Memoirs of the Faculty of Engineering, Okayama University, Vol.26, No. 2, pp.25-41, March 1992

The study on the Evaluation of Eye Movement Using Computer Program

Masaaki Matsuda* and Hirokazu Osaki*

(Received January 17 , 1992)

SYNOPSIS

The eye movement has been studied widely, because it gives various information about the characteristics of visual work. Almost all of those studies have concentrated on the distribution of the visual points, the direction and width of the saccadic movement and the fixation time of the measures.

However there are few measures to evaluate the relationship between the visual objects and eye movement.

Recently, the personal computer become portable and this makes it possible to analyze the many data and to show the results graphically on the experimental field.

In this study, we propose a series of methods to evaluate how a subject looks at a visual field using the queue theory, and how often man looks at the central part of the visual field using the statistical two-dimensional ellipse.

We developed the Pascal program on an personal computer(NEC PC9800). This program has 3 functions. One is the measurement of eye movement, the second is the calculation of the proposed measures and the third is the display of the results of these calculations.

* Department of Mechanical Engineering

1. INTRODUCTION

Much reserch has been carried out on the eye movement. In the telecast field, eye movement has been studied to clarify the effect of the New Media using the distribution of the visual points and saccadic span.[1]
In the industrial field, it has been studied to determine the inspection time in relation to the fixation time.[2].

However there are few studies which have developed the measures to evaluate how often man looks at the visual field. Our eye moves rapidly to see the changing surroundings. Therefore, eye movement has been measured at the interval of under 1 second. As a result, the number of data from every experiment is large. To clarify the characteristics of eye movement, the data obtained is required to record, to calculate, and to show the results of the calculation. As these processes take a lot of time, it is necessary to develop a computer program which provides the required functions.

We propose methods to evaluate how a subject looks at a visual field using the queue theory, and how often man looks at the peripheral part in the visual field using the statistical two-dimensional ellipse.[3]

In this paper, we show the Pascal program for the above mentioned methods. This program has 3 functions, which measures eye movement, calculates the proposed measures and displays the results of these calculations.

2. THE EQUIPMENT

The equipment is composed of a VTR, TV monitor, Data Output Unit, and a personal computer.

The coordinates of the visual points(X_i,Y_i) are measured by the Eye Camera, and recorded on the VTR as analog value at every 33ms. The coordinates are transformed into digital values using the Data Output Unit and sent to the personal computer through the communication port(RS232C).

3. ANALYTICAL METHOD

As the apparent distance varies during the work, the coordinates of the visual points are shown as the visual angle calculated using the X or Y axis.

To clarify the characteristics of the movement of visual points, the visual objects are regarded as wickets, and the visual points as queue. Then, the

queue theory is applied to analyze the movement of visual points. The visual field is divided into square meshes. The width of each mesh is determined to be 2° x 2° because the eye movement is classified by 2° variance between the fixation and saccadic movement.[4]

3.1 Traffic Intensity

In the case where the queue theory is applied to the eye movements, the firstly, AR_{ijk} (k=1,2,,,NA_{ij}), ST_{ijk}(k=1,2,,NS_{ij}) are put as the arrival time and the stationary time of the visual points at the mesh(i,j). The mean values of the interval of the arrival time and the stationary time is put as follows.

$$IT=\sum_{k=1}^{NA}(AR_{ijk}-AR_{ijk-1})/(NA_{ij}-1) \quad (1)$$

$$ST=\sum_{k=1}^{NS} ST_{ijk}/NS_{ij} \quad (2)$$

As the visual point passes through the mesh, the stationary time becomes equal to the interval time of the measurement. So, such meshes are omitted from the analysis.

Traffic intensity in mesh(i,j) is given by the following equations.

$$\rho_{ij}=ST_{ij}/IT_{ij} \quad (3)$$

The mean value of the traffic intensity is put as follows.

$$\rho=\sum_{i=1}^{30}\sum_{j=1}^{23}\rho_{ij}/NT \quad (4)$$

Here, NT is the number of meshes that can be defined as traffic intensity.

3.2 Gazing region

The degree of gazing can be evaluated by the number of visual points in the mesh(i,j). In order to estimate the area where a man predominantly looks,

the gazing region(N_n) is defined as the number of the mesh which have more than n-th visual points.

$$N_n=\sum_{k=n}^{m} M_k \qquad (5)$$

Here, M_k : The frequency of meshes which include the k-th visual points.

3.3 Gazing Ratio

To study the distribution of saccadic eye movements, the statistical two-dimensional ellipse is applied. [5].

According to the characteristics of visual works, a subject looks at the peripheral part of the visual field. The two-dimensional ellipse is applied to evaluate how often a subject looks at the periphery.

Visual points are distributed around visual objects, such as VDT and typescript etc. Since the statistical two-dimensional ellipse should approximate the distribution of visual points around one place, the approximation is not valid if the subject looks at 2 isolated visual objects. In this case, the visual field is divided into two areas using the Mode-Method, and the statistical two-dimensional ellipse is calculated for each area as follows.

The coordinates (X_i,Y_i) are shown in the histogram in each axis to determine the boundary by which divides the visual field. The class interval is determined as 2°.

$F_i=f_i/N$ (i=1,2,,k) is put as the relative frequency of the visual points along the X,(Y) axis in i-th class. Here, $N=\Sigma f_i$

Two maximum values of the left(PK_{LMAX}) and right side(PK_{RMAX}) are calculated from the centoroid (G) of the histogram.(6)(7)

The boundary(BD) is the class that shows the minimum value between the two peaks of LMAX and RMAX.

In the event that the distance(D) between the 2 peaks is under 14°, the visual field is not divided because of the diameter of the distinct vision is 14°.

$$PK_{LMAX}=\max\{F_i \mid i=1,2,,G\} \qquad (6)$$

$$PK_{RMAX}=\max\{F_j \mid j=G+1,,M\} \qquad (7)$$

$$BD_k=\min\{F_k \mid k=LMAX,,RMAX\} \qquad (8)$$

$$G=\sum_{i=1}^{k}(i\cdot F_i)\Big/\sum_{i=1}^{k}F_i \quad (9)$$

$$D=|\,LMAX - RMAX\,| \quad (10)$$

The gazing ratio(GR) is defined as follow.

$$GR=N_s/N_1 \quad (11)$$

Here, N_s:The number of the meshes that are distributed in the ellipse.
N_1:The number of the meshes that include the visual points.

As a subject mainly looks at the central part of the visual field, the gazing ratio nearly equals to 1.0.

4. COMPUTER PROGRAM

The program 'RUN' is a set of 3 programs:'MEASURE', 'CALC' and 'DISPLAY'. These 3 programs can be executed as one program by the batch program 'RUN'.

This batch program plays 2 roles. One is to select the transaction 'Analysis of eye movements', or 'Display the measures' or 'Exit from program'. The other role is to execute the remaining programs in a given order.

The diagram of the program is shown in Fig.1.

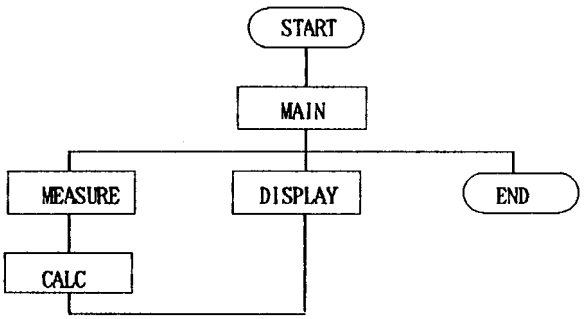

Fig. 1 The diagram of the developed program

4.1 Transaction : 'Analysis of eye movements'

This transaction has 2 functions. One is to load the coordinates of the visual points from VTR, and the other is to calculate the measures.

4.1.1 Program List

(1) Program : 'MEASURE'

This program loads the coordinates of the visual points from VTR and saves them in the text file 'EYEMARK.TXT'. The form of the record is shown in Fig.2. The text file 'EYEMARK.TXT' is a set of records which are composed of the coordinates of the visual points.

The data set (100,100,100,100) is added to identify the end of the file.

·Language : Basic

·Input : Measurement time and interval time.(Keyboard)

·Output : Text file 'EYEMARK.$$$' which includes the coordinates of visual point in time series.

RX_1 , RY_1 , LX_1 , LY_1

: : : :

RX_i , RY_i , LX_i , LY_i

: : : :

RX_n , RY_n , LX_n , LY_n

100 , 100 , 100 , 100

RX_i , RY_i : The coordinate in X and Y axis of the right eye.

LX_i , LY_i : The coordinate in X and Y axis of the left eye.

Each coordinates are expressed by the visual angle.

Fig 2. The form of the text file(EYEMARK.TXT')

(2) Program : 'CALC'

This Pascal program is executed after the program MEASURE. The program list where the traffic intensity is calculated is shown in Table 1.

Firstly, this reads the coordinates of the visual pionts from the text file 'EYEMARK.$$$' and feeds into the variable 'work'. The variable 'work' is overlaid by the variable 'data' which shows the coordinates by mesh.

Secondly, the proposed measures are calculated and feed into the variable 'mesh'. Fig 3 shows the structure of the variable 'mesh'. These are also saved into a file the form of which is shown in Fig.4. The program 'RUN' starts again after this program.

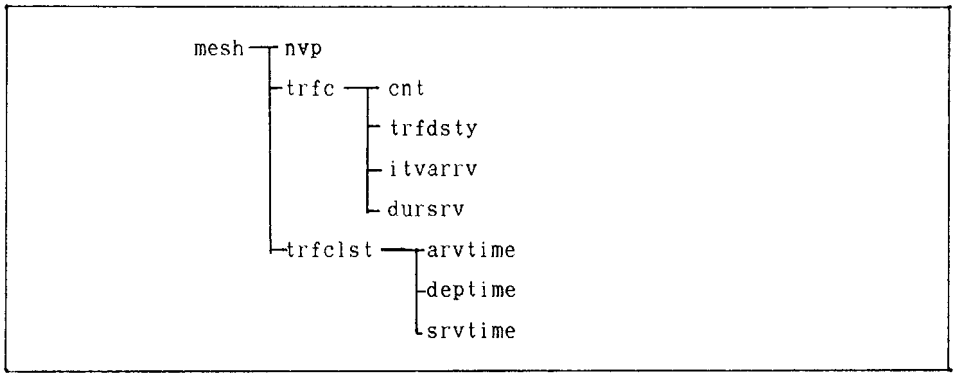

Fig.3 The structure of the variable : 'mesh'

$i, j, n_{ij}, \rho_{ij}, ARV_{ij}, ST_{ij}$

(i,j) : Coordinate of the mesh.
n_{ij} : Number of the visual points in the mesh(i,j)
ρ_{ij} : Traffic intensity
ARV_{ij}: Average of the interval of araival time of the visual points
ST_{ij} : Average of the stationary time

Fig 4 The form of the record created by the program 'CALC'

(i) Gloval Variables list

Name	Type	Size	Defnition
·Work	array	4000x2	The coordinate of the visual points
·Data	array	4000x2	The coordinate of the visual points shown by mesh
·mesh	array	60x45	
·nvp	integer	1	The number of visual points in mesh (i,j).
·trfc	pointer	1	-
·trfdsty	real	1	The value of traffic intensity in mesh (i,j).
·itvarrv	real	1	The value of interval of arrival time in mesh (i,j).
·dursrv	real	1	The value of stationary time in mesh (i,j).
·trfclst	pointer	1	-
·arvtime	real	1	The k-th arrival time in mesh (i,j).
·deptime	real	1	The k-th departure time in mesh (i,j).
·srvtime	real	1	The k-th stationary time in mesh (i,j).

4.2 Transaction : 'Display the measures'

The function of this transaction is to show the measures graphically and print out the value of the measures.

Program list

(1) Program : 'DISPLAY'

Firstly, this Pascal program opens the window and requests selection of the file to be shown on the screen.

Secondly, it calculates the statistical two-dimensional ellipse and feeds into the variable 'mshobj' using the selected file. Fig.5 shows the structure of the variable 'mshobj'.

Finally, it requests the selection of the screen from 'Distribution of the gazing region', 'Distribution of the traffic intensity', and 'Statistical values list'.

This Pascal program feed the measures from a file which is created by the program 'CALC' and shows them graphically. The program list is shown in Table 20.

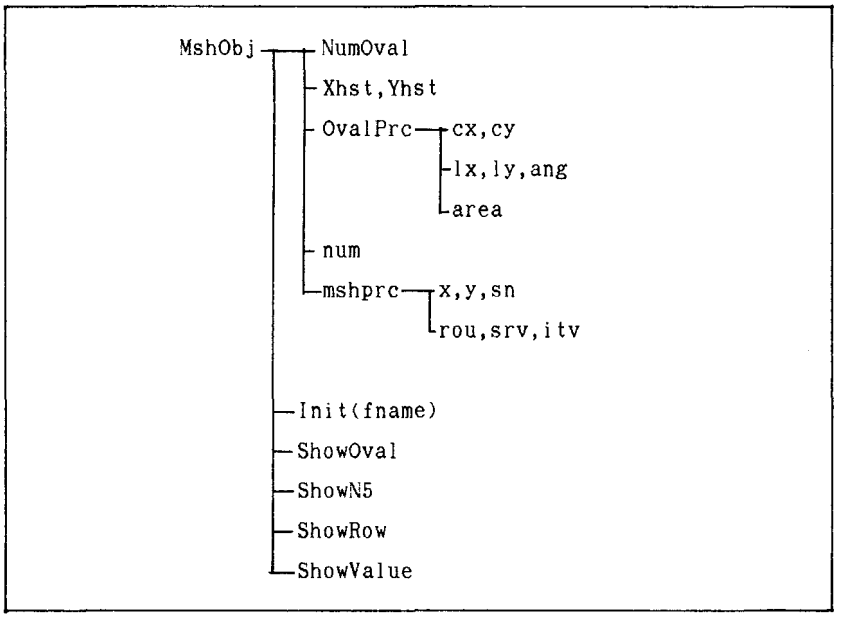

Fig 5 The structure of the variable : 'mshobj'

(i) Valiable List in 'mshobj'

variable type size difinition

·Xhst	array	60	The histogram in X axis.
·Yhst	array	45	The histogram in Y axis.
·Ovalprc	pointer	1	-
·cx,cy	real	1	The center coordinate of the ellipse.
·lx,ly	real	1	The length of the axis of the ellipse.
·area	real	1	The area of the ellipse.
·num	integer	1	The number of visual points.
·mshprc	pointer	1	-
·x,y	integer	1	The coordinate of the mesh
·sn	integer	1	The number of the visual points of mesh(x,y).
·rou	real	1	The traffic intensity of the mesh(x,y).
·itv	real	1	The mean value of the interval of arrival time.
·srv	real	1	The mean value of the stationary time.

(ii) Method list in 'mshobj'

Name	Type	argument	Definition
·Init	constructor	fname	Read the file shown by fname and calculate the statistical two-dimensional ellipse.
·ShowOval	method	none	Show the ellipse.
·ShowRou	method	none	Show the distribution of the traffic intensity which is divided into 5 classes by its value.
·ShowN5	method	none	Show the distribution of the gazing region which is divided into 5 classes by its value.
·ShowValue	method	none	Show the statistical values about the ellipse and the traffic intensity.

5. EXAMPLE

5.1 Visual Work

This method is applied to the assembly work. The subject assembles the small electric parts while sitting ,and the width of the visual field is 36°x48°.

5.2 Application of the program developed

Fig.6 shows the start screen of the main program 'RUN', and '[1] Analysis of eye movements', or '[2] Display of the Analysis Results' or '[3] Exit ' can be chosen.

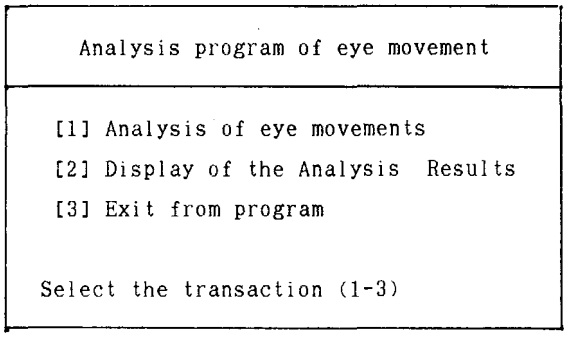

Fig 6 The start screen of the program 'RUN'

In the first step, it is necessary to analyze the eye movements recorded on VTR. So, press the key '1' and begin the program 'MEASURE'. Then the program asks for the measurement time and the interval time of measurement. The coordinates of the visual points are sent to the personal computer through the communication port(RS232C) and saved in the text file 'EYEMARK.TXT'. Then the program 'MEASURE' is over and the program 'CALC' begins to calculate the proposed measures. The program asks for the filename to save the results, and saves them in that file. Then the batch program 'RUN' begins again.

In the second step, '[2] Display of the Analysis Results' is chosen to show the measure values graphically. Then, press key '2' and the program 'DISPLAY' begins. Here, the following commands can be chosen.

[1] Load the results
[2] Distribution of the gazing region
[3] Distribution of the traffic intensity
[4] Cummulative frequency
[5] Statistical values list
[6] End

To load the results of calculations from the file, '[1] Load the results' is chosen.

Fig 7 show the distribution of the gazing region in assembly work. Each mesh is divided into 5 classes by its number of visual points and the statistical two-dimensional ellipse is described at the same time.

As the subject looked around the assembly jig, most of the visual points were distributed around the center of the visual field. The other visual points were distributed at the peripheral part of the visual field where the

tools and parts were.

As 52% of the meshes that the subject looked at had under 4 visual points, it was shown that the subject looked repeatedly at half the region that he looked at.

Fig 8 shows the distribution of traffic intensity and the statistical two-dimensional ellipse. Each mesh was divided into 5 classes by its value of traffic intensity.

The traffic intensity was defined in 33% of the meshes. It was shown that the traffic intensity was under 0.01 in 65% of meshes . So, it was shown that the visual points were served to one third the region in which the visual points existed. It was in about 10% of the region that the visual points achieved high service in assembly work.

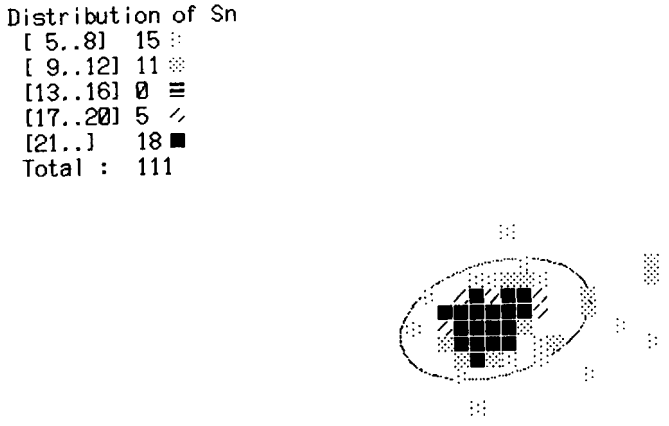

Fig.7 The distribution of the gazing region of assembly work

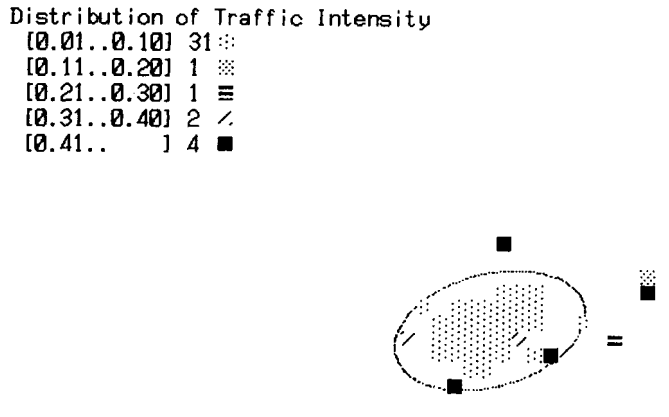

Fig.8 The distribution of the traffic intensity of assembly work

6. CONCLUSION

We proposed a method as to how the subject looked at a visual field, and developed the computer program of this method showing how often the subject looks at the peripheral part of the visual field.

The developed program is applied to the assembly work, and it is shown that the proposed method can clarify the characteristics of the visual work on the basis of eye movement.

References

(1) T.Fukuda, "New electoric media and the human interface", Ergonomics(1990), Vol 33, No 6,687-706

(2) E.D.Megaw, J.Richardson,"Eye movements and industial inspection",Applied Ergonomics(1979),Vol 10 No 3,145-154

(3) M.Matsuda,H.Osaki,"The study on the evaluation of visual work using the distribution of visual points",JIMA(1988),Vol 39,No 2,95-101

(4) T.Inui,"An Introduction to Human Visual Information Processing (in Japanease)", Science,(1990),148-149

(5) H.Collewijn, E.P.Tamminga, "Human Smooth and Saccadic Eye Movements during

Table 1 Program List : 'Calc'

```
Const   MSHSIZEX=2; MSHSIZEY=2; MSHNUMX=60; MSHNUMY=45;  MAXVSPNUM=4000;
Type
    traficrec_type=record
      arvtime,deptime,srvtime:integer
    end;
    traficrec=array[1..500] of traficrec_type;
    trfcptr=record
      cnt:integer;     trfdsty,itvarrv,dursrvc:real;   trfclst:^traficrec;
    end;
  meshrec=record    nvp,nin:integer;  trfc:^trfcptr;  end;
  mesh_type=array[1..MSHNUMX,1..MSHNUMY] of meshrec;
  workarea=array[1..MAXVSPNUM,1..2] of real;
  vspdatary=array[1..MAXVSPNUM,1..2] of integer;
  sttvsprec=record  numvsp:integer; minvspx,minvspy,avevspx,avevspy:real;  end;
  sttmshrec=record  cntrx,cntry,sftx,sfty,avemshx,avemshy  : integer;      end;
  rangerectype=record   minmeshx,minmeshy,maxmeshx,maxmeshy:integer;       end;

  Procedure   UseVram;                    forward;
  Procedure   ReadDataFile(fname:dirstr); forward;
  Procedure   DataConvert;                forward;
  Procedure   CalcSn;                     forward;
  Procedure   CalcTraffic;                forward;
  Procedure   PrintFile;                  forward;

Procedure CalcMain;
    work : workarea absolute $B800:0;
    data : absolute work;
    mesh : meshtype;
  begin
    UseVram;
    ReadDataFile('Eyemark.dps');
    DataConvert;
    CalcSn;
    CalcTraffic;
    PrintFile;
  end;

Procedure CalcSn;
var i,j,n:integer;  newx,newy,oldx,oldy:byte;
begin
  with rangerec do begin
    for i:=MINMESHX to MAXMESHX do  for j:=MINMESHY to MAXMESHY do begin
      with MESH[i,j] do begin   nvp:=0; nin:=0; trfc:=nil; end;
    end;
  end;
  oldx:=255; oldy:=255;
  for n:=1 to STTVSP.numvsp do  begin
    newx:=DATA[n,1];  newy:=DATA[n,2]; inc(MESH[newx,newy].nvp);
    if not(newx=oldx) or not(newy=oldy) then inc(MESH[newx,newy].nin);
    oldx:=newx; oldy:=newy;
  end;
end;

Procedure CalcTraffic;
  procedure CalcTrafficMain;
  var loop,ix,iy:integer;   ita,ta,ts:real;
  begin
    for ix:=rangerec.MINMESHX to rangerec.MAXMESHX do
```

```
  for iy:=rangerec.MINMESHY to rangerec.MAXMESHY do  with MESH[ix,iy] do begin
  if trfc<>nil then begin
      ta:=0; ts:=0;
      with trfc^ do begin
        for loop:=1 to cnt do  ts:=ts+trfclst^[loop].srvtime;
        for loop:=1 to cnt-1 do begin
          ita:=trfclst^[loop+1].arvtime-trfclst^[loop].arvtime;
          ta:=ta+ita;
        end;
        itvarrv:=ta/(cnt-1); dursrvc:=ts/cnt;  trfdsty:=dursrvc/itvarrv;
      end;
    end;
  end; { ix,iy-for }
end; {of Proc }

procedure CalcTime;
var newx,newy,oldx,oldy,ix,iy:byte;  ct,loop:integer;  size:word;
begin
  with rangerec do begin
    for ix:=MINMESHX to MAXMESHX do  for iy:=MINMESHY to MAXMESHY do
    with MESH[ix,iy] do begin
      if (nvp>=3) and (nin>1) then begin
        size:=sizeof(traficrec_type)*nin;
        new(trfc); getmem(trfc^.trfclst,size);
        trfc^.cnt:=0;
      end else trfc:=nil;
    end;
  end;
  oldx:=DATA[1,1]; oldy:=DATA[1,2];
  if MESH[oldx,oldy].trfc<>nil then with MESH[oldx,oldy].trfc^ do begin
    inc(cnt); trfclst^[cnt].arvtime:=1;
  end;
  for loop:=2 to STTVSP.numvsp-1 do begin
    newx:=DATA[loop,1]; newy:=DATA[loop,2];
    if (newx<>oldx) or (newy<>oldy) then begin
      if MESH[newx,newy].trfc<>nil then  with  MESH[newx,newy].trfc^ do begin
        inc(cnt); trfclst^[cnt].arvtime:=loop;
      end;
      if MESH[oldx,oldy].trfc<>nil then with MESH[oldx,oldy].trfc^ do
      with  trfclst^[cnt] do begin
        deptime:=loop-1; srvtime:=deptime-arvtime;
        if srvtime=0 then dec(MESH[oldx,oldy].trfc^.cnt);
      end
    end else begin  oldx:=newx; oldy:=newy end;  (* if (newx.. ) *)
  end; (*  for loop *)
  newx:=DATA[STTVSP.numvsp,1]; newy:=DATA[STTVSP.numvsp,2];
  with MESH[oldx,oldy] do begin
    if trfc<>nil then with trfc^ do begin
      if trfclst^[cnt].arvtime<STTVSP.numvsp-1 then begin
        with trfclst^[cnt] do begin
          deptime:=STTVSP.numvsp;  srvtime:=deptime-arvtime;
        end;
      end else dec(cnt);
    end;
  end;
  with rangerec do begin
    for ix:=MINMESHX to MAXMESHX do  for iy:=MINMESHY to MAXMESHY do
    with MESH[ix,iy] do begin
      if (trfc<>nil) and (trfc^.cnt<2) then begin
        trfc^.cnt:=0;  dispose(trfc); trfc:=nil;
```

```
        end;
      end;
    end;
  end;
begin
  CalcTime;   CalcTrafficMain;
end;
```

Table.2 Program List : 'DISPLAY'

```
Type
    mshrectype=record  x,y,sn : integer;   rou,itv,srv : real;  end;
    mshtemp=array[1..1000] of mshrectype;
    mshtempptrtype=^mshtemp;
    mshobjtype=object
        size       : longint;   num     : integer;
        sftx, sfty : integer;
        fname   : dirstr; mshprc  : ^char; mshdumy : mshtempptrtype;
      constructor init(filename:dirstr);
      destructor  done; virtual;
    end;
  HstRecType=record
    list   : array[0..60] of integer;
    PosDiv : integer;  IsDiv  : boolean;  DivRatio : real;
  end;
  OvalRecType=record   cx,cy, lx,ly,ang,  area , rc : real;   end;
  OvalPrcType=^OvalRecType;
  OvalTempType=array[1..5] of OvalRecType;
  OvalTempptrType=^OvalTempType;
  MshOvalObjType=object(MshObjType)
      NumOval : integer;
      Xhst, Yhst    :  HstRecType;
      Ovalprc :  OvalPrcType;   OvalDumy:  OvalTempPtrType;
    Constructor Init(filename:dirstr);
    Destructor  Done; Virtual;
    Procedure   Calc;
    Procedure   ShowOval;
    Procedure   ShowN5;
    Procedure   ShowRou;
   end;

Procedure MshOvalObjType.Calc;
  procedure CalcHst(var Hst : HstRectype);
  var i,ix,iy : integer;
    begin
      for i:=1 to num do begin
        ix:=mshdumy^[i].x + sftx;   iy:=mshdumy^[i].y + sfty;
        if mshdumy^[i].sn > 5 then begin
          Xhst.list[ix]:=Xhst.list[ix]+mshdumy^[i].sn;
          Yhst.list[iy]:=Yhst.list[iy]+mshdumy^[i].sn;
        end;
      end;
    end;
  procedure CalcDivPos(var hst : hstrectype);
  var  lmaxpos,midpos,rmaxpos  : integer;
    procedure calc4p;
      type zerorectype=record
          zlist : array[1..100] of integer;   znum : integer;
        end;
    var i,n,max : integer;  t : real;   zrec : zerorectype;
      begin
```

```
    fillchar(zrec,sizeof(zrec),#0);  t:=0; n:=0;
    with hst do begin
      for i:=1 to 60 do  begin n:=n + list[i];  t:=t + list[i] * i; end;
      midpos:=round(t / n);  max:=0;
      for i:=midpos downto 1 do
        if list[i] > max then begin max:=list[i]; lmaxpos:=i; end;
      max:=0;
      for i:=midpos to 60 do
        if list[i] > max then begin max:=list[i]; rmaxpos:=i; end;
      for i:=lmaxpos to rmaxpos do  with zrec do
        if list[i]=0 then begin inc(znum); zlist[znum]:=i;    end;
      t:=0;
      if zrec.znum > 0 then begin
        with zrec do  for i:=1 to znum do t:=t+zlist[i];
        PosDiv:=round(t /  zrec.znum);
      end;
      isdiv:=not(zrec.znum=0);
    end;
  end; (* proc calc 4p *)

procedure CheckDiv;
const _crtcl=4.0;
var t1,t2 : real;  i : integer;
  begin
    t1:=0; t2:=0;
    if not(hst.isdiv) then exit;
    with hst do begin
      for i:=PosDiv downto 1 do t1:=t1+list[i];
      for i:=PosDiv to 60    do t2:=t2+list[i];
      if t1 > t2 then DivRatio:=t1/t2 else DivRatio:=t2/t1;
      IsDiv:=DivRatio < _crtcl;
    end;
  end;
begin  (*  main of CalcDivPos *)
  calc4p;   checkdiv;
end;
procedure InitOvalprc;
begin
  GetMem(Ovalprc,num*sizeof(ovalrectype));
  ovaldumy:=OvalTempPtrType(Ovalprc);
end;
procedure CalcMain;
const FV=2.45;
type  statrectype=record
        n  : integer; xavg , yavg, xstd,ystd,xystd,  r : real;
      end;
var statlist : array[1..2] of statrectype;
  procedure Step1(i:integer; var stat:statrectype);
    begin
      with mshdumy^[i],stat do begin
        if sn > 5 then begin
          xavg:=xavg + x*sn; xstd:=xstd+x*x*sn; xystd:=xystd + x*y*sn;
          yavg:=yavg + y*sn; ystd:=ystd+y*y*sn;   n:=n+sn;
        end;
      end;
    end;
  procedure Step2(var stat : statrectype);
    begin
      if stat.n =0 then exit;
      with stat do begin
```

```
        if xstd*ystd<>0 then r:=xystd/(xstd*ystd) else r:=0;
      end;
    end;
  procedure Step3(var stat : statrectype; var ovalrec : ovalrectype);
    procedure Get_Axis(xstd,ystd,xystd,la,lb:real; Var ang,lx,ly:real);
      begin
        lx:=FV * GetRMax(la,lb); ly:=FV * GetRMin(la,lb);
        if (xystd > 0)  and (xstd < ystd) then ang:=ang+90;
        if (xystd < 0)  and (xstd < ystd) then ang:=ang-90;
      end;

  var sinx , cosx ,la,lb : real;
    begin
      with stat,ovalrec  do begin
        cx:=xavg; cy:=yavg;
        if xstd*ystd<>0 then begin
          ang:=0.5*arctan(2*xystd/(xstd*xstd-ystd*ystd));
          sinx:=sin(ang); cosx:=cos(ang);
          la:=xstd*xstd*cosx*cosx + 2*xystd*cosx*sinx + ystd*ystd*sinx*sinx;
          lb:=xstd*xstd*sinx*sinx - 2*xystd*cosx*sinx + ystd*ystd*cosx*cosx;
          la:=sqrt(la); lb:=sqrt(lb);  ang:=ang*180/Pi;
          Get_Axis(xstd,ystd,xystd,la,lb,ang,lx,ly);
          area:=Pi*lx*ly;  rc:=r;
        end else
          if ystd=0 then  lx:=xstd*FV  else begin ly:=ystd*FV; ang:=90 end;
      end; (* with stat,ovl  *)
    end;

  var ovrec : ovalrectype; i : integer;
  begin
    fillchar(statlist,sizeof(statlist),#0);
    for i:=1 to num do  with mshdumy^[i] do
      if (x>(xhst.PosDiv-sftx)) or (y>(yhst.PosDiv-sfty)) then
        Step1(i,statlist[2]) else  Step1(i,statlist[1]);
    Step2(statlist[1]);  Step3(statlist[1],ovrec);
    ovaldumy^[1]:=ovrec;
    if numoval=2 then begin
      Step2(statlist[2]); Step3(statlist[2],ovrec); ovaldumy^[2]:=ovrec;
    end;
  end;
begin
  CalcHst(Xhst);    CalcHst(Yhst);  CalcDivPos(Xhst); CalcDivPos(Yhst);
  if Xhst.IsDiv or  Yhst.IsDiv then NumOval:=2 else NumOval:=1;
  InitOvalPrc;
  if Xhst.IsDiv and Yhst.IsDiv then
    if Xhst.DivRatio > Yhst.DivRatio then  Xhst.IsDiv:=false
                                     else  Yhst.IsDiv:=false;
  calcmain;
end; (* MshOvalObjType *)
```