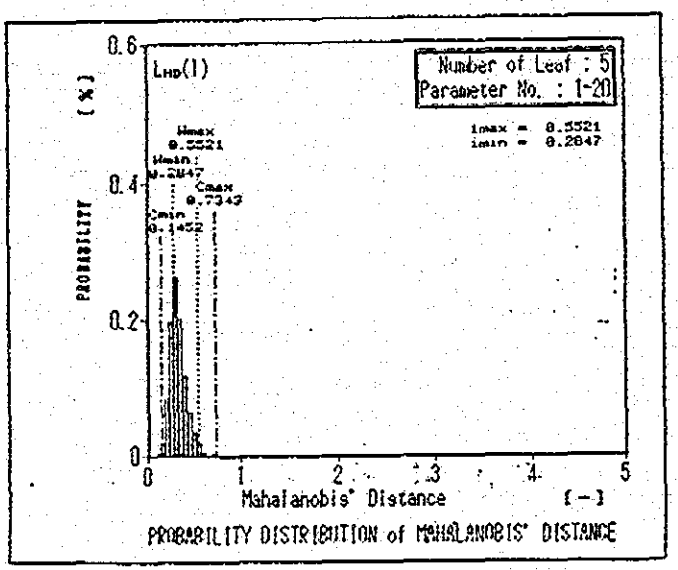
図６．４　枚数による分布の変化の一例

１．４　マハラノビス距離

図 6.5 および図 6.6 に樹種Ｌを検索した場合の全マハラノビス距離の分布の一例を示す(その他の樹種の内、樹種Ｉに関する全マハラノビス距離の分布の変化を APPENDIX 1 に示す)。各図中の４枚は、標本数を１，３，５，１０と変化した場合である。図 6.5 は、樹種Ｌ（未知の樹葉であるが実際は樹種Ｌ）を検索した場合に、樹種Ｌの母集団のデータを用いて算出されたマハラノビス距離の分布を、また図 6.6 は、樹種Ｉの母集団のデータを用いて算出されたマハラノビス距離の分布を示す。







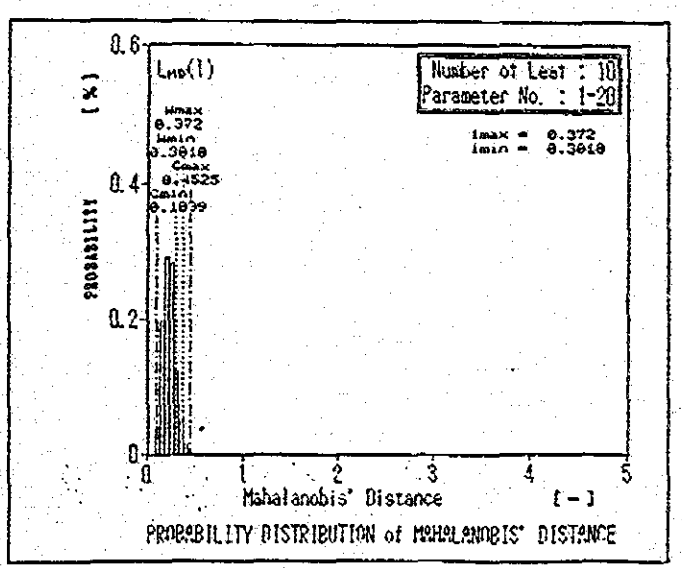


図6．5　標本樹葉の樹種Ｌの時樹種Ｌと認識したマハラノビス距離の分布

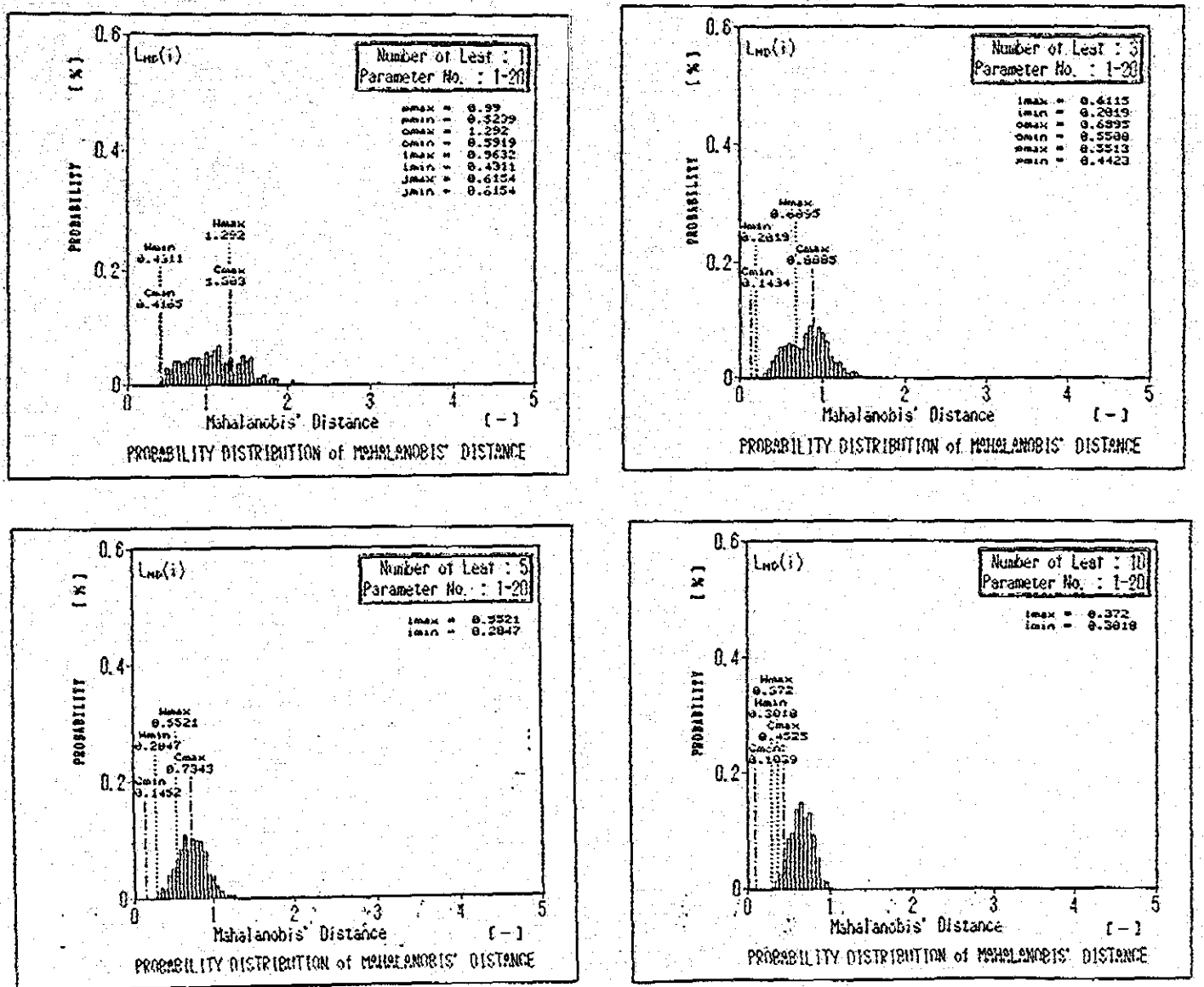


図6．6　標本樹葉の樹種Ｌの時樹種Ｉと誤認識したマハラノビス距離の分布

ソースライブラリ

```
/*##########################################################################*/
/*      PROGRAMNAME = NINSIKI8.C                                            */
/*      FUNCTION = 多種のパラメタで、データベースを構成する個々の変数の自種   */
/*                 ・他種へのマハラノビス距離を計算する。パラメータは自ら選   */
/*                 択する事もできる。そして、それぞれの変数（個体）の樹種を   */
/*                 判定し認識率を求める。また、枚数を指定する事により母集団   */
/*                 から複数枚ランダムに抽出し、その複数枚の各パラメタ平均値   */
/*                 よりマハラノビス距離を求め判定を行い認識率を求める。       */
/*                 但し、複数枚で判定を行う場合は、それらを母集団から間引き   */
/*                 しする事は行わない。                                       */
/*                 1989年度最終バージョン                                     */
/*                                                                          */
/*             << THE TEAM OF LEAF >>  '89-12-吉日      PRODUCED by MAKI    */
/*##########################################################################*/
/**************************************************************************/
/************* DEFINE  GROVAL VARIABLE ************************************/
    #define MAX 250

    double  VARIABLE[ MAX], AVERAGE[ 260], ROOTMS[ 260],RVARIABLE[ MAX];
    double RANRMS[ MAX],RANAVE[ MAX];
    huge double  DDK[ 60][ 250]; /**** 選択した葉の各樹種へのdk 64 kbyte 超えないように*/
    int     RECORDNUM;
    long    aryrand[ 2600];
    int     HMCHOICE;
    int     LIMITNUM;
    int     r, n;
    int     FILNUM[1];          /**** 計算対象となるファイルの数 ****************/
    int     COUNTPAR[ 1];       /**** parameterの種類数 **************************/
    int     COUNTLK[ 1];        /**** 樹種の種類数 ******************************/
    int     COUNTSP[ 1];        /**** 選択したパラメタと同じパラメタを ***********/
                                /**** 持つファイルの数 **************************/
    int     COUNTSLK[ 1];       /**** 選択した樹種と同じ樹種を *******************/
                                /**** 持つファイルの数 **************************/
    int     COUNT[ 200];         /**** DDKに対応したカウント *********************/
    int     COUNT1[ 200];        /**** DDKに対応したカウント *********************/
/*
    unsigned int SEED;          /**** 乱数発生用のシード ************************/
*/
    char    *s;
    char    LKIND[ 260][20];
    char    PARAMETER[ 260][20];
    char    RAWNAME[ 260][ 20];
    unsigned char CLOCK[ 8];
    /**************************************************************************/
    extern  unsigned _STACK = 30000;

    #include    "stdlib.h"
    #include    "stdio.h"
    #include    "string.h"
    #include    "math.h"
```

```
    #include    "dos86.h"
    #include    "maki.h"
    #include    "ctool.h"
    #include    "opefile.h"
    #include    "rname.h"
    #include    "nordisb.h"


main()
{

/###### DEFINE ##########################################################*/
    FILE *fi,*fp;
    FILE *fr,*fs,*ft;           /**** 結果のファイル ***************************/
    FILE *fa;
    double x0, ave , rms , z0, gaus, ingaus , dk ;
    int num1[ 1];               /**** 計算対象となるファイルをオープンした順 ****/
                                /**** fil_call()の引数になっている **************/
    int count2[ 1];             /**** 判断２におけるIF文を通った回数 ************/
    int count3[ 250];           /**** 樹種Aとし判定した回数 *********************/
    int count4[ 1];
    int count5;
    int count6;
    int count7;
    int count8;
    int count9;
    int count10;
    int count11;
    int count12;
    int counter1[ 1];
    int counter2[ 1];
    int counter3;
    int dummy;                  /**** バブルソートの為のダミー ******************/
    int slprcnt[ 1];
    int sllkcnt[ 1];
    int looplmt;
    int draw1[ 260];
    int draw2[ 260];
    int flag1[ 1];
    int  i, j, k,l, m, com,com1;
    int  x1;
    int arynumpar[ 260];        /**** 選択したパラメタのファイル順を表わす *******/
    int arynumlk[ 260];         /**** 選択した樹種のファイル順を表わす ***********/
/*
    int loopcnt;                /**** ループのカウントを代入するための変数 *******/
*/
    char judge[ 3];             /**** プログラムの終了を判断するための文字列 *****/
    char judge1[ 3];            /**** 結果を保存するかを判断するための文字列 *****/
    char judge2[ 3];            /**** 結果を保存するかを判断するための文字列 *****/
```

```
    char name1[ 100];       /**** 結果用のファイル名 **********************/
    char name2[ 100];       /**** 結果用のファイル名 **********************/
    char name3[ 100];       /**** 結果用のファイル名 **********************/
    char name4[ 100];       /**** 結果用のファイル名 **********************/
    char node3[ 10];        /**** 結果用のファイル名 **********************/
    char dumychar[ 10];
    char fname[ 40][ 25];  /**** 計算対象となるファイルの名前 ***************/
    char fname1[ 260][ 25]; /**** 計算対象となるファイルの名前 ***************/
    char name[ 25];         /**** fname[ ][ ]を代入するためのポインタ変数 ****/
    char parname[ 25];      /**** parameter file nameを代入するポインタ変数 **/
    char searpar[ 260][ 20]; /**** パラメタのサーチ ***************************/
    char searlk[ 260][ 20];  /**** 樹種のサーチ *****************************/
    char selpar[ 25];       /**** パラメタを選ぶための配列 *******************/
    char sellk[ 25];        /**** 樹種を選択するための配列 *******************/
    char selectpr[ 25][ 20];/**** パラメタを選択するための配列 ***************/
    char selstr[ 1][20];     /**** 文字列を選ぶための変数 *********************/
/*    char lkind1[ 20][ 20];/**** 真のLKIND[][]を代入するための配列変数 ******/
     char lkind2[ 100][ 20];  /**** 比較される樹種 ************************/*/
    char hlkind[ 1][ 20];   /**** 前回の樹種 *******************************/
    char *s1;
    char *s2;
    char par[ 20];
    double hdk[1];          /**** 前回のマハラノビス距離 *********************/
    double ninritu;         /**** 認識率 ***********************************/
    double ninritu1[ 20];   /**** 樹種の認識率 *****************************/
    double ninritu2[ 50];   /**** 過去の認識率 *************************************/
    double shuusoku[ 3];        /**** 収束条件 ********************************/
    double ranms;
    double ave1;
    double roms;
/*
    double average[ 1];     /**** データから計算した平均値 *******************/
    double msquar[ 1];      /**** データから計算したＭＳ *********************/
    double rmsquar[ 1];     /**** データから計算したＲＭＳ *******************/
*/
/*##########################################################################*/
/*######### READ PARAMETER-FILE ##############################################*/
    printf("¥x1b[>1h");
    printf("¥x1b[>5h");

    if((fi = fopene("testdat4.dat", "r","")) == NULL)
    quit(" Can't Open File ");
    fread_name( fi,fname);

    file_close(fi);

    counter2[ 0] = 0;

    for(i=0 ; i < FILNUM[ 0]; ++i)
    {
```

```
        stccpy(name,fname[ i],strlen(fname[ i]) + 1);

        printf("%d %s ¥n",i,name);

        if((fp = fopene(name, "r","")) == NULL)
        quit(" Can't Open File ");

        num1[ 0] = i;   /*  LKIND & PARAMETER の中の配列番号 */

        fil_cal5( fp,counter2,fname,fname1,num1);

        file_close(fp);
    }

    FILNUM[ 0] = counter2[ 0];

/*######## パラメタ名  &  樹種名  の調査  ##################################*/

    str_search(searpar,PARAMETER,COUNTPAR);  /** パラメタのサーチ **/

    str_search(searlk,LKIND,COUNTLK);        /** 樹種のサーチ **/


/*######## SELECT AUTO or MANUAL ###########################################*/

    t_cls();

    t_loc(5,2);
    printf("================================================== ¥n");
    t_loc(5,3);
    printf("select MANUAL  <<operate yourself >>  ====> hit < 0 > ¥n");
    t_loc(5,4);
    printf("select AUTO1 << use all parameter >>  ====> hit < 1 > ¥n");
    t_loc(5,5);
    printf("select AUTO2 << use each parameter >> ====> hit < 2 > ¥n");
    t_loc(5,6);
    printf("select AUTO3 << use only m's dist. >> ====> hit < 3 > ¥n");
    t_loc(5,7);
    printf("================================================== ¥n");
    t_loc(5,8);
    printf("HIT NO. =====>");

    scanf("%d",flag1);

    if( (flag1[ 0] == 1) || (flag1[ 0] ==2) || (flag1[ 0] ==3))
    {
        t_loc(5,9);
           printf("HOW MANY LEAVES DO YOU CHOICE FOR LIMIT[1-10]? ===>");
        scanf(" %d",&LIMITNUM);
```

```
    }

/*#### MAIN LOOP ##################################################################*/

    count7     = 0;
    count9     = 0;
    judge1[ 0] = 'x';
    judge2[ 0] = 'x';

while(1)
{
/*######### GRAPHIC & TEXT SCREEN INITIALIZE ######################################*/
    g_init();
    g_screen(3,0,0,1);

/*######### INITIALIZE  VARIABLE ##################################################*/

    for(i=0;i<250;++i)
    VARIABLE[ i]  = 0;

    RECORDNUM     = 0;
    r             = 0;
    n             = 0;
    COUNTSP[ 0]   = 0;
    COUNTSLK[ 0]  = 0;
    counter1[ 0]  = 0;
    count5        = 0;
    count6        = 0;
    count10       = 0;
    judge[ 0]     = 'x';
    count4[ 0] = 0;

/*################################################################################*/
/*######### OPEN RESULT FILE ######################################################*/
    if(flag1[ 0] == 0 || flag1[ 0] == 3)
    {

        while( ((judge1[0] != 'y') && (judge1[0] != 'n')) &&
             ((judge1[0] != 'Y') && (judge1[0] != 'N')) && (count9 == 0) )
        {
        t_loc(5 ,10);
            printf("DO YOU SAVE  RATE of RECOGNIZATION RESULT ?");
        t_loc(5 ,11);
            printf("<< YES ===> hit y / NO ===> hit n >> ");
            scanf  (" %s", judge1);
        }

        while( ((judge2[0] != 'y') && (judge2[0] != 'n')) &&
             ((judge2[0] != 'Y') && (judge2[0] != 'N')) && (count9 == 0) )
        {
```

```
        t_loc(5 ,12);
            printf("DO YOU SAVE MAHALANOBIS' DISTANCE RESULT ?");
        t_loc(5 ,13);
            printf("<< YES ===> hit y / NO ===> hit n >> ");
            scanf  (" %s", judge2);
        }

/*########## CREATE RESULT - FILENAME ########################*/


        if( (judge1[ 0] == 'y') || (judge1[ 0] == 'Y') )
        {
            t_cls();
            g_cls();

            printf("¥n filename  for TABLE ");
            creat_kekka_dfname(name1);
            printf("¥n %s ¥n",name1);

            printf("¥n filename  for GRAPH ");
            creat_kekka_dfname(name2);
            printf("¥n %s ¥n",name2);

        }

    }



/*######## SELECT PARAMETER ##################################################*/

    t_cls();

    t_loc(5,2);
    printf("========================================== ¥n");

    t_loc(5,3);
    printf("SELECT LEAF KIND !!! ¥n");

    for( n = 0 ; n < COUNTLK[ 0] ; ++n)
    {
       t_loc( 10, 4+n );
       printf("LEAF KIND NO.%d ---> %s ¥n",n+1,searlk[ n]);
    }

    t_loc(5,4 + COUNTLK[ 0]);
    printf("========================================== ¥n");
```

```
    t_loc(5,4 + COUNTLK[ 0] + 1);
    printf("SELECT NO. &.PUSH THAT NO. ===>");

    if(flag1[ 0] == 0 || flag1[ 0] == 3)
    {
        scanf(" %d", &com);
        strcpy(sellk, searlk[ com - 1], strlen(searlk[ com - 1]) + 1);
    }

    if( (flag1[ 0] == 1) || (flag1[ 0] == 2) )
    {
        com1 = count7;
        strcpy(sellk, searlk[ com1], strlen(searlk[ com1]) + 1);
        count7++;
        t_loc(37,4 + COUNTLK[ 0] + 1);
            printf("%s",sellk);
     }


    str_count(sellk,arynumlk,LKIND,COUNTSLK);

    looplmt = COUNTSLK[ 0];

       for( j = 0 ; j < COUNTSLK[ 0] ; ++j)
       {
           for( i = 0 ; i < COUNTPAR[ 0] ; ++i)
           {

/*
            printf("PARAMETER[ arynumlk[ j]] = %s searpar[ i] = %s ¥n"
                                ,PARAMETER[ arynumlk[ j]],searpar[ i]);
*/
               if(com = strcmp(PARAMETER[ arynumlk[ j]],searpar[ i]) == 0)
             {
                draw2[ j] = i;
                break;
              }
           }
       }

countll = 0;

while(countll < looplmt)
{
    strcpy(selstr[ 0],"A",2);
    count12=0;

    while(1)
    {
```

```
for(i = 0 ; i < looplmt - 1 ; ++i)
{
    for(j = i + 1 ; j < looplmt ; ++j)
    {
        if(draw2[ i] > draw2[ j])
        {
            dummy = draw2[ i];
            draw2[ i] = draw2[ j];
            draw2[ j] = dummy;
        }
    }
}

t_cls();

t_loc(5,1);
    printf("========================================== ¥n");

t_loc(5,2);
    printf("SELECT PARAMETER !!! ¥n");

for(i = 0 ; i < looplmt ; ++i)
{
    t_loc(5,3 + i);

        printf("PARAMETER NO.%d ---> %s ¥n", draw2[ i]+1 , searpar[ draw2[ i]] );

}

t_loc(5,3 + looplmt);
    printf("PARAMETER NO.X  ---> < ALL > ¥n");

t_loc(5,3 + looplmt + 1);
    printf("        E       ---> < EXIT > ¥n");

t_loc(5,3 + looplmt + 2);
    printf("========================================== ¥n");

t_loc(5,3 + looplmt + 3);
    printf("SELECT NO. & PUSH THAT NO. ===>");


switch(flag1[ 0])
{
    case 0:
        scanf(" %s", selstr[ 0]);
        break;

    case 1:
        t_loc(37,4 + looplmt + 3);
```

```
            printf("x");
            strcpy(selstr[ 0],"x",2);
            break;

        case 2:
            count12++;
            if(count12 == 1)
                sscanf(selstr[ 0],draw2[ count11] +1);
            if(count12 == 2)
                strcpy(selstr[ 0],"e",2);
            break;

        case 3:
            scanf(" %s", selstr[ 0]);
            break;

        default:
            exit(1);
    }


    if( (com = strcmp(selstr[ 0],"X") == 0) ||
                        (com = strcmp( selstr[ 0],"x") == 0) )
    {
        for( i = 0 ; i < COUNTSLK[ 0] ; ++i)
            strcpy(selectpr[ i], searpar[ i],strlen(searpar[ i])+1);
        break;
    }


    if( (com = strcmp(selstr[ 0],"E") == 0) ||
                        (com = strcmp(selstr[ 0],"e") == 0) )
        break;

    else
    {
        com = (int)atoi(selstr[ 0]);

        strcpy(selectpr[ count4[ 0]], searpar[ com - 1]
                                ,strlen(searpar[ com - 1])+1);

        printf("%s",selectpr[ count4[ 0]]);
        count4[ 0] = count4[ 0] + 1;
    }
}


if( count4[ 0] != 0)
{
```

```
    for( i = 0 ; i < count4[ 0] ; ++i)
    {
        for( j = 0 ; j < looplmt ; ++j)
        {

            if(com = strcmp(PARAMETER[ arynumlk[ j]],selectpr[ i]) == 0)
            {
               draw1[ counter1[ 0]] = arynumlk[ j];
               counter1[ 0] = counter1[ 0] + 1;
               break;
            }
        }
    }

    for( k = 0 ; k < counter1[ 0] ; ++k)
        arynumlk[ k] = draw1[ k];


}

if(flag1[ 0] == 0)
{
    t_loc(5,3 + COUNTSLK[ 0] + 4);
        printf(" HOW MANY LEAVES DO YOU CHOICE [1-10]? ===>");
    scanf(" %d",&HMCHOICE);
    LIMITNUM = HMCHOICE +1;
}



if( count4[ 0] != 0)
{
    COUNTSLK[ 0] = count4[ 0];
}

t_cls();

g_line(0,0,639,399,4,1,0);

t_loc(55,2);
    printf("<選択したパラメータ> Yn");

for(i = 0; i < COUNTSLK[ 0] ; ++i)
{
    if(COUNTSLK[ 0] <= 12)
    {
        t_loc(60,3 + i);
        printf("%s",selectpr[ i]);
    }
    else
```

```
        {
            if(i <= 12)
            {
                t_loc(52,3 + i);
                printf("%s",selectpr[ i]);
            }

            else
            {
                t_loc(66,3 + i - 13);
                printf("%s",selectpr[ i]);
            }
        }
    }

    if(COUNTSLK[ 0] <= 12)
        g_line(55*8-5,2*16-8,77*8,(3+i)*16+8,1,2,0);

    else
        g_line(52*8-5,2*16-8,77*8+15,(3+13)*16+5,1,2,0);



    count8 = 0;

while(HMCHOICE < LIMITNUM)
{

    if( (flag1[ 0] == 1) || (flag1[ 0] == 2) || (flag1[ 0] == 3))
    {
        count8++;
        HMCHOICE = count8;
    }

/*############ INITIALIZE VARIABLE ##########################################*/
    ninritu = 0;
    for( n = 0 ; n < COUNTLK[ 0] ; ++n )
        ninritu1[ n] = 0;

    for(i = 0 ; i < 3 ; ++i)
        shuusoku[ i] = (double)100;

/*############ CREATE FILENAME (flag1=3) ####################################*/

    stccpy(node3,"ddk",4);
    strcat(node3,sellk);
    stci_d(dumychar,HMCHOICE);
    strcat(node3,dumychar);
```

```
    creat_kekka_dfname3(name3,node3);

    if( (judge2[ 0] == 'y') || (judge2[ 0] == 'Y') )
    {
        creat_kekka_dfname3(name3,node3);
        t_loc(52,16);
            printf("<SAVING MD FILE-NAME>");
        t_loc(52,17);
        printf(" %s ",name3);
    }
/*############ HANNDANN (MAIN  CULCURATE ROUTINE) ##########################*/
/*
    creat_kekka_dfname(name4);/** variable filename **/
*/
    while( (shuusoku[ 0] > (double)1.0 || shuusoku[ 1] > (double)1.0
                    || shuusoku[ 2] > (double)1.0) && count6 < 50 )
    {
        count5++;

        /* INITIALIZE */

        for(j=0;j<60;++j)
        {
            for(i=0;i<250;++i)
            DDK[ j][ i] = 0;
        }

        for(i=0;i<10;++i)
        {
            COUNT[ i] = 0;
            COUNT1[ i] = 0;
        }

        for(i = 0 ; i < COUNTSLK[ 0] ; ++i)
        {


            stccpy(selpar, PARAMETER[ arynumlk[ i]]
                                    ,strlen(PARAMETER[ arynumlk[ i]])+1);

            str_count(selpar,arynumpar,PARAMETER,COUNTSP);

            stccpy(parname, fname1[ arynumlk[ i]] ,strlen(fname1[arynumlk[ i]]) + 1);

            if((fp =  fopene(parname, "r",""")) == NULL)
                quit(" Can't Open File ");

            fscanf(fp," %s %s %d", s1, s2, &x1);

            RECORDNUM = x1;
```

```
counter3 = 0;

while((n=fscanf(fp," %s %lf %lf",  par, &avel, &roms)) != EOF)
{
 counter3++;

 if(com=strcmp(selpar, par) == 0)
     slprcnt[ 0] = counter3;

 sllkcnt[ 0] = counter3;
 }

file_close(fp);


strcpy(name, RAWNAME[ arynumlk[ i]]
                        ,strlen(RAWNAME[arynumlk[ i]]) + 1);

t_loc(52,20);
    printf("<< READING NOW !!! >> ");
t_loc(60,21);
    printf(" %s \n", name);

if((fi = fopene(name, "r",""")) == NULL)
    quit(" Can't Open File ");

fil_cal6( fi,slprcnt,sllkcnt);

file_close(fi);


t_loc(5,1);
    printf("========================================= ");

t_loc(8,2);
    printf(" <SEKECTED LEAF-KIND>   ===> %s",sellk);


t_loc(5,3);
    printf("   <CALCULATING PARAMETER> ===>                    ");

t_loc(33,3);
    printf("%s",selpar);

t_loc(6,4);
    printf("<CALCULATING LEAF NUMBER> ===> %d",HMCHOICE);

t_loc(5,5);
    printf("========================================= ");
```

```
        if(i == 0 && HMCHOICE > 1)
            choice_random();


        if(HMCHOICE > 1)
            choice_leaves();
/*
        if((fa = fopene(name4, "a","")) == NULL)
            quit(" Can't Open File ");
*/
        for( j = 0 ; j < RECORDNUM ; ++j)
        {
            if(HMCHOICE > 1)
                x0    = RVARIABLE[ j];
            else
                x0    = VARIABLE[ j];          /*** 葉が一枚で判断 ***/
/*
            fprintf(fa,"%lf ¥n",x0);
*/
            for( m = 0 ; m < COUNTSP[ 0] ; ++m)
            {

                ave   = AVERAGE[ arynumpar[ m]];
                rms   = ROOTMS[ arynumpar[ m]];

                z0 = (x0-ave)/rms;                          /* STANDERDLIZE */

                dk = cal_mahala(x0,ave,rms);

                for( n = 0 ; n < COUNTLK[ 0]; ++n)
                {

                    if(com = strcmp(LKIND[arynumpar[ m]],searlk[n]) == 0)
                    {
                        DDK[ n][ j] = DDK[ n][ j] + dk;
                      COUNT[ n] = COUNT[ n] + 1;
                    }
                }
            }
        }
/*
        file_close(fa);
*/

        for( n = 0 ; n < COUNTLK[ 0]; ++n)
        {
            COUNT1[ n] = COUNT1[ n] + COUNT[ n] / RECORDNUM;
            COUNT[ n] = 0;
```

```
                t_loc(2 ,25);
                    printf("%d ",count5);
                t_loc(22 + 3 * n,25 );
                    printf(" %d ",COUNT1[ n]);
            }

        }

        for( n = 0 ; n < COUNTLK[ 0] ; ++n)
        {
            count2[ 0]  = 0;
            count3[ n]  = 0;
        }


        if( (judge2[ 0] == 'y') || (judge2[ 0] == 'Y') )
        {
            if((ft = fopene(name3, "a","")) == NULL)
            quit(" Can't Open File ");
        }

        for( j =0 ; j < RECORDNUM ; ++j)
        {

            for( n = 0 ; n < COUNTLK[ 0]; ++n)
            {
                if(COUNT1[ n] != 0)
                DDK[ n][ j] = DDK[ n][ j]/ (double)COUNT1[ n];

                if(flag1[ 0] == 3 && ((judge2[ 0] == 'y') || (judge2[ 0] == 'Y')) )
                fprintf(ft,"%f ",(float)DDK[ n][ j]);

            }


            hdk[0]    = 10000;                          /******* hdk の初期化 ********/

            HAN_DAN( j, searlk, hdk, hlkind) ;

/*########## print result ###########################################################*/

/*
            if( (flag1[ 0] == 0) && ( (judge1[ 0] == 'y') || (judge1[ 0] == 'Y') ))
            {
                if( fmod( (double)j,(double)20) == 19)
                    fprintf(fs,"%s ¥n",hlkind[ 0]);

                else
                    fprintf(fs,"%s ",hlkind[ 0]);
            }
```

```
*/
            HAN_DAN1( sellk, hlkind, searlk, count2, count3);

            if(flag1[ 0] == 3 && ((judge2[ 0] == 'y') || (judge2[ 0] == 'Y')) )
                fprintf(ft,"%f  %s ¥n",(float)hdk[ 0],hlkind[ 0]);

        }

        if( (judge2[ 0] == 'y') || (judge2[ 0] == 'Y') )
        file_close(ft);

/*##### CULCLATE RATE of RECOGNIZATION ####################################*/

        if(HMCHOICE > 1)
        {
            ninritu2[ count6] = ninritu;

            ninritu  = ((ninritu*RECORDNUM*count6/100 + (double)count2[ 0])
                                / ((double)RECORDNUM*(count6+1))) * 100;

            if(count6 > 2)
            {
                shuusoku[ 0] = fabs(ninritu2[ count6-1] - ninritu2[ count6-2]);
                shuusoku[ 1] = fabs(ninritu2[ count6] - ninritu2[count6 -1]);
                shuusoku[ 2] = fabs(ninritu - ninritu2[ count6]);
                t_loc(52,22);
                printf("%3.3lf %2.3lf %2.3lf %2.3lf ",ninritu,shuusoku[ 0],shuusoku[ 1],sh
uusoku[ 2]);
            }

            for( n = 0 ; n < COUNTLK[ 0] ; ++n )
            {
                ninritul[ n] =
                 ((ninritul[ n]*RECORDNUM*count6/100 + (double)count3[ n])
                                / ((double)RECORDNUM*(count6+1))) * 100;
            }
            count6++;
            if(flag1[ 0] == 0)
                LIMITNUM = HMCHOICE - 1;

        }

        if(HMCHOICE == 1)
        {
            ninritu  = ((double)count2[ 0] / (double)RECORDNUM) * 100;

            for( n = 0 ; n < COUNTLK[ 0] ; ++n )
            {
                ninritul[ n] = ((double)count3[ n] / (double)RECORDNUM) * 100;
            }
```

```
            if(flag1[ 0] == 0)
            LIMITNUM = HMCHOICE - 1;

            break;
        }

    }


/*##### PRINT RESULT ##########################################################*/
/*
    for(i = 0; i <100; ++i)
    printf("COUNTLK = %d Yn",COUNTLK[ 0]);
*/

    t_loc(5 ,25- COUNTLK[ 0] -3 );
        printf("C-RATE IS %lf%C-LEAF KIND IS %s Yn", ninritu, sellk );

    for( n = 0 ; n < COUNTLK[ 0] ; ++n )
        {
          t_loc(5 ,25 - COUNTLK[ 0] + n - 2 );
          if(COUNT1[ n] !=0)
              printf(" %s = %lf% Yn",searlk[ n], ninritu1[ n]);
        }

   g_line(35,16*(25- COUNTLK[ 0]-3) - 4,8*44,368,2,2,0);

   if( (flag1[ 0] == 1) || (flag1[ 0] == 2) ||
       (flag1[ 0] == 0 && ((judge1[ 0] == 'y') || (judge1[ 0] == 'Y')))
       ||(flag1[ 0] == 3 && ((judge1[ 0] == 'y') || (judge1[ 0] == 'Y'))) )
   {
       if( (flag1[ 0] == 1) || (flag1[ 0] == 2))
       {
           creat_kekka_dfname1(sellk,LIMITNUM,name1,name2,flag1);
       }

       if((fr = fopene(name1, "a","")) == NULL)
       quit(" Can't Open File ");

       if((fs = fopene(name2, "a","")) == NULL)
       quit(" Can't Open File ");


       t_loc(52,18);
           printf("<SAVING FILE-NAME>");

       t_loc(52,19);
           printf(" %s",name1);
```

```
    count10++;

    t_loc(52,23);
    printf("LIMITNUM = %d",LIMITNUM);

    if( ( com = fmod((double)count10,(double)LIMITNUM)) == 1)
    fprintf(fs," %s Yn NUMBER of LEAVES Yn RATE of RECOGNIZATION  Yn",sellk);

    fprintf(fs,"%d  %lf Yn",HMCHOICE,ninritu);

    if(ninritu == (double)100)
    {
        for(i=HMCHOICE + 1;i <= LIMITNUM ; ++i)
        {
            fprintf(fs,"%d  %lf Yn",i,ninritu);
            count10++;

        }
    }

}

ke_kkal(fr,name1,ninritu,ninritu1,sellk,searlk,selectpr);

if(flag1[ 0] == 0 && ((judge1[ 0] == 'y') || (judge1[ 0] == 'Y')) )
{
    file_close(fr);

    file_close(fs);

    break;
}

if( (flag1[ 0] == 1) || (flag1[ 0] == 2)
 ||(flag1[ 0] == 3 && ((judge1[ 0] == 'y') || (judge1[ 0] == 'Y'))) )
{
    file_close(fr);
    file_close(fs);
    count5 = 0;
    count6 = 0;
}

if(flag1[ 0] == 3)
{
    count5 = 0;
    count6 = 0;
}

if(flag1[ 0] != 3 && ninritu == (double)100)
    break;
```

```
}

    HMCHOICE =0;

    switch(flag1[ 0])
    {
        case 0:
            count11 = looplmt +1;
            break;

        case 1:
            count11 = looplmt +1;
            break;

        case 2:
            count11++;
            count4[ 0] = 0;
            COUNTSP[ 0]   = 0;
            counter1[ 0]  = 0 ;
            break;

        case 3:
            count11 = looplmt +1;
            break;

        default:
            g_init;
            exit(1);
    }

}

    if(flag1[ 0] == 0)
    {
        while( ((judge[0] != 'y') && (judge[0] != 'n')) &&
                            ((judge[0] != 'Y') && (judge[0] != 'N')) )
        {
            t_loc(5 ,25 );
            printf (" 終了しますか？？ << YES ===> hit y / NO ===> hit n >>  ");

            scanf  (" %s", judge);              /* read on keyboard */
        }

        if( (judge[ 0] == 'y') || (judge[ 0] == 'Y') )
        {
            g_init();
            t_cls();
            printf("¥x1b[>5l");
            exit(1) ;
        }
```

```
    }

    count9++;

    if(flag1[ 0] == 3)
    {
        printf("¥x1b[>51");
        exit(1);
    }

    if(count9 == COUNTLK[ 0])
    {
        printf("¥x1b[>51");
        exit(1);
    }
}

}

/*++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++*/
/* サーチ SUBROUTINE ++++++++++++++++++++++++++++++++++++++++++++++++++++++++++*/
/* サーチされた文字列の種類数を数える ++++++++++++++++++++++++++++++++++++++++*/
/*++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++*/
str_search(search,set,count)

char search[ 20][ 20];
char set[ 260][ 20];
int  count[ 1];
{
    int i , j;
    int com;
    int counter1;
    int counter2;

    j = 0;
    counter1 = 0;

    stccpy(search[ j],set[ 0],strlen(set[ 0])+1);

    for( i = 0 ; i < FILNUM[ 0]-1; ++i)
    {
        counter2 = 0;

        for(j = 0 ; j <= counter1 ; ++j)
        {
            if(com = strcmp(search[ j],set[ i+1]) != 0 )
                counter2 = counter2 + 1;
            else
                break;
        }
```

```
        if(counter1 == counter2 -1)
        {

            counter1 = counter1 + 1;

            strcpy(search[ counter1],set[ i+1],strlen(set[ i+1]) +1);

        }
    }
    count[ 0] = counter1 + 1;
}

/*++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++*/
/* カウント & 配列 SUBROUTINE ++++++++++++++++++++++++++++++++++++++++++++++++*/
/* 選択された文字列と同じ文字列の数を数え、その配列番号を返す++++++++++++++++*/
/*++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++*/

str_count(select,arraynum,set,count)

int  arraynum[260];      /**** 選択したパラメタのファイル順を表わす ******/
int  count[ 1];
char select[ 20];       /**** 選択された配列 ***************************/
char set[ 50][ 20];
{
    int  i , j , com;

    count[ 0] = 0;
    for(i = 0 ; i < FILNUM[ 0] ; ++i)
    {
        if(com=strcmp(select, set[ i]) == 0)
        {
            j = count[ 0];
            arraynum[ j] = i;
            count[ 0] = count[ 0] + 1;
        }
    }
}

/*++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++*/
/* 判断 SUBROUTINE 1++++++++++++++++++++++++++++++++++++++++++++++++++++++++++*/
/*++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++*/

HAN_DAN(j,searlk,hdk,hlkind)

/*****************************************************************************/
int    j;
double hdk[1];              /**** 前回のマハラノビス距離 *********************/
char   searlk[ 20][ 20];    /**** 比較される樹種 *****************************/
char   hlkind[ 1][ 20];     /**** 判断した樹種 *******************************/
/*****************************************************************************/
```

```
{
int n;
    for( n =0 ; n < COUNTLK[ 0] ; ++n)
    {


        if( (COUNT1[ n] != 0) && (DDK[ n][ j] < hdk[0]) )
        {
            hdk[0]= DDK[ n][ j];
            stccpy(hlkind[0], searlk[ n],strlen(searlk[ n])+1);
                  /*printf(" %lf %lf %d",hdk[0],dk);*/
        }
    }
}
/*++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++*/
/* 判断 SUBROUTINE 2 ++++++++++++++++++++++++++++++++++++++++++++++++++++++++++*/
/*++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++*/

HAN_DAN1( lkind1, hlkind, searlk, count2, count3)

/*****************************************************************************/
char    lkind1[ 1][ 20];     /*** 真のLKIND[][]を代入するためのポインター変数*/
char    hlkind[ 1][ 20];     /*** 判断した樹種 ****************************/
char    searlk[ 20][ 20];    /*** 樹種のサーチ ****************************/
int     count2[ 1];
int     count3[ 10];
/*****************************************************************************/
{
int cp;

    if(cp = strcmp(lkind1[ 0],hlkind[ 0]) == 0)
    {
    count2[ 0]=count2[ 0] + 1;
    }
    for( n = 0 ; n < COUNTLK[ 0] ;++n)
    {
        if(cp = strcmp( searlk[ n], hlkind[ 0]) == 0)
        {
        count3[ n] = count3[ n] + 1;
        }
    }
}
/*++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++*/
/* 結果 SUBROUTINE ++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++*/
/*++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++*/

ke_kka1(fr,name,ninritu,ninritu1,sellk,searlk,selectpr)

/*****************************************************************************/
FILE *fr;               /*****************************************************/
```

```
char *name;             /**** fname[ ][ ]を代入するためのポインタ変数 *******/
char sellk[ 20];        /**** 樹種を選択するための配列 ****************/
char searlk[ 20][ 20];  /**** 樹種のサーチ ****************************/
char selectpr[ 25][ 20];/**** パラメタを選択するための配列 *************/
double ninritu;         /**** 真の樹種の認識率 ****************************/
double ninritul[ 10];   /**** ある樹種としての認識率 **********************/
/*************************************************************************/

{
    int i;

    fprintf(fr,"選択した葉の枚数 ===> %d",HMCHOICE);
    fprintf(fr," 認識率は %lf% 真の樹種は %s ¥n", ninritu, sellk );

    for( n = 0 ; n < COUNTLK[ 0] ; ++n )
        if(COUNT1[ n] != 0)
        fprintf(fr," %s = %lf% ¥n",searlk[ n], ninritul[ n]);

    fprintf(fr," <選択したパラメータ> ¥n");


    for(i = 0; i < COUNTSLK[ 0] ; ++i)
    {
        fprintf(fr,"%s  ¥n",selectpr[ i]);
    }
}

/*+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++*/
/* RANDOM SUBROUTINE ++++++++++++++++++++++++++++++++++++++++++++++++++++++*/
/*+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++*/
choice_random()

{
int    i;
int    j;
int    k;
int    shoki;
int    flag;
long   random;
   for( i = 0 ; i < HMCHOICE *RECORDNUM ; ++i)
   {

       random = lrand48();
       aryrand[ i] = (long)fmod( (double)random,(double)RECORDNUM );
       /*選択したHMCHOICE個の中に同じ数字があるかどうかのチェック*/
       if( (int)fmod( (double)i ,(double)HMCHOICE ) == HMCHOICE - 1)
       {
           flag = 0;
           for(j = i - HMCHOICE + 1 ; j < i ; ++j)
           {
```

```
                shoki = (int)fmod( (double)j,(double)HMCHOICE) + 1;

                for(k = shoki ; k < HMCHOICE ; ++k)
                {
                    if(aryrand[ j] == aryrand[ i - HMCHOICE + k+1])
                    {
                        i = i - HMCHOICE ;
                        flag = 1;
                        break;
                    }
                }
                if(flag == 1)
                break;
            }
        }
    }
}
/*++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++*/
/* RANDOM SUBROUTINE ++++++++++++++++++++++++++++++++++++++++++++++++++++++++++*/
/*++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++*/
choice_leaves()
{
int i;
int j;
int k;
long addt;
double x0[ 250];
double y0[ 250];
double average[ 1];      /**** データから計算した平均値 ******************/
double msquar[ 1];       /**** データから計算したＭＳ ********************/
double rmsquar[ 1];      /**** データから計算したＲＭＳ ******************/
double randomav;

addt = 0;
randomav = (double)0;

    for( i = 0 ; i < HMCHOICE *RECORDNUM ; ++i)
    {
        if( (int)fmod( (double)i ,(double)HMCHOICE ) == HMCHOICE - 1)
        {

            for(j = 0 ; j < HMCHOICE  ; ++j)
            {
                x0[ j] = VARIABLE[ aryrand[ i - HMCHOICE + 1 +j]];
            }

            cal_average(x0,HMCHOICE,average,msquar,rmsquar);

            RVARIABLE[ (int)( i / HMCHOICE )] = average[ 0];
        }
```

```
        addt = addt + aryrand[ i];
    }

    randomav = (double)addt / ((double)HMCHOICE *(double)RECORDNUM);

    t_loc(4 ,25);
        printf(" ranave = %3.3lf ",randomav);
}
/*++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++*/
/* QUIT SUBROUTINE ++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++*/
/*++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++*/

quit(s)
    char *s;
    {
        printf(s);
        exit(1);
    }
```

```
/*##############################################################################*/
/*      PROGRAMNAME = hon1.c                                                    */
/*      FUNCTION = 葉っぱ１枚持って来て、その樹種を診断するものである。temp   */
/*                 というバッチファイルで起動する。                             */
/*                                                                              */
/*              << THE TEAM OF LEAF >>  '89-12-吉日       PRODUCED by MAKI     */
/*##############################################################################*/
/******************************************************************************/
/************* DEFINE  GROVAL VARIABLE ****************************************/
    double  PARAVE[ 50], AVERAGE[ 300], ROOTMS[ 300];
    huge double  DDK[ 60]; /**** 選択した葉の各樹種へのdk 64 kbyte 超えないように*/
    int     RECORDNUM;
    int     r, n;
    int     FILNUM[1];          /**** 計算対象となるファイルの数 ****************/
    int     COUNTPAR[ 1];       /**** parameterの種類数 **************************/
    int     COUNTLK[ 1];        /**** 樹種の種類数 ******************************/
    int     COUNTSP[ 1];        /**** 選択したパラメタと同じパラメタを ***********/
                                /**** 持つファイルの数 **************************/
    int     COUNTSLK[ 1];       /**** 選択した樹種と同じ樹種を ******************/
                                /**** 持つファイルの数 **************************/
    int     COUNT[ 200];        /**** DDKに対応したカウント *********************/
    char    *s;
    char    LKIND[ 260][20];
    char    PARAMETER[ 260][20];
    char    RAWNAME[ 260][ 20];
/******************************************************************************/
    extern  unsigned _STACK = 30000;

    #include    "stdlib.h"
    #include    "stdio.h"
    #include    "string.h"
    #include    "math.h"
    #include    "dos86.h"
    #include    "maki.h"
    #include    "ctool.h"
/*  #include    "need.h"*/
    #include    "rname.h"
    #include    "nordisb.h"


main()
{

/###### DEFINE ################################################################*/
    FILE *fi,*fp,*fj;
    double x0, ave , rms , z0, gaus, ingaus , dk ;

    int num1[ 1];               /**** 計算対象となるファイルをオープンした順 ****/
                                /**** fil_call()の引数になっている **************/
```

```
    int dummy;
    int  i, j, k,l, m, com,coml;
    int  xl;
    int arynumpar[ 260];     /**** 選択したパラメタのファイル順を表わす *******/
    int arynumlk[ 260];      /**** 選択した樹種のファイル順を表わす **********/
    int counter2[ 1];

    char judge[ 3];          /**** プログラムの終了を判断するための文字列 *****/
    char fname[ 40][ 25];    /**** 計算対象となるファイルの名前 ***************/
    char fname1[ 260][ 25]; /**** 計算対象となるファイルの名前 ***************/
    char name[ 25];          /**** fname[ ][ ]を代入するためのポインタ変数 ****/
    char parname[ 25];       /**** parameter file nameを代入するポインタ変数 **/
    char searpar[ 260][ 20];/**** パラメタのサーチ ***************************/
    char searlk[ 260][ 20]; /**** 樹種のサーチ *****************************/
    char selpar[ 25];        /**** パラメタを選ぶための配列 ******************/
    char sellk[ 25];         /**** 樹種を選択するための配列 ******************/
    char hlkind[ 1][ 20];    /**** 前回の樹種 *******************************/
    char par[ 20];
    char *c;
    double hdk[1];           /**** 前回のマハラノビス距離 ********************/
    double readpar;
    double ranms;
    double avel;
    double roms;
/*##########################################################################*/
/*######## READ PARAMETER-FILE ##############################################*/


    printf("¥x1b[>1h");
    printf("¥x1b[>5h");

    if((fi = fopene("testdat4.dat", "r","")) == NULL)
    quit(" Can't Open File ");

    fread_name( fi,fname);

    file_close(fi);

    counter2[ 0] = 0;

    for(i=0 ; i < FILNUM[ 0]; ++i)
    {
        stccpy(name,fname[ i],strlen(fname[ i]) + 1);

        printf("%d %s ¥n",i,name);

        if((fp = fopene(name, "r","")) == NULL)
        quit(" Can't Open File ");

        num1[ 0] = i;   /*  LKIND & PARAMETER の中の配列番号 */
```

```
        fil_cal5( fp,counter2,fname,fname1,num1);

        file_close(fp);
    }

    FILNUM[ 0] = counter2[ 0];

/*######### パラメタ名  &  樹種名  の調査  ###################################*/

    str_search(searpar,PARAMETER,COUNTPAR);  /** パラメタのサーチ **/

    str_search(searlk,LKIND,COUNTLK);        /** 樹種のサーチ **/

/*#### MAIN LOOP ###########################################################*/
while(1)
{

/*######### INITIALIZE  VARIABLE ##########################################*/
    for(i=0;i<50;++i)
    PARAVE[ i]  = 0;

    r             = 0;
    n             = 0;
    COUNTSP[ 0]   = 0;
    COUNTSLK[ 0]  = 0;
    judge[ 0]     = 'x';

/*########################################################################*/
/*
        PARAVE[ 0] = 0.225304 ;    /** HENPEIRITU **/
        PARAVE[ 1] = 28.213679 ;   /** FUKUZATUDO **/
        PARAVE[ 2] = 0.671722 ;    /** SENYUURITU **/
        PARAVE[ 3] = 0.465199 ;    /** JYUUSINHI **/
        PARAVE[ 4] = 0.079178 ;    /** COUNT5 **/
        PARAVE[ 5] = 0.086532 ;    /** COUNT6 **/
        PARAVE[ 6] = 0.107623 ;    /** COUNT7 **/
        PARAVE[ 7] = 0.131718 ;    /** COUNT8 **/
        PARAVE[ 8] = 0.189081 ;    /** COUNT9 **/
        PARAVE[ 9] = 0.249254 ;    /** COUNT10 **/
        PARAVE[ 10] = 6.693989 ;   /** CHYOUTENSUU **/
        PARAVE[ 11] = 0.076501 ;   /** CHYOUTENRITU **/
        PARAVE[ 12] = 0.007585 ;   /** COUNT1 **/
        PARAVE[ 13] = 0.026990 ;   /** COUNT2 **/
        PARAVE[ 14] = 0.048973 ;   /** COUNT3 **/
        PARAVE[ 15] = 0.070086 ;   /** COUNT4 **/
        PARAVE[ 16] = 195.167022 ; /** NAGASA **/
        PARAVE[ 17] = 1412.210560 ;/** HABA **/
        PARAVE[ 18] = 95.930199 ;  /** SYUUICHYOU **/
        PARAVE[ 19] = 20.763181 ;  /** MENSEKI **/
```

```
*/
    if((fj = fopene("sex.dat", "r",""")) == NULL)
    quit(" Can't Open File ");

    i=0;

    while((n=fscanf(fj," %lf",  &readpar)) != EOF)
    {
        PARAVE[ i] = readpar;
        i++;
    }

    file_close(fj);

/************* HANNDANN (MAIN  CULCURATE ROUTINE) ***************************/
        /** INITIALIZED **/
        for(j=0;j<60;++j)
            DDK[ j] = 0;

        for(i=0;i<200;++i)
            COUNT[ i] = 0;


        for(i = 0 ; i < COUNTPAR[ 0] ; ++i)  /** loop 1 **/
        {

            strcpy(selpar, searpar[ i],strlen( searpar[ i])+1);

            str_count(selpar,arynumpar,PARAMETER,COUNTSP);

            x0  = PARAVE[ i];

            for( m = 0 ; m < COUNTSP[ 0] ; ++m)  /** loop 2 **/
            {

                ave   = AVERAGE[ arynumpar[ m]];

                rms   = ROOTMS[ arynumpar[ m]];

                dk = cal_mahala(x0,ave,rms);

                for( n = 0 ; n < COUNTLK[ 0]; ++n)
                {
                    if(com = strcmp( LKIND[arynumpar[ m]] ,searlk[n]) == 0)
                    {
                        DDK[ n] = DDK[ n] + dk;
                        COUNT[ n] = COUNT[ n] + 1;
                     }
                }
            }   /** loop 2 **/
```

```
}   /** loop 3 **/

for( n = 0 ; n < COUNTLK[ 0]; ++n)
{
    if(COUNT[ n] != 0)
    DDK[ n] = DDK[ n]/ (double)COUNT[ n];
}


hdk[0]    = 10000;                          /******* hdk の初期化 ********/

HAN_DAN( j, searlk, hdk, hlkind) ;

/** DISPLAY RESULT **/
g_init();
t_cls();

for( i = 0 ; i < 5 ; ++i)
{
    t_loc(5,2 + i);
    printf("Recognized Leaf = %s ¥n",hlkind[ 0]);
}

while(1)
{
    t_loc(5 ,22 );
    printf (" 終了しますか？？ << YES ===> hit y / NO ===> hit n >>  ");

        for(i = 0 ; i < 3 ; ++i)
        g_init();

        for(i = 0 ; i < 40 ; ++i)
        g_line(35,11,197,61,2,2,0);

    direct_key(c);

    if( *c == 'y' || *c == 'Y' || *c == 'n' || *c == 'N')
    {
        stccpy(judge ,c ,strlen(c) + 1);
        break;
    }
}



if( (judge[ 0] == 'y') || (judge[ 0] == 'Y') )
{
    g_init();
    t_cls();
    printf("¥x1b[>11");
```

```
            printf("¥x1b[>5l");
            exit(1) ;
        }

}

}
/*+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++*/
/* fil_cal5 SUBROUTINE ++++++++++++++++++++++++++++++++++++++++++++++++++++++++*/
/*+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++*/
fil_cal5( fi,counter2,fname,fname1,num1)
FILE *fi;
int counter2[ 1];
int num1[ 1];           /**** 計算対象となるファイルをオープンした順 ****/
char fname[ 40][ 25];  /**** 計算対象となるファイルの名前 **************/
char fname1[ 260][ 25];/**** 計算対象となるファイルの名前 **************/
{
    int i;
    int rawdatct;
    double ave,roms;
    char    lk[ 20],par[ 20];
    char rawdatnm[ 20];

    fscanf(fi," %s %s %d", rawdatnm, lk, &rawdatct);

    while((n=fscanf(fi," %s %lf %lf",  par, &ave, &roms)) != EOF)
    {
            strcpy(LKIND[ counter2[ 0]],lk,strlen(lk)+1);
            strcpy(PARAMETER[ counter2[ 0]],par,strlen(par)+1);
            strcpy(RAWNAME[ counter2[ 0]],rawdatnm,strlen(rawdatnm)+1);
            strcpy(fname1[ counter2[ 0]], fname[ num1[ 0]]
                                        ,strlen(fname[ num1[ 0]])+1);
            AVERAGE[ counter2[ 0]]  = ave;
            ROOTMS[ counter2[ 0]]   = roms;


            counter2[ 0]++;
    }

 }
/*+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++*/
/* サーチ SUBROUTINE ++++++++++++++++++++++++++++++++++++++++++++++++++++++++++*/
/* サーチされた文字列の種類数を数える ++++++++++++++++++++++++++++++++++++++++*/
/*+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++*/
str_search(search,set,count)

char search[ 20][ 20];
char set[ 260][ 20];
int  count[ 1];
{
```

```
    int i , j;
    int com;
    int counter1;
    int counter2;

    j = 0;
    counter1 = 0;

    strcpy(search[ j],set[ 0],strlen(set[ 0])+1);

    for( i = 0 ; i < FILNUM[ 0]-1; ++i)
    {
        counter2 = 0;

        for(j = 0 ; j <= counter1 ; ++j)
        {
            if(com = strcmp(search[ j],set[ i+1]) != 0 )
                counter2 = counter2 + 1;
            else
                break;
         }
        if(counter1 == counter2 -1)
        {

            counter1 = counter1 + 1;

            strcpy(search[ counter1],set[ i+1],strlen(set[ i+1]) +1);

        }
     }
    count[ 0] = counter1 + 1;
}

/*++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++*/
/* カウント & 配列 SUBROUTINE ++++++++++++++++++++++++++++++++++++++++++++++++*/
/* 選択された文字列と同じ文字列の数を数え、その配列番号を返す++++++++++++++++++*/
/*++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++*/

str_count(select,arraynum,set,count)

int  arraynum[260];      /**** 選択したパラメタのファイル順を表わす ******/
int  count[ 1];
char select[ 20];        /**** 選択された配列 ************************/
char set[ 50][ 20];
{
    int  i , j , com;

    count[ 0] = 0;
    for(i = 0 ; i < FILNUM[ 0] ; ++i)
    {
```

```
        if(com=strcmp(select, set[ i]) == 0)
        {
            j = count[ 0];
            arraynum[ j] = i;
            count[ 0] = count[ 0] + 1;
        }
    }
}

/*++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++*/
/* 判断 SUBROUTINE 1++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++*/
/*++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++*/

HAN_DAN(j,searlk,hdk,hlkind)

/*******************************************************************************/
int    j;
double hdk[1];              /**** 前回のマハラノビス距離 ********************/
char   searlk[ 20][ 20];    /**** 比較される樹種 ****************************/
char   hlkind[ 1][ 20];     /**** 判断した樹種 ******************************/
/*******************************************************************************/
{
int n;
    for( n =0 ; n < COUNTLK[ 0] ; ++n)
    {


        if( (COUNT[ n] != 0) && (DDK[ n] < hdk[0]) )
        {
            hdk[0]= DDK[ n];
            strcpy(hlkind[0], searlk[ n],strlen(searlk[ n])+1);
        }
    }
}

/*++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++*/
/* DIRECT KEY SUBROUTINE ++++++++++++++++++++++++++++++++++++++++++++++++++++++*/
/*++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++*/
direct_key(c)
char *c;
{
    union   REGS inregs;
    union   REGS outregs;

    int intno;

    intno = 0x21;
    inregs.h.ah = 0x06;
    inregs.h.dl = 0xff;
```

```
    int86(intno, &inregs, &outregs);

    *c = outregs.h.al;

    return ((*c != 0) ? 0 : 1);
}



/*++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++*/
/* QUIT SUBROUTINE ++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++*/
/*++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++*/

quit(s)
    char *s;
    {
        printf(s);
        exit(1);
     }
```

```
/*************************************************************************/
/*      PROGRAMNAME = SQUARE                                             */
/*      FUNCTION = CALICULATION OF SQUARE                                */
/*      << THE TEAM OF LEAF >>  '89-8-31    PRODUCED by MAKI             */
/*      このプログラムは、確率変数ｘの範囲  x1≦ｘ≦x2の間の確率（面積）*/
/*      を求めるものである。引数は必ずｚとし、ｚ＝（ｘ－ｍ）／ｒｍｓであ*/
/*      る。                                                             */
/*************************************************************************/

cal_square(z0)

    double z0;
{
    double gaus1 ,gaus2 ,gaus3 ,ingaus ,ave ,rms ,x1 ,x2;       /* define */
    double pai ,a1 ,a2 ,step;
    int    i,count;

    pai = 3.1415926535 ;                    /* initialize */
    ave = 0 ;
    rms = 1 ;
    ingaus = 0;
    step =0.005;
    z0 = fabs(z0);
    count = (z0 + step)/step;

    for(i = 1 ;i < count ;++i)
        {
        x1 = step*(i - 1);
        x2 = step*i;

        a1 = rms * sqrt(2*pai);
        a2 = - ((x1-ave)*(x1-ave))/(2*rms*rms);
        gaus1 = (1/a1) * exp(a2);

        a1 = rms * sqrt(2*pai);
        a2 = - ((x2-ave)*(x2-ave))/(2*rms*rms);
        gaus2 = (1/a1) * exp(a2);

        gaus3 = (gaus1+gaus2)*step/2;

        ingaus = ingaus + gaus3;
        }
printf(" ¥n square ===>");
printf(" %lf",ingaus);
}
```

```
/*************************************************************************/
/* FILE CALL SUBROUTINE **************************************************/
/*************************************************************************/
fil_call( fi,num1)
FILE *fi;
int num1[ 1];
{
    double ave,roms;
    char    lk[ 20],par[ 20];
    int dum1,dum2;

    fscanf(fi," %s %s %lf %lf", lk, par, &ave, &roms);
        dum1=strcpy(LKIND[ num1[ 0]],lk,strlen(lk)+1);
        dum2=strcpy(PARAMETER[ num1[ 0]],par,strlen(par)+1);
        AVERAGE[ num1[ 0]]=ave;
        ROOTMS[ num1[ 0]]=roms;
}


fil_cal2( fi)
FILE *fi;

{
    double ab;
    char *s;
    int i;

    s=0;
    for( i = 0 ; i < 4 ; ++i)
        {
            fscanf(fi,"%s", s);
        }

    i = 0;

    while((n = fscanf( fi," %lf ", &ab)) != EOF)
        {
            VARIABLE[ i] = ab;
        /*  printf("  %lf \n",VARIABLE[ i]);*/
            i = i + 1;
        }
        RECORDNUM = i;

}
fil_cal3( fi,num1)
FILE *fi;
int num1[ 1];
{
    double  ave,roms;
    char    lk[ 20],par[ 20],name[ 20];
```

```
    int     leafnum;

    fscanf(fi," %s %s %lf %lf", lk, par, &ave, &roms);
        strcpy(LKIND[ num1[ 0]],lk,strlen(lk)+1);
        strcpy(PARAMETER[ num1[ 0]],par,strlen(par)+1);
        AVERAGE[ num1[ 0]]=ave;
        ROOTMS[ num1[ 0]]=roms;
}


fil_cal4( fi)
FILE *fi;

{
    double ab;
    char *s;
    int i;

    s=0;
    for( i = 0 ; i < 4 ; ++i)
        {
            fscanf(fi,"%s", s);
        }

    i = 0;

    while((n = fscanf( fi," %lf ", &ab)) != EOF)
        {
            VARIABLE[ i] = ab;
        /*  printf("  %lf ¥n",VARIABLE[ i]);*/
            i = i + 1;
        }
        RECORDNUM = i;

}

fil_cal5( fi,counter2,fname,fname1,num1)
FILE *fi;
int counter2[ 1];
int num1[ 1];           /**** 計算対象となるファイルをオープンした順 ****/
char fname[ 40][ 25];  /**** 計算対象となるファイルの名前 **************/
char fname1[ 260][ 25];/**** 計算対象となるファイルの名前 **************/
{
    int i;
    int rawdatct;
    double ave,roms;
    char    lk[ 20],par[ 20];
    char rawdatnm[ 20];

    fscanf(fi," %s %s %d", rawdatnm, lk, &rawdatct);
```

qefile 1

```
    while((n=fscanf(fi," %s %lf %lf",  par, &ave, &roms)) != EOF)
    {
            stccpy(LKIND[ counter2[ 0]],lk,strlen(lk)+1);
            stccpy(PARAMETER[ counter2[ 0]],par,strlen(par)+1);
            stccpy(RAWNAME[ counter2[ 0]],rawdatnm,strlen(rawdatnm)+1);
            stccpy(fname1[ counter2[ 0]], fname[ num1[ 0]]
                                        ,strlen(fname[ num1[ 0]])+1);
            AVERAGE[ counter2[ 0]]  = ave;
            ROOTMS[ counter2[ 0]]   = roms;


            counter2[ 0]++;
    }

/*
            for(i = 0 ; i < counter2[ 0] ; ++i )
            printf(" %d %s %s %lf %lf ¥n",i,LKIND[ i],PARAMETER[ i],AVERAGE[ i]
                                                                ,ROOTMS[ i]);
*/
}

fil_cal6( fi,slprcnt,sllkcnt)
FILE *fi;
int slprcnt[ 1];
int sllkcnt[ 1];
{
    double ab;
    char *s;
    int i;

    s = 0;
    i = 0;

    fscanf(fi," %s ", s);
    while((n = fscanf( fi," %lf ", &ab)) != EOF)
        {

            if((int)fmod((double)i,(double)sllkcnt[ 0]) == (slprcnt[ 0] - 1))
            VARIABLE[ (i - slprcnt[ 0] + 1) / sllkcnt[ 0]] = ab;

            i++;
        }

}
```

```
/*++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++*/
/*      PROGRAMNAME = CAL_AVERAGE                                               */
/*      FUNCTION = IF YOU INPUT THE SUM of RANDOM VARIABLES(X) and THE NUMBER */
/*                 of RANDOM VARIABLES(X), THIS PROGRAM WILL OUTPUT AVERAGE.  */
/*                                                                             */
/*      このプログラムは、確率変数xのとその個数を与えると平均値を          */
/*      返してくれる プログラムである。                                        */
/*                                                                             */
/*              << THE TEAM OF LEAF >>  '89-11-06   PRODUCED by MAKI           */
/*++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++*/

cal_average(x0,num,average,msquar,rmsquar)

double  x0[ 250];
double  average[ 1];
double  msquar[ 1];
double  rmsquar[ 1];
int     num;

{
int     i;
double  sumave[ 1];
double  sumrms[ 1];

/*#### INITIALIZE #####*/

  sumave[ 0] = 0;
  sumrms[ 0] = 0;

for(i = 0;i < num ;++i)
    {
        sumave[ 0]   = sumave[ 0] + x0[ i] ;
    }
    average[ 0]  = sumave[ 0] / (double)num ;

for(i = 0;i < num ;++i)
    {

        sumrms[ 0]   = sumrms[ 0] + (x0[ i] - average[ 0])*(x0[ i] - average[ 0]);
/*
        sumrms[ 0]   = sumrms[ 0] + x0[ i] *x0[ i] ;
*/
    }
/*
    sumrms[ 0] = sumrms[ 0] - (double)num*average[ 0]*average[ 0];
*/
    msquar[ 0] = sumrms[ 0] / (double)num ;
    rmsquar[ 0]  = sqrt(fabs(msquar[ 0])) ;
}
/*++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++*/
```

```
/*      PROGRAMNAME = NORDISB                                                  */
/*      FUNCTION = IF YOU INPUT RANDOM VARIABLES(X) AND AVERAGE(m)             */
/*                 AND RMS(sigma),THIS PROGRAM WILL OUTPUT PROBABILITY.        */
/*                                                                             */
/*      このプログラムは、確率変数ｘと平均値ｍと標準偏差シグマを与えると      */
/*      その確率を返してくれるプログラムである。                               */
/*                                                                             */
/*             << THE TEAM OF LEAF >>  '89-9-1     PRODUCED by MAKI            */
/*+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++*/

double  cal_nordisb(x0, ave, rms)

    double x0 ,ave , rms;
{
    double gaus ;                               /* define */
    double a1 ,a2 ;


        a1 = rms * sqrt(2.0*PI);
        a2 = -1 * ((x0-ave)*(x0-ave))/(2*rms*rms);
        gaus = (1/a1) * exp(a2);

    return (gaus);
}
/*+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++*/
/*      PROGRAMNAME = SQUARE                                                   */
/*      FUNCTION = CALICULATION OF SQUARE                                      */
/*      << THE TEAM OF LEAF >>  '89-8-31    PRODUCED by MAKI                   */
/*      このプログラムは、確率変数ｘの範囲  x1≦ｘ≦x2の間の確率（面積）       */
/*      を求めるものである。引数は必ずｚとし、ｚ＝（ｘ－ｍ）／ｒｍｓであ       */
/*      る。                                                                   */
/*+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++*/

double  cal_square(z0)

    double z0;
{
    double ingaus;
    double gaus1 ,gaus2 ,gaus3 ,ave ,rms ,x1 ,x2;       /* define */
    double a1 ,a2 ,step;
    int    i,count;

    ave = 0 ;
    rms = 1 ;
    ingaus = 0;
    step =0.005;
    z0 = fabs(z0);
    count = (z0 + step)/step;

    for(i = 1 ;i < count ;++i)
```

```
        {
        x1 = step*(i - 1);
        x2 = step*i;

        a1 = rms * sqrt(2*PI);
        a2 = -1 * ((x1-ave)*(x1-ave))/(2*rms*rms);
        gaus1 = (1/a1) * exp(a2);

        a1 = rms * sqrt(2*PI);
        a2 = -1 * ((x2-ave)*(x2-ave))/(2*rms*rms);
        gaus2 = (1/a1) * exp(a2);

        gaus3 = (gaus1+gaus2)*step/2;

        ingaus = ingaus + gaus3;
        }
    return (ingaus);
}
/*++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++*/
/*      PROGRAMNAME = MAHALA                                                    */
/*      FUNCTION = CALICULATION OF MAHALANOBIS' GENERALIZED DISTANCE            */
/*      << THE TEAM OF LEAF >>  '89-10-3    PRODUCED by MAKI                    */
/*      このプログラムは、確率変数ｘに対するマハラノビスの距離を求める          */
/*      ものである。引数はｘ０，ａｖｅ，ｒｍｓとして、返り値はｄｋとす        */
/*      る。                                                                    */
/*++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++*/

double  cal_mahala(x0,ave,rms)

    double x0,ave,rms;
{
    double dk;
    dk = fabs(x0 - ave) / rms ;

    return (dk);
}
```

```
/*++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++*/
/*      PROGRAMNAME = RNAME                                                     */
/*      FUNCTION = READING WORDS ON KEYBOARD                                    */
/*      << THE TEAM OF LEAF >>  '89-6-28    PRODUCED by MAKI                    */
/*++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++*/
read_name(name)

    char    name[25];
{
    char    judge[3];                       /* declare various */
    judge[0] = 'x';                         /* initialize */

    while( (judge[0] != 'y') && (judge[0] != 'Y') )
    {
        judge[0] = 'x';                     /* initialize */

        printf ("¥n Enter file name --- ");
        scanf  (" %s",name);                /* read on keyboard */
        printf (" %s",name);

        while( ((judge[0] != 'y') && (judge[0] != 'n')) &&
                                ((judge[0] != 'Y') && (judge[0] != 'N')) )

        {
            printf ("¥n File name OK ? << YES ===> hit y / NO ===> hit n >>  ");

            scanf  (" %s", judge);          /* read on keyboard */
        }
    }
}
/*++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++*/
/*      PROGRAMNAME = FRNAME                                                    */
/*      FUNCTION = READING WORDS ON DATAFILE                                    */
/*      << THE TEAM OF LEAF >>  '89-10-24   PRODUCED by MAKI                    */
/*++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++*/
fread_name( fi,fname)

    FILE *fi;
    char fname[ 40][25];
    {
        int i,dum,na;
        char a[25];
        i=0;
        while( na = fscanf( fi,"%s",a) != EOF)  /* read on file */
        {
            dum=strcpy(fname[i],a,strlen(a)+1);
            printf(" %s ¥n",fname[i]);
            i = i + 1;
        }
```

r name.lv

```
    FILNUM[0] = i;
    }

/*+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++*/
/*     PROGRAMNAME = creat_kekka_dfname                                          */
/*      FUNCTION = create kekka - datafilename                                   */
/*      << THE TEAM OF LEAF >>  '89-11-06   PRODUCED by MAKI                     */
/*+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++*/
creat_kekka_dfname(name)

    char name[ 25];

{
    char drive1[ 1][ 4];
    char path1[ 1][ 15];
    char ext1[ 1][ 5];
    char node[ 10];

    printf("a:¥¥path¥¥node.ext  の nodeのみinputせよ！！！");

    strcpy(drive1[ 0],"a",strlen("a")+1);

    strcpy(path1[ 0],"¥¥rdata¥¥",strlen("¥¥rdata¥¥")+1);

    strcpy(ext1[ 0],"dat",strlen("dat")+1);

    read_name(node);

    printf(" ¥n %s ¥n",node);

    strmfn(name,drive1[ 0],path1[ 0],node,ext1[ 0]);

}

/*+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++*/
/*     PROGRAMNAME = creat_kekka_dfname1                                         */
/*      FUNCTION = create kekka - datafilename                                   */
/*      << THE TEAM OF LEAF >>  '89-11-06   PRODUCED by MAKI                     */
/*+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++*/
creat_kekka_dfname1(set,LIMITNUM,name,name1,flag)

    int  flag[1];
    char name[ 25];
    char name1[ 25];
    char set[ 20];
{
    char drive1[ 1][ 4];
    char path1[ 1][ 15];
    char node1[ 1][ 20];
    char ext1[ 1][ 5];
```

```
    char node2[ 1][ 20];

    stccpy(drive1[ 0],"a",strlen("a")+1);

    stccpy(path1[ 0],"¥¥rdata¥¥",strlen("¥¥rdata¥¥")+1);

    stccpy(ext1[ 0],"dat",strlen("dat")+1);

    stccpy(node2[ 0],"g",2);

    stci_d(node1[ 0],LIMITNUM);

    if(flag[ 0] == 1)
       strins(node1[ 0] ,"x");

    strins(node1[ 0] ,set);

    strins(node2[ 0] ,node1[ 0]);

/*
    printf(" ¥n %s ¥n",node1[ 0]);
*/
    strmfn(name,drive1[ 0],path1[ 0],node1[ 0],ext1[ 0]);
    strmfn(name1,drive1[ 0],path1[ 0],node2[ 0],ext1[ 0]);

}

/*++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++*/
/*     PROGRAMNAME = creat_kekka_dfname                                         */
/*      FUNCTION = create kekka - datafilename(for DISK)                        */
/*      << THE TEAM OF LEAF >>  '89-11-06   PRODUCED by MAKI                    */
/*++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++*/
creat_kekka_dfname3(name,node)

    char name[ 25];
    char node[ 10];
{
    char drive1[ 1][ 4];
    char path1[ 1][ 15];
    char ext1[ 1][ 5];

    t_loc(5,7);
    printf("INPUT DISK C DRIVE! ! !");

    stccpy(drive1[ 0],"c",strlen("c")+1);

    stccpy(path1[ 0],"¥¥",strlen("¥¥")+1);

    stccpy(ext1[ 0],"dat",strlen("dat")+1);
```

fname.h

```
/*
    printf(" ¥n %s ¥n",node);
*/
    strmfn(name,drive1[ 0],path1[ 0],node,ext1[ 0]);

}

/*+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++*/
/*     PROGRAMNAME = creat_ram_fname                                             */
/*      FUNCTION = create kekka - datafilename                                     */
/*      << THE TEAM OF LEAF >>  '89-11-06   PRODUCED by MAKI                      */
/*+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++*/
creat_ram_fname(name,name1)

    char name [ 15];
    char *name1;

{
    char path1[ 1][ 6];
    char node[ 1][ 10];

    strcpy(path1[ 0],"d:¥¥",5);

    strcpy(node[ 0],name1,8);

    strmfp(name,path1[ 0],node[ 0]);

}

/*+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++*/
/* FILE CLOSE SUBROUTINE ++++++++++++++++++++++++++++++++++++++++++++++++++++++*/
/*+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++*/
file_close(fi)
FILE *fi;
{
    if((r = fclose( fi)) == -1)
        quit(" Can't Close File ");
}
```

```
/*##############################################################################*/
/*      PROGRAMNAME = gt1.c                                                     */
/*      FUNCTION =    */
/*                    */
/*                    */
/*                    */
/*                    */
/*                    */
/*                    */
/*                    */
/*                    */
/*              << THE TEAM OF LEAF >>  '89-1-someday      PRODUCED by MAKI   */
/*##############################################################################*/

    huge   double ddk[ 3000];

    extern  unsigned _STACK = 30000;

    #include    "stdlib.h"
    #include    "stdio.h"
    #include    "string.h"
    #include    "math.h"
    #include    "dos86.h"
    #include    "maki.h"
    #include    "ctool.h"

    #define   RP  0x0c
    #define   DC1  0x11
    #define   DC3  0x13
    #define   RT  0x0A
    #define   CR  0x0D
    #define   EOT  0x04

main()
{
/*###### DEFINE ################################################################*/

    FILE   *fi,*fp;
    int    i,j,n;
    int    i1;
    int    i2;
    int    x1,y1;
    int    x2,y2;
    int    x0,y0;              /** graph no genntenn **/
    int    tx0,ty0;            /** text no genntenn **/
    int    dymax;              /** Y - point of max probability **/
    int    ystd;               /** Y - standard **/
    int    memcnt;             /** the number of Y - memori **/
    int    dummy1;
    int    comp1;
```

```
    int    comp2;
    int    flag1;
    int    amflag2;
    int    draw1;
    int    count1;
    int    count2;
    int    count3;
    int    count4;
    int    count5;
    int    count6;
    int    jyushusu;            /** the number of leaf kind **/
    int    seljyush;            /** selected  leaf kind no.**/
    int    setcolor1;           /** color no.**/
    char   gcflag3[ 3];
    char   drive1[ 3];          /** drive character **/
    char   name1[ 25];          /** name character **/
    char   ext1[ 5];            /** filename extend character **/
    char   h1kind[ 25];         /** recognized leaf kind **/
    char   jyushu[ 20][ 25];    /** leaf kind **/
    char   chardum[ 256];       /** dummy for character **/
    char   inpchar[ 256];       /** variable for input character **/
    char   selpar[ 256];        /** selected parameter **/
    char    selnum[ 15][ 3];     /** selected  number of leaf **/
    char   test1k[ 10];         /** tested leaf kind **/
    char   w1kind[ 30][ 5];     /** recognized leaf kind wrong **/
    char   judge1[ 3];          /** judge to input data disket in b drive **/
    char   brkjudge[ 3];        /** for judgement break while-loop **/
    char   *prg;
    double step;                /** histgram step **/
    double memstep;             /** one step of Y - memori **/
    double 1kddk;
    double ddkdum;
    double cddkmax;             /** MAX ddk in  recognized leaf kind correctly **/
    double cddkmin;             /** MIN ddk in  recognized leaf kind correctly **/
    double wddkmax;             /** MAX ddk in  recognized all leaf kinds wrong **/
    double wddkmin;             /** MIN ddk in  recognized all leaf kinds wrong **/
    double w1kmax[ 50];          /** MAX ddk in  recognized a leaf kind wrong **/
    double w1kmin[ 50];         /** MIN ddk in  recognized a leaf kind wrong **/
    double distrib[ 500];

/*##### SET UP SCREEN ##########################################################*/

    printf("¥x1b[>1h");
    printf("¥x1b[>5h");
    g_init();
    t_cls();
    g_screen(3,0,0,1);


/*##### INITIALIZE #############################################################*/
```

```
    count1 = 0;
    count6 = 0;
    jyushusu = 0;
    for( i = 0 ; i < 500 ; ++i)
    distrib[ i] = 0;
    for( i = 0 ; i < 15 ; ++i)
    sellnum[ i][ 0] = 0;
    judge1[ 0] = 0;
    brkjudge[ 0] = 0;
    x0 = 140;
    y0 = 320;
    tx0 = 2;
    ty0 = 0;
    amflag2 = 100;
    gcflag3[ 0] = 0;
/*##### DATA DISK CHECK ###########################################################*/

    while(judge1[ 0] != 'Y' && judge1[ 0] != 'y')
    {
        for( i = 0 ; i < 3 ; ++i)
        {
            t_loc(20,10 + i);
            printf("INPUT DATA DISKET B DRIVE !!!");
        }

        t_loc(20,13);
        printf(" DO YOU OK ? -- hit <Y> or <N> --");
        scanf("%s",judge1);
    }

    t_cls();

/*##### MENU ######################################################################*/

    t_loc(tx0-1,ty0);
    printf("================================");
    t_loc(tx0,ty0+1);
    printf("<< Manual >>    ===> Hit < 0 > ");
    t_loc(tx0,ty0+2);
    printf("<< Half-Auto >> ===> Hit < 1 > ");
    t_loc(tx0,ty0+3);
    printf("<< All-Auto >>  ===> Hit < 2 > ");
    t_loc(tx0-1,ty0+4);
    printf("================================");

    while(amflag2 != 0 && amflag2 != 1 && amflag2 != 2)
    {
        t_loc(tx0,ty0+5);
        printf("Select & Hit Key===>");
```

```
    scanf("%d",&amflag2);
}

t_loc(tx0,ty0+0);
printf("Input Tested Leaf Kind ===>");
scanf("%s",testlk);
bigto_small(testlk);

while(brkjudge[ 0] != 'q' && brkjudge[ 0] != 'Q')
{
    t_loc(tx0,ty0 + 7);
    printf("Input Number of Leaf [ 1 - 10 ] ===>");
    scanf("%s",brkjudge);
    stccpy(sellnum[ count6++], brkjudge,strlen(brkjudge)+1);

    t_loc(tx0+43,ty0 +count6);
    printf(" Inputed Number of Leaf  =  %s",brkjudge);

    t_loc(tx0,ty0 + 8);
    printf("QUIT ===> HIT < Q > Key");
}

if((fi = fopene("grainfo1.dat", "r","")) == NULL)
quit(" Can't Open File ");

    t_loc(tx0,ty0 + 9);
    printf("=========================================");

        while((n=fscanf(fi," %s ", jyushu[ count1])) != EOF)
        {
            count1++;
            t_loc(tx0+5,ty0 + 9 + count1);
            printf("  %s ---> %d",jyushu[ count1-1],count1);
        }
            jyushusu = count1;
    t_loc(tx0,ty0 + 9 + count1 +1);
    printf(" =========================================");

    if(amflag2 == 0)
    {
        t_loc(tx0,ty0 + 9 + count1 +2);
        printf("HIT a LEAF KIND'S NO.[ 1 - %d ]===> ",jyushusu);
        scanf("%d",&seljyush);
        count1 = 1;
     }

    t_loc(tx0,ty0 + 9 + jyushusu +3);
    printf("Input Selected Parameter No.===> ");
    scanf("%s",selpar);
```

```
        t_loc(tx0,ty0 + 8 + jyushusu +4);
        printf("Input Selected Histgram Step [0.1 or 0.05 or 0.025] ===> ");
        scanf("%lf",&step);


        if(amflag2 == 1 || amflag2 == 2 )
        {
            while(gcflag3[ 0] != 'y' &&  gcflag3[ 0] != 'Y'
                    && gcflag3[ 0] != 'n' &&  gcflag3[ 0] != 'N')
            {
                t_loc(tx0,ty0 + 8 + count1 +5);
                printf("Do You Print ===> < Y >  or  < N >");
                scanf("%s",gcflag3);
            }
        }


/** set graph condition **/
        if(step == 0.05)
        {
            ystd = 500;         /** y-axis max is 0.6 **/
            memcnt = 4;
            memstep = 0.2;
        }

        else if(step == 0.025)
        {
            ystd = 750;         /** y-axis max is 0.4 **/
            memcnt = 5;
            memstep = 0.1;
        }
        else
        {
            ystd = 300;         /** y-axis max is 1.0 **/
            memcnt = 6;
            memstep = 0.2;
        }

    file_close(fi);


for( i1 = 0 ; i1 < count6 - 1 ; i1++)     /** NO.1 LOOP START **/
{
for( i2 = 0 ; i2 < count1 ; i2++)         /** NO.2 LOOP START **/
{
    if(amflag2 == 1 || amflag2 ==2)
    seljyush = i2 + 1;

/*##### CREATE RAW-DATA FILNAME ##########################################################*/
```

```
    name1[ 0] = 0;
    stccpy(drive1,"b:",strlen("b:") + 1);
    stccpy(ext1,".dat",strlen(".dat") + 1);
    stccpy(name1,"ddk",strlen("ddk") + 1);
    strncat(name1,testlk,strlen( testlk) + 1);
    strncat(name1,sellnum[ i1],strlen(sellnum[ i1]) + 1);
/*
for(i=0;i<20;++i)
printf("%d %s %s ¥n",i1,sellnum[ i1],name1);
*/


/** CHECK BIG WORD or SMALL WORD **/
    bigto_small(name1);

/** INPUT TESTED LEAF KIND **/
/*
    testlk[ 0] = name1[ 3];
    testlk[ 1] = 0;
*/
    strins(name1,drive1);
    strcat(name1,ext1);

/###### READ RAW-DATA ##########################################################*/

    if((fi = fopene(name1, "r","")) == NULL)
    quit(" Can't Open File ");

/** initialize **/
    cddkmax = 0;
    cddkmin = 100000;
    wddkmax = 0;
    wddkmin = 100000;
    for( i = 0 ; i < 50 ; ++i)
        wlkmax[ i] = 0;
    for( i = 0 ; i < 50 ; ++i)
        wlkmin[ i] = 100000;
    count2 = 0;
    count3 = 0;
    count4 = 0;
    count5 = 0;
    flag1  = 0;

    stccpy(wlkind[ 0],"maki",strlen("maki")+1);


    while(1)
    {
```

```
        for(i = 0 ; i < jyushusu ; ++i)
        {
            if(n=fscanf(fi,"%lf",&ddkdum) == EOF)
            {
                flag1 = 1;
                break;
            }


            if((int)fmod((double)count3,(double)(jyushusu + 2)) == seljyush-1)
                ddk[ count2 ] = ddkdum;

            count3++;
        }

        if(flag1 == 1)
            break;

        fscanf(fi,"%lf %s",&lkddk,hlkind);

/** input cmaxddk & cminddk **/

        if(0x40 < (int)hlkind[ 0] && (int)hlkind[ 0] < 0x60)
        {
            dummy1 = (int)hlkind[ 0] + 0x20;
            hlkind[ 0] = (char)dummy1;
            hlkind[ 1] = 0;
        }

        if( (j = strcmp(hlkind,testlk)) == 0)
        {
            if(cddkmax < lkddk)
                cddkmax = lkddk;
            if(cddkmin > lkddk)
                cddkmin = lkddk;
        }
        else
        {

            for( i = 0 ; i <= count4 ; i++)
            {
                if( (j = strcmp(hlkind,wlkind[ i])) == 0)
                {
                    if(wlkmax[ i] < lkddk)
                        wlkmax[ i] = lkddk;
                    if(wlkmin[ i] > lkddk)
                        wlkmin[ i] = lkddk;
                    break;
                }
            }
```

```
            if( i == count4 + 1)
            {
                    strcpy(wlkind[ count4],hlkind,strlen(hlkind)+1);
                    strcpy(wlkind[ count4+1],"maki",strlen("maki")+1);

                    if(wlkmax[ count4] < lkddk)
                       wlkmax[ count4] = lkddk;
                    if(wlkmin[ count4] > lkddk)
                       wlkmin[ count4] = lkddk;

                    count4++;
            }

            if(wddkmax < lkddk)
                wddkmax = lkddk;
            if(wddkmin > lkddk)
                wddkmin = lkddk;

            count5++;
        }

        count3=count3+2;
        count2++;
    }

    file_close(fi);



/*##### CALUCULATE ###################################################################*/

    for(i = 0 ; i < count2 ; ++i)
    {
        drawi = (int)floor(ddk[ i]/step);
        distrib[ drawi] = distrib[ drawi] + 1;
    }

    for(i = 0 ; i < 5/step ; ++i)
    {
        distrib[ i] = distrib[ i] / (double)count2;
    }

/*##### DRAWING GRAPH - WIRE ##########################################################*/

    t_cls();
        /** 0..... BLACK   **/
        /** 1..... BLUE    **/
```

```
        /** 2------ RED     **/
        /** 3------ PURPLE  **/
        /** 4------ GREEN   **/
        /** 5------ CYAN    **/
        /** 6------ YELLOW  **/
        /** 7------ WHITE   **/
        /** 8------ BLACK   **/


    g_color(0,-1,-1,2);

    setclor1=7;

/** outer wire **/
    g_line(25,2,574,397,setclor1,1,0);
    g_line(26,3,573,398,setclor1,1,0);
/** inner wire **/
    g_line(x0,y0,x0+400,y0-300,setclor1,1,0);
    g_line(x0+1,y0+1,x0+401,y0-301,setclor1,1,0);

/*##### DRAWING DISTRIBUTION ##################################################*/

/** initialize **/
    dymax = y0;

    for( i = 0 ; i < 5/step ;++i)
    {
        if(ystd*distrib[ i]-floor(ystd*distrib[ i]) >= 0.5)
            y1 = y0-floor(ystd*distrib[ i])-1;
        else
            y1 = y0-floor(ystd*distrib[ i]);

            g_line(x0+80*step*i,y1,x0+(80*step-1)+80*step*i,y0,setclor1,1,0);
        if(dymax > y1)
            dymax = y1;
    }


/*##### WRITING WRONG MAX & MIN  of leaf kind ##################################*/
    for( j = 0 ; j < count4 ; ++j)
    {
    /******** maximum ********/
    /** input comment **/
        strcpy(chardum,wlkind[ j],strlen(wlkind[ j]) + 1);
        strncat(chardum,"max = ",strlen("max = ") + 1);
        strcpy(inpchar,chardum ,strlen(chardum) + 1);
        for( i = 0 ; i < strlen(inpchar) ; ++i)
        {
            x1 = x0 + 270;
            y1 = y0-240+20*j;
```

```
            gr4_loc(x1+8*i,y1, (int)inpchar[ i]);
        }

        gcvt(wlkmax[ j],4,inpchar);
        for( i = 0 ; i < strlen(inpchar) ; ++i)
        {
            x1 = x0 + 334;
            y1 = y0-240+20*j;
            gr4_loc(x1+8*i,y1, (int)inpchar[ i]);
        }

    /******** minimum *********/
    /** input comment **/
        strcpy(chardum,wlkind[ j],strlen(wlkind[ j]) + 1);
        strncat(chardum,"min = ",strlen("min = ") + 1);
        strcpy(inpchar,chardum ,strlen(chardum) + 1);
        for( i = 0 ; i < strlen(inpchar) ; ++i)
        {
            x1 = x0 + 270;
            y1 = y0-230+20*j;
            gr4_loc(x1+8*i,y1, (int)inpchar[ i]);
        }

        gcvt(wlkmin[ j],4,inpchar);
        for( i = 0 ; i < strlen(inpchar) ; ++i)
        {
            x1 = x0 + 334;
            y1 = y0-230+20*j;
            gr4_loc(x1+8*i,y1, (int)inpchar[ i]);
        }
    }
/****** DRAWING WRONG MAX & MIN ##################################################*/
if(count5 != 0)
{
    /******** maximum *********/
        if( (wddkmax*80) - floor(wddkmax*80) >= 0.5)
            x2=x0 + floor((wddkmax*80)+1);
        else
            x2=x0 + floor(wddkmax*80);

        if( (dymax - 30 ) - (y0 - 290) < 96)
        {
            g_line(x2,y0,x2,y0-290+70,setcolor1,0,1,0x3333);
            g_line(x2,y0-290+70,x2+50-16,y0 - 270 ,setcolor1,0,1,0x3333);
            x2 = x2 + 50;
            dymax = y0 - 180;
        }
        else
        g_line(x2,y0,x2,dymax - 90,setcolor1,0,1,0x3333);
```

```
    else
        x2=x0 + floor(cddkmax*80);

    if( (dymax - 30 ) - (y0 - 290) < 96)
    {
        g_line(x2,y0,x2,y0-290+70,setclor1,0,1,0xf99f);
        g_line(x2,y0-290+70,x2+50-16,y0 - 250 ,setclor1,0,1,0xf99f);
        x2 = x2 + 50;
        dymax = y0 - 200;
    }
    else
    g_line(x2,y0,x2,dymax-50,setclor1,0,1,0xf99f);

/** input comment **/
    strcpy(inpchar, "Cmax" ,strlen("Cmax") + 1);
    for( i = 0 ; i < strlen(inpchar) ; ++i)
    {
        x1 = x2 - 16;
        y1 = dymax - 70;
        gr4_loc(x1+8*i,y1, (int)inpchar[ i]);
    }

    gcvt(cddkmax,4,inpchar);
    for( i = 0 ; i < strlen(inpchar) ; ++i)
    {
        x1 = x2 - 24;
        y1 = dymax - 60;
        gr4_loc(x1+8*i,y1, (int)inpchar[ i]);
    }

/******** minimum *********/
    if( (cddkmin*80)  - floor(cddkmin*80) >= 0.5)
        x2 = x0 + floor(cddkmin*80)+1;
    else
        x2 = x0 + floor(cddkmin*80);

    g_line(x2,y0,x2,dymax-30,setclor1,0,1,0xf99f);

/** input comment **/
    strcpy(inpchar, "Cmin" ,strlen("Cmin") + 1);
    for( i = 0 ; i < strlen(inpchar) ; ++i)
    {
        x1 = x2 - 16;
        if(x1 <= 140)
        x1 = x0 + 2;
        y1 = dymax - 50;
        gr4_loc(x1+8*i,y1, (int)inpchar[ i]);
    }

    gcvt(cddkmin,4,inpchar);
```

```
    /** input comment **/
        strcpy(inpchar, "Wmax" ,strlen("Wmax") + 1);
        for( i = 0 ; i < strlen(inpchar) ; ++i)
        {
            x1 = x2 - 16;
            y1 = dymax - 110;
            gr4_loc(x1+8*i,y1, (int)inpchar[ i]);
        }

        gcvt(wddkmax,4,inpchar);
        for( i = 0 ; i < strlen(inpchar) ; ++i)
        {
            x1 = x2 - 24;
            if(x1 <= 140)
            x1 = x0 + 2;
            y1 = dymax - 100;
            gr4_loc(x1+8*i,y1, (int)inpchar[ i]);
        }

    /******** minimum *********/
        if( (wddkmin*80)  - floor(wddkmin*80) >= 0.5)
            x2 = x0 + floor(wddkmin*80)+1;
        else
            x2 = x0 + floor(wddkmin*80);

        g_line(x2,y0,x2,dymax-70,setcolor1,0,1,0x3333);

    /** input comment **/
        strcpy(inpchar, "Wmin" ,strlen("Wmin") + 1);
        for( i = 0 ; i < strlen(inpchar) ; ++i)
        {
            x1 = x2 - 16;
            y1 = dymax - 90;
            gr4_loc(x1+8*i,y1, (int)inpchar[ i]);
        }

        gcvt(wddkmin,4,inpchar);
        for( i = 0 ; i < strlen(inpchar) ; ++i)
        {
            x1 = x2 - 24;
            if(x1 <= 140)
            x1 = x0 + 2;
            y1 = dymax - 80;
            gr4_loc(x1+8*i,y1, (int)inpchar[ i]);
        }
    }
/****** DRAWING CORRECT MAX & MIN ##########################################*/
/******** maximum *********/
    if( (cddkmax*80)  - floor(cddkmax*80) >= 0.5)
       x2=x0 + floor(cddkmax*80)+1;
```

```
    for( i = 0 ; i < strlen(inpchar) ; ++i)
    {
        x1 = x2 - 24;
        if(x1 <= 140)
        x1 = x0 + 2;
        y1 = dymax - 40;
        gr4_loc(x1+8*i,y1, (int)inpchar[ i]);
    }

/*##### DRAWING MEMORI ##################################################################*/

    for( i = 0 ; i < 6 ; ++i)
    {
        g_line(x0+80*i,y0,x0+80*i,y0+5,setclor1,1,0);
        g_kanji(x0-4+80*i,y0+7,0x0030+i,0,1,7,0);
    }


    for( j = 0 ; j < memcnt ; ++j)
    {
        g_line(x0,y0-300/(memcnt-1)*j,x0-5,y0-300/(memcnt-1)*j,setclor1,1,0);

        gcvt(memstep*j,2,inpchar);
        for( i = 0 ; i < strlen(inpchar) ; ++i)
        {
            x1 = x0 -strlen(inpchar)*8-7;
            y1 = y0-8-300/(memcnt-1)*j;
            gr_loc(x1+8*i,y1, (int)inpchar[ i]);
        }

    }

/*##### WRITNG CONDITION ################################################################*/

    if( (int)(name1[ 5]) > 0x60)
        dummy1 = (int)(name1[ 5]) - 0x20;
    else
        dummy1 = (int)(name1[ 5]);

    inpchar[ 0] = (char)dummy1;
    inpchar[ 1] = 0;
    for( i = 0 ; i < strlen(inpchar) ; ++i)
    {
        x1 = x0+8;
        y1 = y0-290;
        gr_loc(x1+8*i,y1, (int)inpchar[ i]);
    }

    inpchar[ 0] = 'M';
    inpchar[ 1] = 'D';
```

```
inpchar[ 2] = 0;
for( i = 0 ; i < strlen(inpchar) ; ++i)
{
    x1 = x0+16;
    y1 = y0-282;
    gr4_loc(x1+8*i,y1, (int)inpchar[ i]);
}

inpchar[ 0] = '(';
inpchar[ 1] = 0;
strncat(inpchar,jyushu[ seljyush-1],256);
inpchar[ 2] = ')';
inpchar[ 3] = 0;
for( i = 0 ; i < strlen(inpchar) ; ++i)
{
    x1 = x0+32;
    y1 = y0-290;
    gr_loc(x1+8*i,y1, (int)inpchar[ i]);
}

strcpy(inpchar,"Number of Leaf : ",strlen("Number of Leaf : ") + 1);
strncat(inpchar,sellnum[ il],strlen(sellnum[ il]) + 1);

for( i = 0 ; i < strlen(inpchar) ; ++i)
{
    x1 = x0+390-8*strlen(inpchar);
    comp1 = x1;
    y1 = y0-290;
    gr_loc(x1+8*i,y1, (int)inpchar[ i]);
}

strcpy(inpchar,"Parameter No. : "
                        ,strlen("Parameter No. : ") + 1);
strcat(inpchar,selpar);
for( i = 0 ; i < strlen(inpchar) ; ++i)
{
    x1 = x0+390-8*strlen(inpchar);
    comp2 = x1;
    y1 = y0-272;
    gr_loc(x1+8*i,y1, (int)inpchar[ i]);
}

if(comp1 < comp2)
    x1 = comp1;
else
    x1 = comp2;


g_line(x1-2,28,x0+392,y0-254,setcolor1,1,0);
g_line(x1-4,26,x0+394,y0-252,setcolor1,1,0);
```

```
/*##### NAME of AXIS ##################################################################*/

    stccpy(inpchar,"Mahalanobis' Distance",strlen("mahalanobis' distance") + 1);
    for( i = 0 ; i < strlen(inpchar) ; ++i)
    {
        x1 = x0+80;
        y1 = y0+25;
        gr_loc(x1+8*i,y1, (int)inpchar[ i]);
    }

    stccpy(inpchar,"PROBABILITY DISTRIBUTION of MAHALANOBIS' DISTANCE"
    ,strlen("PROBABILITY DISTRIBUTION of MAHALANOBIS' DISTANCE") + 1);
    for( i = 0 ; i < strlen(inpchar) ; ++i)
    {
        x1 = x0;
        y1 = y0+51;
        gr_loc(x1+8*i,y1, (int)inpchar[ i]);
    }



    if(amflag2 == 0)
    {
        while(gcflag3[ 0] != 'y' &&  gcflag3[ 0] != 'Y'
                && gcflag3[ 0] != 'n' &&  gcflag3[ 0] != 'N')
        {
            t_loc(0,25);
            printf("Do You Print ===> < Y >  or  < N >");
            scanf("%s",gcflag3);
        }
    }

    if(gcflag3[ 0] == 'y' ||  gcflag3[ 0] == 'Y')
    {
        g_copy(1);
        stccpy(prg,"lp",strlen("lp") +1);
        forkvp(prg);
    }

    if(amflag2 == 0)
    gcflag3[ 0] = 0;

    g_init();
    t_cls();
    g_screen(3,0,0,1);
    printf("Yx1b[>1h");
    printf("Yx1b[>5h");
```

```
}               /** NO.2 LOOP END **/

}               /** NO.1 LOOP END **/

    printf("¥x1b[>5l");

}



/##### SUBROUTINE ###########################################################*/
/*+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++*/
/*      PROGRAMNAME = RNAME                                                  */
/*      FUNCTION = READING WORDS ON KEYBOARD                                 */
/*      << THE TEAM OF LEAF >>  '89-6-28    PRODUCED by MAKI                 */
/*+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++*/
read_name(name)

    char    name[25];
{
    char    judge[3];                       /* declare various */
    judge[0] = 'x';                         /* initialize */

    while( (judge[0] != 'y') && (judge[0] != 'Y') )
    {
        judge[0] = 'x';                     /* initialize */

        printf ("¥n Enter file name --- ");
        scanf  (" %s",name);                /* read on keyboard */
        printf (" %s",name);

        while( ((judge[0] != 'y') && (judge[0] != 'n')) &&
                                ((judge[0] != 'Y') && (judge[0] != 'N')) )

        {
            printf ("¥n File name OK ? << YES ===> hit y / NO ===> hit n >>  ");

            scanf  (" %s", judge);          /* read on keyboard */
        }
    }
}

/*+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++*/
/* CHARACTER LOCATE ON GRAPHIC SCREEN ++++++++++++++++++++++++++++++++++++++*/
/*+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++*/
gr_loc(x,y, code)
```

```
int x,y;
int code;
{
    code = code - 0x30;

    g_kanji(x,y,0x0030+code,0,1,7,0);
}
/*++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++*/
/* CHARACTER LOCATE ON GRAPHIC SCREEN ++++++++++++++++++++++++++++++++++++++++*/
/*++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++*/
gr4_loc(x,y, code)
int x,y;
int code;
{
    code = code - 0x30;

    g_kanji(x,y,0x0130+code,0,1,7,0);
}
/*++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++*/
/* CHECK BIG WORD  WORD to  SMALL WORD  ++++++++++++++++++++++++++++++++++++++*/
/*++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++*/
bigto_small(strings)
char strings[ 256];
{
int i,j;
int dummy1;
    j = strlen(strings);
    for( i = 0 ; i < j ; ++i )
    {
        if(0x40 < (int)strings[ i] && (int)strings[ i] < 0x60)
        {
            dummy1 = (int)strings[ i]+ 0x20;;
            strings[ i] = (char)dummy1;
        }
        if( i == j - 1)
        strings[ j] = 0;
    }
}

/*++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++*/
/* FILE CLOSE SUBROUTINE ++++++++++++++++++++++++++++++++++++++++++++++++++++++*/
/*++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++*/
file_close(fi)
FILE *fi;
{
int r;
    if((r = fclose( fi)) == -1)
        quit(" Can't Close File ");
}
```

```
/*++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++*/
/* QUIT SUBROUTINE ++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++*/
/*++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++*/

quit(s)
    char *s;
    {
        printf(s);
        exit(1);
     }
```

ユーザーマニュアル素案

ユーザーマニュアル素案

1990.3.25

1．樹木自動検索システムの構成

本システムは、つぎの機器およびソフトウェアからなる。

(1) 構成機器

装置名	仕様・特徴	機　能・大きさ
CCDカメラ (TI-23A)	撮像素子	インターライン方式、 CCD固体撮像素子画素数 266664素子 (有効画素数 492V×512H)
	画面サイズ	1/2インチ水平
	解像度	380本、垂直 350本
	外形寸法	44W×45H×117.5D (mm)
レンズ	絞り	1:2.8
	焦点距離	35mm
画像処理装置 (TAICHI-E)	機能	画像処理機能
	処理速度	1画素　80nsec
コンピュータ	PC-9801 RX2	40MB ハードディスク内蔵
画像記憶装置 (R5000MH)	機能	録画再生機能
	信号方式	ＮＴＳＣ標準方式準拠
	外形寸法	330W×99H×365D (mm)
画像記憶装置 ドライバ (IF-5000RS)	外形寸法	62W×99H×200D (mm)

(2) 判断用ソフトウェア

本プログラムは以下の開発用ソフトウエアを使用している。

開発用言語　　Lattice C (Ver.4.0)
開発用ツール　　TAICHI-E画像処理ライブラリ

2．操作上の制約

(1) 対象の樹種

本システムは、パプア・ニューギニアの熱帯雨林に生息する以下の？？種類の樹木を対象とした検索システムである。

科　名	樹　種
(LEGUMINOSAE)	MANILTOA PLURIJUGA
	PTEROCARPUS INDICUS
	INTSIA BIJUGA
	MANILTOA PSILOGYNE
	ALBIZIA PROCERA
(RUBIACEAE)	TIMONIUS TIMON
(ANACARDIACEAE)	MANGIFERA MINOR
(EUPHORBIACEAE)	PIMELODENDRON AMBOINICUM
(APOCYNACEAE)	ALSTONIA SCHOLARIS
(BORAGINACEAE)	CORDIA DICHOTOMA

(2) 樹葉の管理

本システムで樹種の検索を行う場合は、採取した樹葉を直ちに紙の間に挟み平な場所に数日保管し、乾燥した状態で使用して下さい。

(3) 樹葉の枚数

本システムは、標本数を増すほど正確な樹種の検索ができます。（最大１０枚）

3．起動方法

本システム起動から判断終了までの手順は以下の通りである。

(1) "TEMP"と入力し、改行キーを押す（以下[改行]と記す）ことにより起動する。

入　力　　" TEMP "

↓

[改　行]

↓

表示画面　，　DO YOU NEED BASIC LEMGTH ?

YES　NO

(2) 通常のカメラ入力では、レンズやその他の条件により同じ大きさでも入力された画面上での樹葉の大きさが変化する。現在の入力画像の絶対的大きさを決定するために、ある基準長さを入力する必要がある。
[　DO YOU NEED BASIC LENGTH?　]のメッセージが表れた後、設定する必要があれば[Y E S]に、必要が無ければ[N O]にカーソルを移動し[改行]する。　設定する必要がある場合、次に標準スケールをカメラから入力する。
標準スケールは横10cm、縦6cmの菱形スケールを使用している。

表示画面　　DO YOU NEED BASIC LEMGTH ?

YES　NO

↓

カーソル移動後[改行]　　[Y E S]　　[N O]

↓　　↓

標準スケール入力

↓

表示画面　　IS BASIC LINE DISPLAY NECESSARY ?

YES　NO

(3) 樹葉を設置する際の基準水平線を決定する。
[IS BASIC LINE DISPLAY NECESSARY?] のメッセージにより、必要であれば [ＹＥＳ], そうでなければ [ＮＯ] にカーソルを移動し [改行] する。

表示画面　　IS BASIC LINE DISPLAY NECESSARY ?
　　YES　　NO
↓
カーソル　　[ＹＥＳ]　　[ＮＯ]
移動後[改行]　　↓
　　水平線画面表示

(4) ＜＜ ＳＥＱＵＥＮＣＥ ＲＵＮＮＩＮＧ ＞＞ の表示が出た後、対象樹葉をカメラから入力する。基準線がある場合、樹葉の両端を線上に置くように移動する。設置した後、[改行] して画像を固定し、適性画像であれば更に確認の際、[ＹＥＳ] にカーソル移動し [改行] する。

表示画面　　<< SEQUENCE RUNNING >>
↓
[改　行]
↓
表示画面　　IS LEAF OK ?
　　YES NO
↓
カーソル　　[ＹＥＳ]　　[ＮＯ]
移動後[改行]　　↓　　↓
↓　　↓　　再画像入力
表示画面　　<< SEQUENCE RUNNING >>

(5) 入力された画像をもとに、各特徴パラメータを抽出する。終了後各パラメータ値を表示し終了する。

表示画面　　<< SEQUENCE RUNNING >>
↓
特徴抽出　　各特徴値表示
↓
表示画面　　<< SEQUENCE COMPLETED >>

(6) 対象樹葉が一枚だけでなく、複数の対象樹葉を用いて診断させたい場合、
[DO YOU WISH TO CONTINUE?] のメッセージの後、ＹＥＳと答えた場合には、以上の(4)～(5)の作業を繰り返す。

表示画面　　<< SEQUENCE RUNNING >>
↓
[改　行]
↓
表示画面　　DO YOU WISH TO CONTINUE ?
YES　NO
↓
カーソル移動後[改行]　[ＹＥＳ]　[ＮＯ]
↓　↓
(4)へ　診断処理

(7) 以上の結果をもとに、診断を行い、結果表示を行なう。その後、
[終了しますか？....] のメッセージに従い、”Ｙ”を入力すると終了する。

表示画面　　診断結果表示
RECOGNIZED LEAF = '判断された樹種名'
↓
表示画面　　終了しますか？？
<< YES ==> hit y / NO ==> hit n >>
↓
入力　　' y'　　' n'
↓　↓
終了　継続

(8) 診断を終了する場合は、[ＳＴＯＰ] を押した後、電源を切る。