Cellular Automaton Graphics(12)

Kikuchi Morio

Abstract :

In a figure which is composed of hyper surfaces of a hyper cube in space from 3 dimensions, we do a tiling.

1. Hyper cube

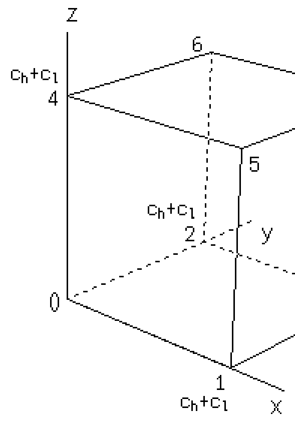

Figure 1

Figure 1 shows a hyper cube in 3 dimensional space namely cube. This time, we do a tiling not inside hyper cube but on hyper surface.

We define the name of hyper surfaces which compose the figure in Figure 1 as follows:

・□ 0132, □ 4576 : xy×2
・□ 0154, □ 2376 : xz×2
・□ 0264, □ 1375 : yz×2

Each is in parallel with xy plane, xz plane, yz plane.

In 4 dimensions

・xyz×2
・xyu×2

・xzu×2
・yzu×2

In 5 dimensions

・xyzu×2
・ ・xyzv×2

・xyuv×2
・xzuv×2
・yzuv×2

In 6 dimensions

・xyzuv×2
・ ・xyzuw×2

・ ・xyzvw×2

- xyuvw×2
- xzuvw×2
- yzuvw×2

In 7 dimensions

- xyzuvw×2
- · xyzuvr×2

- · xyzuwr×2
- · xyzvwr×2
- xyuvwr×2
- xzuvwr×2
- yzuvwr×2

In 8 dimensions

- xyzuvwr×2
- · xyzuvws×2

- · xyzuvrs×2
- · xyzuwrs×2
- · xyzvwrs×2
- xyuvwrs×2
- xzuvwrs×2
- yzuvwrs×2

In 9 dimensions

- xyzuvwrs×2
- · xyzuvwrt×2

- · xyzuvwst×2
- · xyzuvrst×2
- · xyzuwrst×2
- · xyzvwrst×2
- xyuvwrst×2
- xzuvwrst×2
- yzuvwrst×2

In 10 dimensions

- xyzuvwrst×2
- · xyzuvwrso×2

- · xyzuvwrto×2
- · xyzuvwsto×2
- · xyzuvrsto×2
- · xyzuwrsto×2
- · xyzvwrsto×2
- xyuvwrsto×2
- xzuvwrsto×2
- yzuvwrsto×2

2. Composition

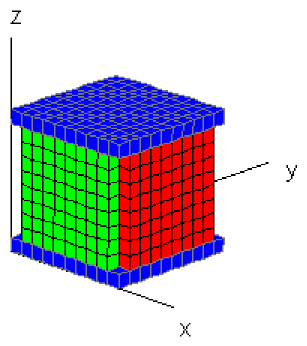

Figure 2

If dimension number increases, a problem that we can not calculate the number of points to be painted easily arises. Therefore, we cut a hyper surface of hyper cube like Figure 2. We call hyper surface xz×2, yz×2 in this case of which length of side is short by 2 SMALL.

- xy×2 : LARGE(L)
- xz×2 : SMALL(S)
- yz×2 : SMALL(S)

The length of side of LARGE, SMALL is as follows:

- `cpwidth*(2*delta_x+1)`≡RESO
- LARGE : RESO
- SMALL : RESO−2

If dimension number increases, the number of hyper surfaces becomes larger, so we restrict the number of hyper surfaces which we use to four. We use not hyper surfaces to which the mark ・・ is put but hyper surfaces to which the mark ・ is put.
In 3 dimensions

- xy×2
- xz×2
- yz×2

In 4 dimensions

- xyz×2
- xyu×2
- xzu×2
- yzu×2

In 5 dimensions

- xyzu×2
- xyuv×2
- xzuv×2
- yzuv×2

In 6 dimensions

- xyzuv×2
- xyuvw×2
- xzuvw×2
- yzuvw×2

In 7 dimensions

- xyzuvw×2
- xyuvwr×2
- xzuvwr×2
- yzuvwr×2

In 8 dimensions

- xyzuvwr×2
- xyuvwrs×2
- xzuvwrs×2
- yzuvwrs×2

In 9 dimensions

- xyzuvwrs×2
- xyuvwrst×2
- xzuvwrst×2
- yzuvwrst×2

In 10 dimensions

- xyzuvwrst×2
- xyuvwrsto×2
- xzuvwrsto×2
- yzuvwrsto×2

If we adopt the choice, the number of points to be painted is

- `pow(RESO,2)*2+pow(RESO-2,3-1)*4` : 3 dimensions
- `pow(RESO,2)*2+pow(RESO-2,DMS-1)*6` : from 4 dimensions

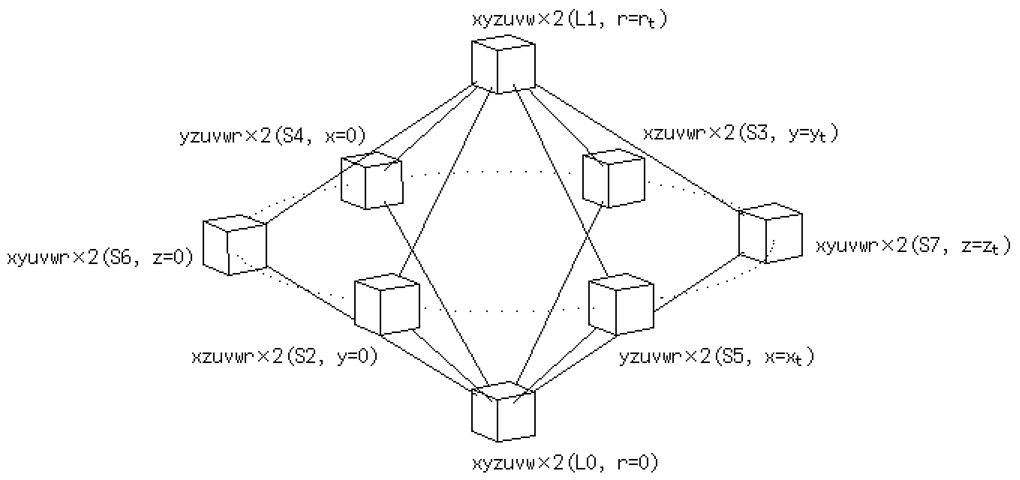

Figure 3

Figure 3 shows the route map of painting when dimension number is 7. The value of the constant in the figure is as follows:

- x_t : RESO−1
- y_t : RESO−1
- z_t : RESO−1
- u_t : RESO−1
- v_t : RESO−1

・w_t : RESO−1
・r_t : RESO−1

3. Painting algorithm
For example, when the following plane algorithm is used in practice

・(c1,c3,c2,c4)≡en_xy

in 4 dimensions for example, the full description is as follows:

```
if((c1==ca || c3==ca || c2==ca || c4==ca)&&(Nz_==Nz && Nu_==Nu)){
jmax=en_xy();                           /* xy */
}
```

The more dimension number increases, the more the second term in if() becomes larger. In 7 dimensions

・`Nz_==Nz && Nu_==Nu && Nv_==Nv && Nw_==Nw && Nr_==Nr`

This time, we add some expression statements.

```
else if((c3==ca || c5==ca || c4==ca || c6==ca)&& (Nz_==Nz && Nu_==Nu)){
jmax=en_yz();                           /* yz */
if(i==1) {if(algo%2==0) algo++;else algo--;}
}
```

CW, CCW correspond to variable algo=even number, odd number respectively. The added if statement does switching of CW, CCW. i is painting point number.
We express the above (else) if statement like the following:

・"en_yz();− − −;PM_algo(1);"

Besides, the following is also adopted.

```
int pm_algo(int algo)
{
if(algo%2==0) algo++;else algo--;

return algo;
}/** pm_algo **/
```

・`else if(i>=4 && (algo=pm_algo(algo))<0) ;`≡"i_geq_4_PM_algo"
・`alg=pm_algo(alg);`≡"i_eq_4_PM_algo"

The following is a painting algorithm when in 7 dimensions, CPMAX is 8 and seed points are fixed like the last Figure 5(b) in L0 or L1 in Figure 3.

・xy(alg=0, 1) :
・i=0, 1 :
・`algo=alg;`
・"en_xy();− − −;− − −;"
・"en_yz();− − −;PM_algo(0+1);"
・"en_zx();− − −;PM_algo(0+1);"

・"en_xu();− − −;PM_algo(0+1);"
・"en_yu();− − −;PM_algo(0+1);"
・"en_zu();− − −;− − −;"

· "en_xv();– – –;PM_algo(0+1);"
· "en_yv();– – –;PM_algo(0+1);"
· "en_zv();– – –;– – –;"
· "en_uv();– – –;– – –;"

· "en_xw();– – –;PM_algo(0+1);"
· "en_yw();– – –;PM_algo(0+1);"
· "en_zw();– – –;– – –;"
· "en_uw();– – –;– – –;"
· "en_vw();– – –;– – –;"

· "en_xr();– – –;PM_algo(0+1);"
· "en_yr();– – –;PM_algo(0+1);"
· "en_zr();– – –;– – –;"
· "en_ur();– – –;– – –;"
· "en_vr();– – –;– – –;"
· "en_wr();– – –;– – –;"

· i=2, 3 :
· `algo=alg;`
· "en_xy();– – –;– – –;"
· "en_zx();– – –;PM_algo(0+2);"
· "en_yz();– – –;PM_algo(0+3);"

· "en_yu();– – –;PM_algo(0+2);"
· "en_xu();– – –;PM_algo(0+3);"
· "en_zu();– – –;– – –;"

· "en_yv();– – –;PM_algo(0+2);"
· "en_xv();– – –;PM_algo(0+3);"
· "en_zv();– – –;– – –;"
· "en_uv();– – –;– – –;"

· "en_yw();– – –;PM_algo(0+2);"
· "en_xw();– – –;PM_algo(0+3);"
· "en_zw();– – –;– – –;"
· "en_uw();– – –;– – –;"
· "en_vw();– – –;– – –;"

· "en_yr();– – –;PM_algo(0+2);"
· "en_xr();– – –;PM_algo(0+3);"
· "en_zr();– – –;– – –;"
· "en_ur();– – –;– – –;"
· "en_vr();– – –;– – –;"
· "en_wr();– – –;– – –;"

· "i_eq_4_PM_algo"

· i=4, 5 :
· `algo=alg;`
· "en_xy();– – –;– – –;"
· "en_zx();– – –;PM_algo(4+0);"
· "en_yz();– – –;PM_algo(4+0);"

- "en_yu();– – –;PM_algo(4+0);"
- "en_xu();– – –;PM_algo(4+0);"
- "en_zu();– – –;– – –;"

- "en_yv();– – –;PM_algo(4+0);"
- "en_xv();– – –;PM_algo(4+0);"
- "en_zv();– – –;– – –;"
- "i_geq_4_PM_algo"
- "en_uv();– – –;– – –;"

- "i_geq_4_PM_algo"
- "en_yw();– – –;PM_algo(4+0);"
- "en_xw();– – –;PM_algo(4+0);"
- "en_zw();– – –;– – –;"
- "i_geq_4_PM_algo"
- "en_uw();– – –;– – –;"
- "en_vw();– – –;– – –;"

- "i_geq_4_PM_algo"
- "en_yr();– – –;PM_algo(4+0);"
- "en_xr();– – –;PM_algo(4+0);"
- "en_zr();– – –;– – –;"
- "i_geq_4_PM_algo"
- "en_ur();– – –;– – –;"
- "en_vr();– – –;– – –;"
- "en_wr();– – –;– – –;"

- i=6, 7 :
- `algo=alg;`
- "en_xy();– – –;– – –;"
- "en_yz();– – –;PM_algo(4+2);"
- "en_zx();– – –;PM_algo(4+3);"

- "en_xu();– – –;PM_algo(4+2);"
- "en_yu();– – –;PM_algo(4+3);"
- "en_zu();– – –;– – –;"

- "en_xv();– – –;PM_algo(4+2);"
- "en_yv();– – –;PM_algo(4+3);"
- "en_zv();– – –;– – –;"
- "i_geq_4_PM_algo"
- "en_uv();– – –;– – –;"

- "i_geq_4_PM_algo"
- "en_xw();– – –;PM_algo(4+2);"
- "en_yw();– – –;PM_algo(4+3);"
- "en_zw();– – –;– – –;"
- "i_geq_4_PM_algo"
- "en_uw();– – –;– – –;"
- "en_vw();– – –;– – –;"

- "i_geq_4_PM_algo"
- "en_xr();– – –;PM_algo(4+2);"
- "en_yr();– – –;PM_algo(4+3);"

· "en_zr();− − −;− − −;"
· "i_geq_4_PM_algo"
· "en_ur();− − −;− − −;"
· "en_vr();− − −;− − −;"
· "en_wr();− − −;− − −;"

If en_yz() is at the head of the painting algorithm, the three lines from the head are as follows:

· yz(alg=2, 3) :
· i=0, 1 :
· `algo=alg;`
· "en_yz();− − −;PM_algo(0+1);"
· "en_zx();− − −;PM_algo(0+1);"
· "en_xy();− − −;− − −;"

· i=2, 3 :
· `algo=alg;`
· "en_zx();− − −;PM_algo(0+2);"
· "en_yz();− − −;PM_algo(0+3);"
· "en_xy();− − −;− − −;"

· "i_eq_4_PM_algo"

· i=4, 5 :
· `algo=alg;`
· "en_zx();− − −;PM_algo(4+0);"
· "en_yz();− − −;PM_algo(4+0);"
· "en_xy();− − −;− − −;"

· i=6, 7 :
· `algo=alg;`
· "en_yz();− − −;PM_algo(4+2);"
· "en_zx();− − −;PM_algo(4+3);"
· "en_xy();− − −;− − −;"

If en_zx() is at the head of the painting algorithm, the three lines from the head are as follows:

· zx(alg=4, 5) :
· i=0, 1 :
· `algo=alg;`
· "en_zx();− − −;PM_algo(0+1);"
· "en_yz();− − −;PM_algo(0+1);"
· "en_xy();− − −;− − −;"

· i=2, 3 :
· `algo=alg;`
· "en_yz();− − −;PM_algo(0+3);"
· "en_zx();− − −;PM_algo(0+2);"
· "en_xy();− − −;− − −;"

· "i_eq_4_PM_algo"

· i=4, 5 :
· `algo=alg;`

- "en_yz();− − −;PM_algo(4+0);"
- "en_zx();− − −;PM_algo(4+0);"
- "en_xy();− − −;− − −;"

- i=6, 7 :
- `algo=alg;`
- "en_zx();− − −;PM_algo(4+3);"
- "en_yz();− − −;PM_algo(4+2);"
- "en_xy();− − −;− − −;"

In the above, we use the three groups xy, yz, zx joining them together. We are free to join the groups together. For example

- zx : alg=0, 1
- yz : alg=2, 3

4. Switching of CW, CCW

Switching of CW, CCW can be done also by a means except PM_algo.

```
else if((c3==ca || c5==ca || c4==ca || c6==ca)&& (Nz_==Nz && Nu_==Nu)){
jmax=en_yz();                         /* yz */
/*if(i==1) {if(algo%2==0) algo++;else algo--;}*/
if(i==0) ui=algo;
algo=PM_algo_en(1,algo);
}
```

We express the above (else) if statement like the following:

- "en_yz();ui=algo;PM_algo_en(1);"

Painting algorithm becomes like the following:

- xy(alg=0, 1) :
- `algo=alg;`
- i=0, 1 :
- "en_xy();ui=algo;− − −;"
- "en_yz();ui=algo;PM_algo_en(1);"
- "en_zx();ui=algo;PM_algo_en(2);"

- "en_xu();ui=algo;PM_algo_en(3);"
- "en_yu();ui=algo;PM_algo_en(4);"
- "en_zu();ui=algo;− − −;"

- "en_xv();ui=algo;PM_algo_en(6);"
- "en_yv();ui=algo;PM_algo_en(7);"
- "en_zv();ui=algo;− − −;"
- "en_uv();ui=algo;− − −;"

- "en_xw();ui=algo;PM_algo_en(10);"
- "en_yw();ui=algo;PM_algo_en(11);"
- "en_zw();ui=algo;− − −;"
- "en_uw();ui=algo;− − −;"
- "en_vw();ui=algo;− − −;"

- "en_xr();ui=algo;PM_algo_en(15);"
- "en_yr();ui=algo;PM_algo_en(16);"

- "en_zr();ui=algo;− − −;"
- "en_ur();ui=algo;− − −;"
- "en_vr();ui=algo;− − −;"
- "en_wr();ui=algo;− − −;"

- i=2, 3 :
- "en_xy();− − −;− − −;"
- "en_zx();− − −;PM_algo_en(1);"
- "en_yz();− − −;PM_algo_en(2);"

- "en_yu();− − −;PM_algo_en(3);"
- "en_xu();− − −;PM_algo_en(4);"
- "en_zu();− − −;− − −;"

- "en_yv();− − −;PM_algo_en(6);"
- "en_xv();− − −;PM_algo_en(7);"
- "en_zv();− − −;− − −;"
- "en_uv();− − −;− − −;"

- "en_yw();− − −;PM_algo_en(10);"
- "en_xw();− − −;PM_algo_en(11);"
- "en_zw();− − −;− − −;"
- "en_uw();− − −;− − −;"
- "en_vw();− − −;− − −;"

- "en_yr();− − −;PM_algo_en(15);"
- "en_xr();− − −;PM_algo_en(16);"
- "en_zr();− − −;− − −;"
- "en_ur();− − −;− − −;"
- "en_vr();− − −;− − −;"
- "en_wr();− − −;− − −;"

- i=4, 5 :
- "en_xy();− − −;− − −;"
- "en_zx();− − −;PM_algo_en(1);"
- "en_yz();− − −;PM_algo_en(2);"

- "en_yu();− − −;PM_algo_en(3);"
- "en_xu();− − −;PM_algo_en(4);"
- "en_zu();− − −;− − −;"

- "en_yv();− − −;PM_algo_en(6);"
- "en_xv();− − −;PM_algo_en(7);"
- "en_zv();− − −;− − −;"
- "en_uv();− − −;− − −;"

- "en_yw();− − −;PM_algo_en(10);"
- "en_xw();− − −;PM_algo_en(11);"
- "en_zw();− − −;− − −;"
- "en_uw();− − −;− − −;"
- "en_vw();− − −;− − −;"

- "en_yr();− − −;PM_algo_en(15);"
- "en_xr();− − −;PM_algo_en(16);"

・"en_zr();− − −;− − −;"
・"en_ur();− − −;− − −;"
・"en_vr();− − −;− − −;"
・"en_wr();− − −;− − −;"

・i=6, 7 :
・"en_xy();− − −;− − −;"
・"en_yz();− − −;PM_algo_en(1);"
・"en_zx();− − −;PM_algo_en(2);"

・"en_xu();− − −;PM_algo_en(3);"
・"en_yu();− − −;PM_algo_en(4);"
・"en_zu();− − −;− − −;"

・"en_xv();− − −;PM_algo_en(6);"
・"en_yv();− − −;PM_algo_en(7);"
・"en_zv();− − −;− − −;"
・"en_uv();− − −;− − −;"

・"en_xw();− − −;PM_algo_en(10);"
・"en_yw();− − −;PM_algo_en(11);"
・"en_zw();− − −;− − −;"
・"en_uw();− − −;− − −;"
・"en_vw();− − −;− − −;"

・"en_xr();− − −;PM_algo_en(15);"
・"en_yr();− − −;PM_algo_en(16);"
・"en_zr();− − −;− − −;"
・"en_ur();− − −;− − −;"
・"en_vr();− − −;− − −;"
・"en_wr();− − −;− − −;"

The following is the handling procedure of switching of CW, CCW by PM_algo_en.

(1)At painting point number i=0, we memorize algo to ui.
(2)If painting point number i>0, in PM_algo_en, we do a handling according to the argument.

・argument 1, 3, 6, 10, 15, 21, 28, 36, 45, ... : 01100101
・argument 2, 4, 7, 11, 16, 22, 29, 37, 46, ... : 01010110

The correspondence between painting point number and two rows of handling number, 01100101 and 01010110 is as follows:

・i=0 : 0
・i=1 : 1
・i=2 : 1
・i=3 : 0
・i=4 : 0
・i=5 : 1
・i=6 : 0
・i=7 : 1

・i=0 : 0
・i=1 : 1
・i=2 : 0

· i=3 : 1
· i=4 : 0
· i=5 : 1
· i=6 : 1
· i=7 : 0

If handling number is 0, algo is handled like the following:

· If ui%2 is 0, algo is handled like algo%2 becoming 0.
· If ui%2 is 1, algo is handled like algo%2 becoming 1.

If handling number is 1, algo is handled like the following:

· If ui%2 is 0, algo is handled like algo%2 becoming 1.
· If ui%2 is 1, algo is handled like algo%2 becoming 0.

However, Δalgo=±1 must be chosen in order for algo/2 not to change.
If en_yz(), en_zx() is at the head, we add 0, −1 to the argument groups.

· argument 0, 1, 3, 6, 10, 15, 21, 28, 36, 45, ... : 01100101
· argument −1, 2, 4, 7, 11, 16, 22, 29, 37, 46, ... : 01010110

The three lines from the head are as follows:

· yz(alg=2, 3) :
· `algo=alg;`
· i=0, 1 :
· "en_yz();ui=algo;PM_algo(0);"
· "en_zx();ui=algo;PM_algo(-1);"
· "en_xy();ui=algo;− − −;"

· i=2, 3 :
· "en_zx();− − −;PM_algo(0);"
· "en_yz();− − −;PM_algo(-1);"
· "en_xy();− − −;− − −;"

· i=4, 5 :
· "en_zx();− − −;PM_algo(0);"
· "en_yz();− − −;PM_algo(-1);"
· "en_xy();− − −;− − −;"

· i=6, 7 :
· "en_yz();− − −;PM_algo(0);"
· "en_zx();− − −;PM_algo(-1);"
· "en_xy();− − −;− − −;"

· zx(alg=4, 5) :
· `algo=alg;`
· i=0, 1 :
· "en_zx();ui=algo;PM_algo(-1);"
· "en_yz();ui=algo;PM_algo(0);"
· "en_xy();ui=algo;− − −;"

· i=2, 3 :

· "en_yz();− − −;PM_algo(-1);"
· "en_zx();− − −;PM_algo(0);"
· "en_xy();− − −;− − −;"

· i=4, 5 :
· "en_yz();− − −;PM_algo(-1);"
· "en_zx();− − −;PM_algo(0);"
· "en_xy();− − −;− − −;"

· i=6, 7 :
· "en_zx();− − −;PM_algo(-1);"
· "en_yz();− − −;PM_algo(0);"
· "en_xy();− − −;− − −;"

5. Seed point

The basic seed point place is L0 in Figure 3. Assuming CPMAX to be 8, we fix seed points in the formation of 4+4 like the last Figure 5(b). Painting point moves to the other hyper cube while doing painting. Considering the symmetry, the moving is as follows:

· S2 : 1+1
· S3 : 1+1
· S4 : 1+1
· S5 : 1+1

· S6 : 4
· S7 : 4

· L1 : 4+4

The number is the number of painting points. These hyper cubes can also become seed point places. It is difficult that we fix the places by intuition, so we utilize painting. Doing a painting assuming L0 to be seed point place, if painting points reaches them, we memorize the coordinates. Because the cubes from S2 to S7 are SMALL, they can be discriminated precisely. The following is the amount of change in coordinate from first painting point for getting second painting point when 5 dimensions.

· S2 : $\Delta z + (-\Delta z)$
· S3 : $\Delta z + (-\Delta z)$
· S4 : $\Delta z + (-\Delta z)$
· S5 : $\Delta z + (-\Delta z)$

· S2 : Δu
· S3 : Δu
· S4 : Δu
· S5 : Δu

· S2 : Δv
· S3 : Δv
· S4 : Δv
· S5 : Δv

· S6 : $\Delta x + \Delta y$
· S7 : $\Delta x + \Delta y$

· S6 : Δu

· S7 : Δu

· S6 : Δv
· S7 : Δv

· L1 : $\Delta x + \Delta y$

· L1 : $\Delta z + (-\Delta z)$

· L1 : Δu

$\Delta z + (-\Delta z)$ represents joining of $\Delta z = \pm 1$. $\Delta x + \Delta y$ represents that $\Delta x = \pm 1$ and $\Delta y = \pm 1$ are joined like 4 points doing change in coordinate in image of square vortex.

6. Program
If Windows, make WX 0 and if Linux, make WX 1.

```
#define WX 1                           /* 0:Windows, 1:Xlib */
```

Periodic boundary condition is made off. If on, it becomes the last program.

```
#define PBC 0
```

DMS is dimension number. The maximum dimension number DMS_max is 10 and the minimum dimension number is 3.

```
#define DMS_max 10
#define DMS 10                         /* DMS <= DMS_max */
```

Number of painting points, CPMAX is chosen among 2, 4, 8.

```
#define CPMAX /*2*//*4*//*8*/8
```

If we use normal palette, make PAL 0 and if we use bit palette, make PAL 1.

```
#define PAL 0                          /* 0:normal palette, 1:bit palette */
```

cpwidth is fixed to 2.

```
#define cpwidth 2
```

The following decides the interval of two seed points.

```
#define delta_x 1
```

In 10 dimensions, coordinates are named like the following:

· $x,\ y,\ z,\ u,\ v,\ w,\ r,\ s,\ t,\ o$

We describe coordinates by the number of painting points, so assuming i to be painting point number, coordinates of searching point S are

· nx[i], ny[i], nz[i], nu[i], nv[i], nw[i], nr[i], ns[i], nt[i], no[i]
· i : 0, 1, ... , CPMAX−1

Besides, the following variables are used.

· Nx, Ny, Nz, Nu, Nv, Nw, Nr, Ns, Nt, No
· Nx=nx[i], Ny=ny[i], Nz=nz[i], Nu=nu[i], Nv=nv[i], Nw=nw[i], Nr=nr[i], Ns=ns[i], Nt=nt[i], No=no[i]

Coordinates of previous searching point S′ are

・nx_[i], ny_[i], nz_[i], nu_[i], nv_[i], nw_[i], nr_[i], ns_[i], nt_[i], no_[i]
・i : 0, 1, ... , CPMAX−1

・Nx_, Ny_, Nz_, Nu_, Nv_, Nw_, Nr_, Ns_, Nt_, No_
・Nx_=nx_[i], Ny_=ny_[i], Nz_=nz_[i], Nu_=nu_[i], Nv_=nv_[i], Nw_=nw_[i], Nr_=nr_[i], Ns_=ns_[i], Nt_=nt_[i], No_=no_[i]

7. Example of painting

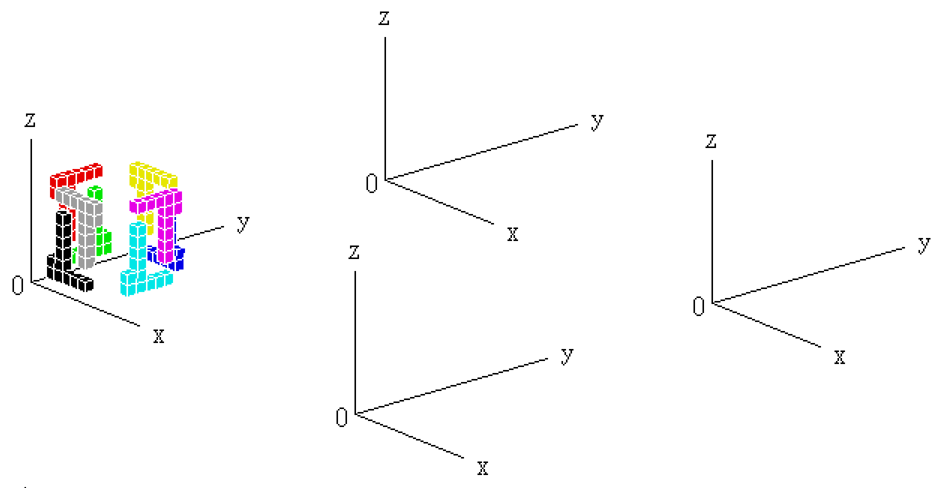

Figure 4

Figure 4 shows an example of painting in 5 dimensions. We choose parameters like the following:

```
#define Xlib 2
#define DMS 5                           /* DMS <= DMS_max */
#define CPMAX /*2*//*4*//*8*/8
#define delta_x 2
```

Assuming execution filename to be dr, the execution type is the following four ways:

・dr
・dr k
・dr 0
・dr 0 0

In dr k, dr 0 0, time is the argument of srand. Use dr or dr k. Esc or Pause is the progress key.
The painting number is from 978 to 988. The coordinates are like the following.

・left : $u = 6,\ v = 1$
・$w = 0,\ r = 0,\ s = 0,\ t = 0,\ o = 0$

The following is parameters for 10 dimensions.

```
#define Xlib 2
#define DMS 10                          /* DMS <= DMS_max */
#define CPMAX /*2*//*4*//*8*/8
#define delta_x 1
```

The coordinates are as follows:

・left : $u = 3,\ v = 0,\ w = 5,\ r = 0,\ s = 0,\ t = 0,\ o = 0$

In 3 dimensions, cube can be developed to plane. Therefore, if a painting is done seed points being placed in fours on the two facing surfaces or in threes around the two facing vertexes like the 6th Figure 7(c) or Figure 8, the same result as the 6th is got. In the former, we use the above mentioned painting algorithm. In the latter, we use the following algorithm:

```
· xy(alg=0, 1) :
· algo=alg;
· i=0, 3 :
· "en_xy();− − −;− − −;"
· "en_yz();− − −;− − −;"
· "en_zx();− − −;− − −;"

· i=1, 4 :
· "en_yz();− − −;− − −;"
· "en_zx();− − −;− − −;"
· "en_xy();− − −;− − −;"

· i=2, 5 :
· "en_zx();− − −;− − −;"
· "en_xy();− − −;− − −;"
· "en_yz();− − −;− − −;"

· yz(alg=2, 3) :
· algo=alg;
· i=0, 3 :
· "en_yz();− − −;− − −;"
· "en_zx();− − −;− − −;"
· "en_xy();− − −;− − −;"

· i=1, 4 :
· "en_zx();− − −;− − −;"
· "en_xy();− − −;− − −;"
· "en_yz();− − −;− − −;"

· i=2, 5 :
· "en_xy();− − −;− − −;"
· "en_yz();− − −;− − −;"
· "en_zx();− − −;− − −;"

· zx(alg=4, 5) :
· algo=alg;
· i=0, 3 :
· "en_zx();− − −;− − −;"
· "en_xy();− − −;− − −;"
· "en_yz();− − −;− − −;"

· i=1, 4 :
· "en_xy();− − −;− − −;"
· "en_yz();− − −;− − −;"
· "en_zx();− − −;− − −;"

· i=2, 5 :
· "en_yz();− − −;− − −;"
· "en_zx();− − −;− − −;"
· "en_xy();− − −;− − −;"
```

8. Raising of maximum dimension number

Beforehand, we modify 10d.4 on the last `/* here A* */`. Inside the program, there are comments which shows the part to modify on this time.

`/* here B1 */` : change of value
`/* here B2 */` : adding of #elif part
`/* here B3 */` : adding of variable in declaration statement
`/* here B4 */` : adding of dummy argument
`/* here B5 */` : adding of argument in prototype declarration
`/* here B6 */` : adding of if statement, if-else statement or else if statement
`/* here B7 */` : adding of expression statement or term in expression statement
`/* here B8 */` : adding of loop(upper part and lower part or only upper part)
`/* here B9 */` : adding of actual argument
`/* here B10 */` : adding or replacement of term in expression
`/* here B11 */` : adding of function

The procedure before or after modification is the same as the last.

If program is so completed that there is no room for improvement, we can make cubes LARGE. Make SMALLER 0.

```
#define SMALLER 0                    /* 0:normal, 1:smaller */
```

****

セルラーオートマトングラフィクス (12)

菊池盛雄

アブストラクト：
　三次元以上の空間内の超立方体の超面を組み合わせた図形において敷き詰めを行います。

1. 超立方体

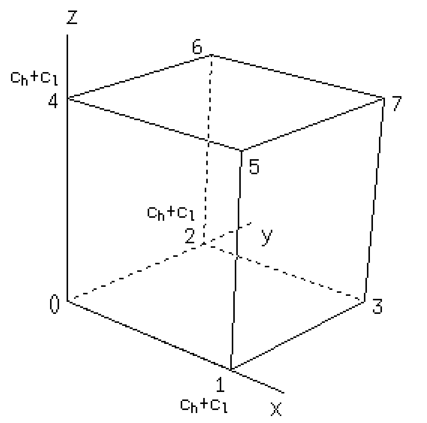

図 1

　図 1 は 3 次元空間内の超立方体、すなわち立方体を示します。今回は超立方体の内部ではなくその超面において敷き詰めを行います。
　図 1 の図形を構成する超面の名称を以下のように定義します。

・□ 0132, □ 4576：xy×2
・□ 0154, □ 2376：xz×2
・□ 0264, □ 1375：yz×2

各々、xy 平面、xz 平面、yz 平面に平行です。

4 次元では

・xyz×2
・xyu×2

・xzu×2
・yzu×2

5 次元では

・xyzu×2
・・xyzv×2

・xyuv×2
・xzuv×2
・yzuv×2

6 次元では

・xyzuv×2
・・xyzuw×2

・・xyzvw×2
・xyuvw×2
・xzuvw×2
・yzuvw×2

7 次元では

・xyzuvw×2
・・xyzuvr×2

・・xyzuwr×2
・・xyzvwr×2
・xyuvwr×2
・xzuvwr×2
・yzuvwr×2

8 次元では

・xyzuvwr×2
・・xyzuvws×2

・・xyzuvrs×2
・・xyzuwrs×2
・・xyzvwrs×2
・xyuvwrs×2
・xzuvwrs×2
・yzuvwrs×2

9 次元では

・xyzuvwrs×2
・・xyzuvwrt×2

・・xyzuvwst×2
・・xyzuvrst×2
・・xyzuwrst×2
・・xyzvwrst×2
・xyuvwrst×2
・xzuvwrst×2
・yzuvwrst×2

10 次元では

・xyzuvwrst×2
・・xyzuvwrso×2

・・xyzuvwrto×2
・・xyzuvwsto×2
・・xyzuvrsto×2
・・xyzuwrsto×2
・・xyzvwrsto×2
・xyuvwrsto×2
・xzuvwrsto×2
・yzuvwrsto×2

2. 構成

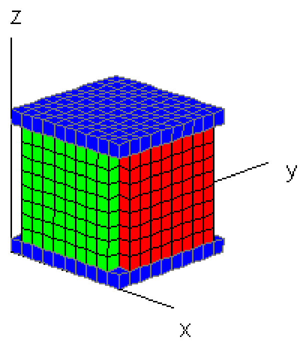

図 2

　次元数が大きくなると塗りつぶされる点の数を算出することが困難になります。そこで超立方体の超面を図 2 のようにカットします。この場合の辺の長さが 2 だけ短い超面 xz×2 および yz×2 を SMALL と称します。

・xy×2：LARGE(L)
・xz×2：SMALL(S)
・yz×2：SMALL(S)

LARGE、SMALL の辺の長さは以下の通りです。

・`cpwidth*(2*delta_x+1)`≡RESO
・LARGE：RESO
・SMALL：RESO−2

　次元数が大きくなると前で示したように超面の数が大きくなります。そこで用いる超面の数を 4 に制限します。・・の超面は用いず・の超面だけを用います。3 次元では

・xy×2
・xz×2
・yz×2

4 次元では

・xyz×2
・xyu×2
・xzu×2
・yzu×2

5 次元では

・xyzu×2
・xyuv×2
・xzuv×2
・yzuv×2

6 次元では

・xyzuv×2
・xyuvw×2
・xzuvw×2
・yzuvw×2

7 次元では

・xyzuvw×2
・xyuvwr×2
・xzuvwr×2
・yzuvwr×2

8 次元では

・xyzuvwr×2
・xyuvwrs×2
・xzuvwrs×2
・yzuvwrs×2

9 次元では

・xyzuvwrs×2
・xyuvwrst×2
・xzuvwrst×2
・yzuvwrst×2

10 次元では

・xyzuvwrst×2
・xyuvwrsto×2
・xzuvwrsto×2
・yzuvwrsto×2

このようにすれば塗りつぶされる点の数は以下の通りになります。

・`pow(RESO,2)*2+pow(RESO-2,3-1)*4`：3 次元
・`pow(RESO,2)*2+pow(RESO-2,DMS-1)*6`：4 次元以上

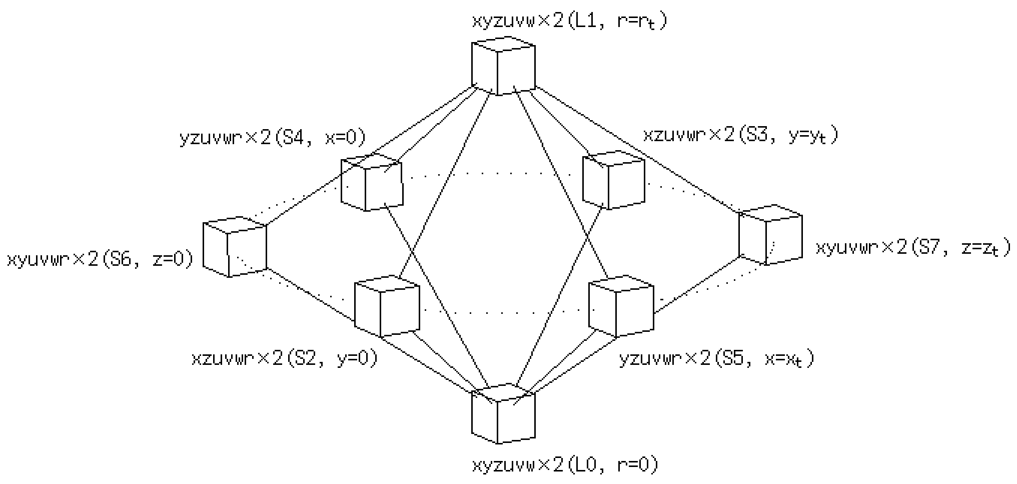

図 3

図 3 は 7 次元の場合の塗りつぶしの経路図です。図中の定数の値は以下の通りです。

・x_t：RESO−1
・y_t：RESO−1
・z_t：RESO−1
・u_t：RESO−1
・v_t：RESO−1
・w_t：RESO−1

・r_t：RESO−1

3. 塗りつぶしアルゴリズム
　たとえば平面アルゴリズム

・(c1,c3,c2,c4)≡en_xy

を前回において実際に用いる場合は、たとえば 4 次元では以下のようになります。

```
if((c1==ca || c3==ca || c2==ca || c4==ca)&&(Nz_==Nz && Nu_==Nu)){
jmax=en_xy();                          /* xy */
}
```

if() 内の第二項は次元数が大きくなればそれだけ大きくなります。7 次元では

・`Nz_==Nz && Nu_==Nu && Nv_==Nv && Nw_==Nw && Nr_==Nr`

今回はいくつかの式文を追加します。

```
else if((c3==ca || c5==ca || c4==ca || c6==ca)&& (Nz_==Nz && Nu_==Nu)){
jmax=en_yz();                          /* yz */
if(i==1) {if(algo%2==0) algo++;else algo--;}
}
```

CW、CCW は各々変数 algo=偶数、奇数に対応しています。追加された if 文は CW、CCW の切替を行います。i は塗点番号です。
　上の (else) if 文を以下のように表現します。

・"en_yz();−−−;PM_algo(1);"

また

```
int pm_algo(int algo)
{
if(algo%2==0) algo++;else algo--;

return algo;
}/** pm_algo **/
```

・`else if(i>=4 && (algo=pm_algo(algo))<0) ;`≡"i_geq_4_PM_algo"
・`alg=pm_algo(alg);`≡"i_eq_4_PM_algo"

とします。
　以下は、7 次元で、CPMAX が 8、シードポイントが図 3 の L0 または L1 において前回の図 5(b) のように 4+4 の形で置かれる場合の塗りつぶしアルゴリズムです。

・xy(alg=0, 1)：
・i=0, 1：
・`algo=alg;`
・"en_xy();−−−;−−−;"
・"en_yz();−−−;PM_algo(0+1);"
・"en_zx();−−−;PM_algo(0+1);"

・"en_xu();−−−;PM_algo(0+1);"
・"en_yu();−−−;PM_algo(0+1);"
・"en_zu();−−−;−−−;"

- "en_xv();− − −;PM_algo(0+1);"
- "en_yv();− − −;PM_algo(0+1);"
- "en_zv();− − −;− − −;"
- "en_uv();− − −;− − −;"

- "en_xw();− − −;PM_algo(0+1);"
- "en_yw();− − −;PM_algo(0+1);"
- "en_zw();− − −;− − −;"
- "en_uw();− − −;− − −;"
- "en_vw();− − −;− − −;"

- "en_xr();− − −;PM_algo(0+1);"
- "en_yr();− − −;PM_algo(0+1);"
- "en_zr();− − −;− − −;"
- "en_ur();− − −;− − −;"
- "en_vr();− − −;− − −;"
- "en_wr();− − −;− − −;"

- i=2, 3 :
- `algo=alg;`
- "en_xy();− − −;− − −;"
- "en_zx();− − −;PM_algo(0+2);"
- "en_yz();− − −;PM_algo(0+3);"

- "en_yu();− − −;PM_algo(0+2);"
- "en_xu();− − −;PM_algo(0+3);"
- "en_zu();− − −;− − −;"

- "en_yv();− − −;PM_algo(0+2);"
- "en_xv();− − −;PM_algo(0+3);"
- "en_zv();− − −;− − −;"
- "en_uv();− − −;− − −;"

- "en_yw();− − −;PM_algo(0+2);"
- "en_xw();− − −;PM_algo(0+3);"
- "en_zw();− − −;− − −;"
- "en_uw();− − −;− − −;"
- "en_vw();− − −;− − −;"

- "en_yr();− − −;PM_algo(0+2);"
- "en_xr();− − −;PM_algo(0+3);"
- "en_zr();− − −;− − −;"
- "en_ur();− − −;− − −;"
- "en_vr();− − −;− − −;"
- "en_wr();− − −;− − −;"

- "i_eq_4_PM_algo"

- i=4, 5 :
- `algo=alg;`
- "en_xy();− − −;− − −;"
- "en_zx();− − −;PM_algo(4+0);"
- "en_yz();− − −;PM_algo(4+0);"

- "en_yu();− − −;PM_algo(4+0);"
- "en_xu();− − −;PM_algo(4+0);"
- "en_zu();− − −;− − −;"

- "en_yv();− − −;PM_algo(4+0);"
- "en_xv();− − −;PM_algo(4+0);"
- "en_zv();− − −;− − −;"
- "i_geq_4_PM_algo"
- "en_uv();− − −;− − −;"

- "i_geq_4_PM_algo"
- "en_yw();− − −;PM_algo(4+0);"
- "en_xw();− − −;PM_algo(4+0);"
- "en_zw();− − −;− − −;"
- "i_geq_4_PM_algo"
- "en_uw();− − −;− − −;"
- "en_vw();− − −;− − −;"

- "i_geq_4_PM_algo"
- "en_yr();− − −;PM_algo(4+0);"
- "en_xr();− − −;PM_algo(4+0);"
- "en_zr();− − −;− − −;"
- "i_geq_4_PM_algo"
- "en_ur();− − −;− − −;"
- "en_vr();− − −;− − −;"
- "en_wr();− − −;− − −;"

- i=6, 7 :
- `algo=alg;`
- "en_xy();− − −;− − −;"
- "en_yz();− − −;PM_algo(4+2);"
- "en_zx();− − −;PM_algo(4+3);"

- "en_xu();− − −;PM_algo(4+2);"
- "en_yu();− − −;PM_algo(4+3);"
- "en_zu();− − −;− − −;"

- "en_xv();− − −;PM_algo(4+2);"
- "en_yv();− − −;PM_algo(4+3);"
- "en_zv();− − −;− − −;"
- "i_geq_4_PM_algo"
- "en_uv();− − −;− − −;"

- "i_geq_4_PM_algo"
- "en_xw();− − −;PM_algo(4+2);"
- "en_yw();− − −;PM_algo(4+3);"
- "en_zw();− − −;− − −;"
- "i_geq_4_PM_algo"
- "en_uw();− − −;− − −;"
- "en_vw();− − −;− − −;"

- "i_geq_4_PM_algo"
- "en_xr();− − −;PM_algo(4+2);"
- "en_yr();− − −;PM_algo(4+3);"

・"en_zr();−−−;−−−;"
・"i_geq_4_PM_algo"
・"en_ur();−−−;−−−;"
・"en_vr();−−−;−−−;"
・"en_wr();−−−;−−−;"

en_yz() を先頭にすれば塗りつぶしアルゴリズムの先頭からの 3 行は以下のようになります。

・yz(alg=2, 3)：
・i=0, 1：
・`algo=alg;`
・"en_yz();−−−;PM_algo(0+1);"
・"en_zx();−−−;PM_algo(0+1);"
・"en_xy();−−−;−−−;"

・i=2, 3：
・`algo=alg;`
・"en_zx();−−−;PM_algo(0+2);"
・"en_yz();−−−;PM_algo(0+3);"
・"en_xy();−−−;−−−;"

・"i_eq_4_PM_algo"

・i=4, 5：
・`algo=alg;`
・"en_zx();−−−;PM_algo(4+0);"
・"en_yz();−−−;PM_algo(4+0);"
・"en_xy();−−−;−−−;"

・i=6, 7：
・`algo=alg;`
・"en_yz();−−−;PM_algo(4+2);"
・"en_zx();−−−;PM_algo(4+3);"
・"en_xy();−−−;−−−;"

en_zx() を先頭にすれば塗りつぶしアルゴリズムの先頭からの 3 行は以下のようになります。

・zx(alg=4, 5)：
・i=0, 1：
・`algo=alg;`
・"en_zx();−−−;PM_algo(0+1);"
・"en_yz();−−−;PM_algo(0+1);"
・"en_xy();−−−;−−−;"

・i=2, 3：
・`algo=alg;`
・"en_yz();−−−;PM_algo(0+3);"
・"en_zx();−−−;PM_algo(0+2);"
・"en_xy();−−−;−−−;"

・"i_eq_4_PM_algo"

・i=4, 5：
・`algo=alg;`

・"en_yz();−−−;PM_algo(4+0);"
・"en_zx();−−−;PM_algo(4+0);"
・"en_xy();−−−;−−−;"

・i=6, 7：
・`algo=alg;`
・"en_zx();−−−;PM_algo(4+3);"
・"en_yz();−−−;PM_algo(4+2);"
・"en_xy();−−−;−−−;"

以上は 3 組、xy、yz、zx を併用する場合です。組合わせは自由です。たとえば

・zx：alg=0, 1
・yz：alg=2, 3

4. CW、CCW の切替
CW、CCW の切替は PM_algo 以外の手段によっても行えます。

```
else if((c3==ca || c5==ca || c4==ca || c6==ca)&& (Nz_==Nz && Nu_==Nu)){
jmax=en_yz();                         /* yz */
/*if(i==1) {if(algo%2==0) algo++;else algo--;}*/
if(i==0) ui=algo;
algo=PM_algo_en(1,algo);
}
```

上の (else) if 文を以下のように表現します。

・"en_yz();ui=algo;PM_algo_en(1);"

塗りつぶしアルゴリズムは以下のようになります。

・xy(alg=0, 1)：
・`algo=alg;`
・i=0, 1：
・"en_xy();ui=algo;−−−;"
・"en_yz();ui=algo;PM_algo_en(1);"
・"en_zx();ui=algo;PM_algo_en(2);"

・"en_xu();ui=algo;PM_algo_en(3);"
・"en_yu();ui=algo;PM_algo_en(4);"
・"en_zu();ui=algo;−−−;"

・"en_xv();ui=algo;PM_algo_en(6);"
・"en_yv();ui=algo;PM_algo_en(7);"
・"en_zv();ui=algo;−−−;"
・"en_uv();ui=algo;−−−;"

・"en_xw();ui=algo;PM_algo_en(10);"
・"en_yw();ui=algo;PM_algo_en(11);"
・"en_zw();ui=algo;−−−;"
・"en_uw();ui=algo;−−−;"
・"en_vw();ui=algo;−−−;"

・"en_xr();ui=algo;PM_algo_en(15);"
・"en_yr();ui=algo;PM_algo_en(16);"

- "en_zr();ui=algo;− − −;"
- "en_ur();ui=algo;− − −;"
- "en_vr();ui=algo;− − −;"
- "en_wr();ui=algo;− − −;"

- i=2, 3 :
- "en_xy();− − −;− − −;"
- "en_zx();− − −;PM_algo_en(1);"
- "en_yz();− − −;PM_algo_en(2);"

- "en_yu();− − −;PM_algo_en(3);"
- "en_xu();− − −;PM_algo_en(4);"
- "en_zu();− − −;− − −;"

- "en_yv();− − −;PM_algo_en(6);"
- "en_xv();− − −;PM_algo_en(7);"
- "en_zv();− − −;− − −;"
- "en_uv();− − −;− − −;"

- "en_yw();− − −;PM_algo_en(10);"
- "en_xw();− − −;PM_algo_en(11);"
- "en_zw();− − −;− − −;"
- "en_uw();− − −;− − −;"
- "en_vw();− − −;− − −;"

- "en_yr();− − −;PM_algo_en(15);"
- "en_xr();− − −;PM_algo_en(16);"
- "en_zr();− − −;− − −;"
- "en_ur();− − −;− − −;"
- "en_vr();− − −;− − −;"
- "en_wr();− − −;− − −;"

- i=4, 5 :
- "en_xy();− − −;− − −;"
- "en_zx();− − −;PM_algo_en(1);"
- "en_yz();− − −;PM_algo_en(2);"

- "en_yu();− − −;PM_algo_en(3);"
- "en_xu();− − −;PM_algo_en(4);"
- "en_zu();− − −;− − −;"

- "en_yv();− − −;PM_algo_en(6);"
- "en_xv();− − −;PM_algo_en(7);"
- "en_zv();− − −;− − −;"
- "en_uv();− − −;− − −;"

- "en_yw();− − −;PM_algo_en(10);"
- "en_xw();− − −;PM_algo_en(11);"
- "en_zw();− − −;− − −;"
- "en_uw();− − −;− − −;"
- "en_vw();− − −;− − −;"

- "en_yr();− − −;PM_algo_en(15);"
- "en_xr();− − −;PM_algo_en(16);"

・"en_zr();−−−;−−−;"
・"en_ur();−−−;−−−;"
・"en_vr();−−−;−−−;"
・"en_wr();−−−;−−−;"

・i=6, 7：
・"en_xy();−−−;−−−;"
・"en_yz();−−−;PM_algo_en(1);"
・"en_zx();−−−;PM_algo_en(2);"

・"en_xu();−−−;PM_algo_en(3);"
・"en_yu();−−−;PM_algo_en(4);"
・"en_zu();−−−;−−−;"

・"en_xv();−−−;PM_algo_en(6);"
・"en_yv();−−−;PM_algo_en(7);"
・"en_zv();−−−;−−−;"
・"en_uv();−−−;−−−;"

・"en_xw();−−−;PM_algo_en(10);"
・"en_yw();−−−;PM_algo_en(11);"
・"en_zw();−−−;−−−;"
・"en_uw();−−−;−−−;"
・"en_vw();−−−;−−−;"

・"en_xr();−−−;PM_algo_en(15);"
・"en_yr();−−−;PM_algo_en(16);"
・"en_zr();−−−;−−−;"
・"en_ur();−−−;−−−;"
・"en_vr();−−−;−−−;"
・"en_wr();−−−;−−−;"

以下に PM_algo_en による CW、CCW の切替の処理手順を示します。

(1) 塗点番号 i=0 において algo を ui に記憶します。
(2) 塗点番号 i>0 においては PM_algo_en においてその引数にしたがった処理を行います。

・引数 1, 3, 6, 10, 15, 21, 28, 36, 45, ...：01100101
・引数 2, 4, 7, 11, 16, 22, 29, 37, 46, ...：01010110

塗点番号と処理番号列 01100101、01010110 との対応は以下のようになっています。

・i=0：0
・i=1：1
・i=2：1
・i=3：0
・i=4：0
・i=5：1
・i=6：0
・i=7：1

・i=0：0
・i=1：1
・i=2：0

・i=3：1
・i=4：0
・i=5：1
・i=6：1
・i=7：0

処理番号 0 は algo を以下のように処理します。

・ui%2 が 0 なら algo%2 が 0 となるように algo を処理します。
・ui%2 が 1 なら algo%2 が 1 となるように algo を処理します。

処理番号 1 は algo を以下のように処理します。

・ui%2 が 0 なら algo%2 が 1 となるように algo を処理します。
・ui%2 が 1 なら algo%2 が 0 となるように algo を処理します。

ただし、Δalgo=±1 は algo/2 が変わらないように選ばなければなりません。
　en_yz()、en_zx() が先頭である場合は引数群に 0、−1 を加えます。

・引数　0, 1, 3, 6, 10, 15, 21, 28, 36, 45, ...：01100101
・引数 −1, 2, 4, 7, 11, 16, 22, 29, 37, 46, ...：01010110

塗りつぶしアルゴリズムの先頭からの 3 行は各々以下のようになります。

・yz(alg=2, 3)：
・`algo=alg;`
・i=0, 1：
・"en_yz();ui=algo;PM_algo(0);"
・"en_zx();ui=algo;PM_algo(-1);"
・"en_xy();ui=algo;− − −;"

・i=2, 3：
・"en_zx();− − −;PM_algo(0);"
・"en_yz();− − −;PM_algo(-1);"
・"en_xy();− − −;− − −;"

・i=4, 5：
・"en_zx();− − −;PM_algo(0);"
・"en_yz();− − −;PM_algo(-1);"
・"en_xy();− − −;− − −;"

・i=6, 7：
・"en_yz();− − −;PM_algo(0);"
・"en_zx();− − −;PM_algo(-1);"
・"en_xy();− − −;− − −;"

・zx(alg=4, 5)：
・`algo=alg;`
・i=0, 1：
・"en_zx();ui=algo;PM_algo(-1);"
・"en_yz();ui=algo;PM_algo(0);"
・"en_xy();ui=algo;− − −;"

・i=2, 3：

・"en_yz();−−−;PM_algo(-1);"
・"en_zx();−−−;PM_algo(0);"
・"en_xy();−−−;−−−;"

・i=4, 5：
・"en_yz();−−−;PM_algo(-1);"
・"en_zx();−−−;PM_algo(0);"
・"en_xy();−−−;−−−;"

・i=6, 7：
・"en_zx();−−−;PM_algo(-1);"
・"en_yz();−−−;PM_algo(0);"
・"en_xy();−−−;−−−;"

5. シードポイント

基本的なシードポイントの位置は図 3 の L0 です。CPMAX を 8 とし、シードポイントを前回の図 5(b) のように 4+4 の形で置きます。塗点は塗りつぶしを行いながら他の超立方体に移動します。対称性を考慮すれば以下のような移動が考えられます。

・S2：1+1
・S3：1+1
・S4：1+1
・S5：1+1

・S6：4
・S7：4

・L1：4+4

数字は塗点の数です。これらの超立方体もシードポイントの位置とすることができます。直観ではわかりにくいので塗りつぶしを利用します。L0 をシードポイントの位置として塗りつぶしを行い、塗点が各々に到達したらそれらの座標を記憶します。S2 から S7 は SMALL なので各々は正確に識別できます。第二塗点を得る第一塗点からの座標変化量を 5 次元の場合について以下に示します。

・S2：$\Delta z + (-\Delta z)$
・S3：$\Delta z + (-\Delta z)$
・S4：$\Delta z + (-\Delta z)$
・S5：$\Delta z + (-\Delta z)$

・S2：Δu
・S3：Δu
・S4：Δu
・S5：Δu

・S2：Δv
・S3：Δv
・S4：Δv
・S5：Δv

・S6：$\Delta x + \Delta y$
・S7：$\Delta x + \Delta y$

・S6：Δu
・S7：Δu

・S6：Δv
・S7：Δv

・L1：$\Delta x + \Delta y$

・L1：$\Delta z + (-\Delta z)$

・L1：Δu

$\Delta z + (-\Delta z)$ は $\Delta z = \pm 1$ を組み合わせることを表します。$\Delta x + \Delta y$ は、4 点が四角い渦のイメージで座標変化を行うように $\Delta x = \pm 1$ と $\Delta y = \pm 1$ を組み合わせることを表します。

6. プログラム
　Windows では 0、 Linux では 1 として下さい。

```
#define WX 1                          /* 0:Windows, 1:Xlib */
```

　周期的境界条件はオフにします。オンにすれば前回のプログラムになります。

```
#define PBC 0
```

　DMS が次元です。最大次元 DMS_max は 10 次元であり、最小次元は 3 次元です。

```
#define DMS_max 10
#define DMS 10                        /* DMS <= DMS_max */
```

　塗点の数 CPMAX は 2、4、8 から選択します。

```
#define CPMAX /*2*//*4*//*8*/8
```

　normal palette は 0、1:bit palette は 1 です。

```
#define PAL 0                         /* 0:normal palette, 1:bit palette */
```

　cpwidth は 2 に固定します。

```
#define cpwidth 2
```

　シードポイントの間隔を決めます。

```
#define delta_x 1
```

　10 次元では座標は以下のように命名します。

・x, y, z, u, v, w, r, s, t, o

塗点の数だけ座標を記述するので、i を塗点番号として、探索点 S の座標は

・nx[i], ny[i], nz[i], nu[i], nv[i], nw[i], nr[i], ns[i], nt[i], no[i]
・i：0, 1, ... , CPMAX−1

また、以下の変数も使用します。

・Nx, Ny, Nz, Nu, Nv, Nw, Nr, Ns, Nt, No
・Nx=nx[i], Ny=ny[i], Nz=nz[i], Nu=nu[i], Nv=nv[i], Nw=nw[i], Nr=nr[i], Ns=ns[i], Nt=nt[i], No=no[i]

一つ前の探索点 S′ は

・nx_[i], ny_[i], nz_[i], nu_[i], nv_[i], nw_[i], nr_[i], ns_[i], nt_[i], no_[i]

・i : 0, 1, ... , CPMAX−1

・Nx_, Ny_, Nz_, Nu_, Nv_, Nw_, Nr_, Ns_, Nt_, No_
・Nx_=nx_[i], Ny_=ny_[i], Nz_=nz_[i], Nu_=nu_[i], Nv_=nv_[i], Nw_=nw_[i], Nr_=nr_[i], Ns_=ns_[i], Nt_=nt_[i], No_=no_[i]

7. 描画例

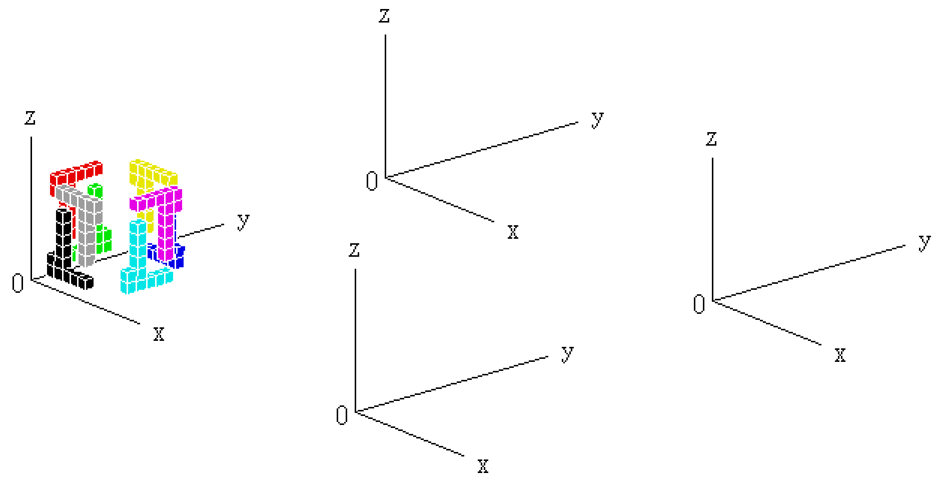

図 4

5 次元の場合の描画例を図 4 に示します。パラメーターは以下のようにします。

```
#define Xlib 2
#define DMS 5                       /* DMS <= DMS_max */
#define CPMAX /*2*//*4*//*8*/8
#define delta_x 2
```

実行ファイル名を dr として実行形式は以下の 4 通りあります。

・dr
・dr k
・dr 0
・dr 0 0

dr k と dr 0 0 では時刻が srand の引数です。dr または dr k で実行して下さい。Esc または Pause で進行します。
塗番号は 978 から 988 です。座標は以下のようになっています。

・左図：$u = 6,\ v = 1$
・$w = 0,\ r = 0,\ s = 0,\ t = 0,\ o = 0$

10 次元の場合は以下のようにします。

```
#define Xlib 2
#define DMS 10                      /* DMS <= DMS_max */
#define CPMAX /*2*//*4*//*8*/8
#define delta_x 1
```

座標は

・左図：$u = 3,\ v = 0,\ w = 5,\ r = 0,\ s = 0,\ t = 0,\ o = 0$

となります。
3 次元では立方体を平面に展開することができます。したがって、第六回の図 7(c) または図 8 のように対向する 2 面に 4 個ずつまたは対向する 2 頂点の周囲に 3 個ずつシードポイントを置いて塗りつぶしを行うと、第六回の結果と同様な結果が得られます。前者では前に述べた塗りつぶしアルゴリズムを用います。後者では以下のような塗りつぶしアルゴリズムを用います。

・xy(alg=0, 1)：
・`algo=alg;`
・i=0, 3：
・"en_xy();−−−;−−−;"
・"en_yz();−−−;−−−;"
・"en_zx();−−−;−−−;"

・i=1, 4：
・"en_yz();−−−;−−−;"
・"en_zx();−−−;−−−;"
・"en_xy();−−−;−−−;"

・i=2, 5：
・"en_zx();−−−;−−−;"
・"en_xy();−−−;−−−;"
・"en_yz();−−−;−−−;"

・yz(alg=2, 3)：
・`algo=alg;`
・i=0, 3：
・"en_yz();−−−;−−−;"
・"en_zx();−−−;−−−;"
・"en_xy();−−−;−−−;"

・i=1, 4：
・"en_zx();−−−;−−−;"
・"en_xy();−−−;−−−;"
・"en_yz();−−−;−−−;"

・i=2, 5：
・"en_xy();−−−;−−−;"
・"en_yz();−−−;−−−;"
・"en_zx();−−−;−−−;"

・zx(alg=4, 5)：
・`algo=alg;`
・i=0, 3：
・"en_zx();−−−;−−−;"
・"en_xy();−−−;−−−;"
・"en_yz();−−−;−−−;"

・i=1, 4：
・"en_xy();−−−;−−−;"
・"en_yz();−−−;−−−;"
・"en_zx();−−−;−−−;"

・i=2, 5：
・"en_yz();−−−;−−−;"
・"en_zx();−−−;−−−;"
・"en_xy();−−−;−−−;"

8. 高次元化

10d.4 を前回の`/* here A* */`に関して修正しておきます。プログラム中には今回に関して修正すべき箇所を示すコメントがあ

ります。

`/* here B1 */`：値の変更
`/* here B2 */`：#elif パートのアド
`/* here B3 */`：宣言文における変数のアド
`/* here B4 */`：仮引数のアド
`/* here B5 */`：プロトタイプ宣言における引数のアド
`/* here B6 */`：if 文、if-else 文、または else if 文のアド
`/* here B7 */`：式文における項または式文のアド
`/* here B8 */`：ループのアド (上と下、または上だけ)
`/* here B9 */`：実引数のアド
`/* here B10 */`：式における項のアドまたはリプレイス
`/* here B11 */`：関数のアド

修正前、修正後の手順は前回と同じです。
　プログラムが改善の余地がないほど十分完成されたら超立方体を LARGE にしてもかまいません。SMALLER を 0 として下さい。

```
#define SMALLER 0                    /* 0:normal, 1:smaller */
```

****

List 1:cag_12.c

```
/* 10d.4((SP4/p)*256)                                               */
/* 2020 Morio Kikuchi                                               */
/* gcc -w dr.c -o dr -I/usr/include/X11 -L/usr/lib -lX11 -lm       */
/* tested in plamo 6.2, slackware 14.2 (both are 32-bit)           */

#define WX /*0*//*1*/1              /* 0:Windows, 1:Xlib */
#define Xlib 2

#if !WX
#include <windows.h>
#else
#if Xlib
#include <X11/Xlib.h>
#include <X11/Xutil.h>
#include <X11/Xlocale.h>
#include <X11/cursorfont.h>
#include <X11/keysym.h>
#endif
#endif
#include <stdio.h>
#include <stdlib.h>
#include <time.h>
#include <math.h>
#include <signal.h>

#define dbl double
#if !WX
#define GKS GetKeyState
#define GKS_ GetKeyState
#define FOPEN fopen
#else
#if Xlib
#define TRANS (65535./255)
#define XDSDY (asct+1)
#endif
#define FOPEN fopen64
#endif

#define PBC 0
#define DMS_limit 100
/* here A1 *//* make DMS_max 11 */
#define DMS_max 10
/* here A1 *//* make DMS 11 */
#define DMS 5                       /* DMS <= DMS_max */

#if PBC
#if   DMS==3
#define CPMAX 8/*125*/
#elif DMS==4
#define CPMAX 16
```

```
#elif DMS==5
#define CPMAX 32
#elif DMS==6
#define CPMAX 64
#elif DMS==7
#define CPMAX 128
#elif DMS==8
#define CPMAX 256
#elif DMS==9
#define CPMAX 512
#elif DMS==10
#define CPMAX 1024
/* here A2 *//* add*/
/*#elif DMS==11
#define CPMAX 2048
*/
#endif

#if DMS==3 && CPMAX>64
#define VGACOLORS CPMAX
#else
#define VGACOLORS 512                /* > 64 */
#endif

#else
#define CPMAX /*2*//*4*//*8*/8
#define VGACOLORS 64
#endif

#define M_or_D 0                     /* 0:memory, 1:disk */
#define PAL 0                        /* 0:normal palette, 1:bit palette */

#if PAL==0
#define fcolor -1
#define wall -2
#else
#define fcolor 0
#define wall 1
#endif

#if PAL || CPMAX<=128
#define INT char
#else
#define INT short/*long*/
#endif

#define cpwidth 2/*5*/

#if DMS==5
#define delta_x 2
#else
#define delta_x 1/*5*/
#endif

#define SMALLER 1                    /* 0:normal, 1:smaller */
```

```
#define _1st__ /*0*//*1*/0
#define _2nd__ /*0*//*1*//*2*/0
#define RESO (cpwidth*(2*delta_x+1))
#define PM_algo {if(algo%2==0) algo++;else algo--;}
#define CHK_H 1                     /* check_hole() */

#define XRESO 1280
#define YRESO 768
#define Zx (XRESO*1)
#define Zy (YRESO*1)
#define Zxdiv Zx/4
#define Zydiv Zy/2
#define CNT 1
#if PBC
#define CUBE -1
#else
#define CUBE 2/*1*/                       /* 0:xyU, 1:xyV, 2:xzU, yzU, 3:xzV, yzV */
#endif
#if DMS==10
#define _V o
#define _VT ot
#else
#define _V v
#define _VT vt
#endif
#if Xlib==1
#define PS 12
#else
#define PS 8
#endif

#define ICEIL(a,b) (((a)+((b)-1))/(b))
#define UdX 10
#define UdY 20
#define TORAD (acos(-1)/180)
#define TODEG (180/acos(-1))

#define ASIZE (256+1)

char Rectflag;
int idx,idx_Rect;
char work_Rect[Zx][Zy];

int Zflag,dO[2];
long rho,dscr;
dbl th,ph,mx[4][3];
dbl DET,xE,yE,zE;
dbl Zbuf[Zx][Zy];

char refill,pauseflag,fieldflag,GRPH;
char charcode,charflag,Trianflag,PMflag,pixelflag;
/* here A3 *//* add P, P_ */
int X,Y,Z,U,V,W,R,S,T,O,X_,Y_,Z_,U_,V_,W_,R_,S_,T_,O_,xg,yg,zg,ui;
/* here A3 *//* add c21,c22 */
INT ca,c1,c3,c2,c4,c5,c6,c7,c8,c9,c10,c11,c12,c13,c14,c15,c16,c17,c18,c19,c20;
INT acolor[CPMAX];
int nx[CPMAX],ny[CPMAX],nz[CPMAX],nu[CPMAX],nx_[CPMAX],ny_[CPMAX],nz_[CPMAX],nu_[CPMAX];
int nv[CPMAX],nw[CPMAX],nr[CPMAX],ns[CPMAX],nv_[CPMAX],nw_[CPMAX],nr_[CPMAX],ns_[CPMAX];
/* here A3 *//* add np[CPMAX], np_[CPMAX] */
int nt[CPMAX],no[CPMAX],nt_[CPMAX],no_[CPMAX];
/* here A3 *//* add Np, Np_ */
int Nx,Ny,Nz,Nu,Nv,Nw,Nr,Ns,Nt,No,Nx_,Ny_,Nz_,Nu_,Nv_,Nw_,Nr_,Ns_,Nt_,No_;
/* here A3 *//* change 1+20 into 1+22 */
int x[1+20],y[1+20],z[1+20],u[1+20],x_[1+20],y_[1+20],z_[1+20],u_[1+20];
int v[1+20],w[1+20],r[1+20],s[1+20],v_[1+20],w_[1+20],r_[1+20],s_[1+20];
/* here A3 *//* add p[1+22], p_[1+22] */
int t[1+20],o[1+20],t_[1+20],o_[1+20];
int enX[4],enY[4],enZ[4],enU[4],enX_[4],enY_[4],enZ_[4],enU_[4];
int enV[4],enW[4],enR[4],enS[4],enT[4],enV_[4],enW_[4],enR_[4],enS_[4],enT_[4];
/* here A3 *//* add enP[4], enP_[4] */
int enO[4],enO_[4];
int ig,putperiod,sn_,jmax;
int xt,yt,zt,ut,nax[CPMAX*2],nay[CPMAX*2],naz[CPMAX*2],nau[CPMAX*2];
int vt,wt,rt,st,tt,nav[CPMAX*2],naw[CPMAX*2],nar[CPMAX*2],nas[CPMAX*2],nat[CPMAX*2];
/* here A3 *//* add pt, nap[CPMAX*2] */
int ot,nao[CPMAX*2];
long long pcount[CPMAX],qcount[CPMAX],cnt,sum;
dbl tmp0,tmp1,tmp2,tmp3;

/*unsigned char*/INT *pixel;

char function,usflag;
unsigned char yorn;
int WB;
long long jpow[DMS_max];
FILE *fp[CPMAX],*fp_pixel;

#if WX
#if Xlib
int argc_,asct;
char **argv_,appliname[]="CAG",fs1[ASIZE];

char *fs2[5]={
              "-*-*-medium-r-normal--16-*",  /* fn_set_0 (SFS) */
              "-*-*-medium-r-normal--20-*",  /* fn_set_1 (SFMM) */
              "-*-*-medium-r-normal--20-*",  /* fn_set_2 (SFM) */
              "-*-*-medium-r-normal--24-*",  /* fn_set_3 (SFL) */

              "none"
                                                   };  /* fontnames */
#endif
#endif

typedef struct {
char p;dbl x,y,z;} swork;
swork work[Zx][Zy];
typedef struct {
int x,y;} sstack;
sstack stack[Zx*1],stack_Rect[Zx*1];
/* here A3 */                                            /* add pp */
```

```
typedef struct {char/*short*/ xx,yy,zz,uu,vv,ww,rr,ss,tt,oo,
/* here A3 */                                      /* add pp_ */
                     xx_,yy_,zz_,uu_,vv_,ww_,rr_,ss_,tt_,oo_;} sss;
sss ss;
#if !WX
typedef struct {
unsigned char red,green,blue;} srgb;
srgb irgb[VGACOLORS];
typedef struct {
unsigned long back_;int back,fore;} bf;
bf bfset[]={{WHITENESS,15,0},{BLACKNESS,0,15}};
#else
#if Xlib
XColor irgb[VGACOLORS],c;
typedef struct {
int back,fore;} bf;
bf bfset[]={{15,0},{0,15}};
#endif
#endif

#if !WX
HINSTANCE hinstance;
HWND hwnd;
HDC hdcdisplay,hdctmp1;
HBITMAP hbitmap1;
HANDLE hfile[CPMAX];
#else
#if Xlib
Display *d;
int screen,depth;
Colormap cmap;
Window rw,ww;
XSizeHints sh;
GC gcdisplay;
Pixmap pmap1;
XFontSet font_fs;
XFontStruct **info;
Cursor cursor;
XEvent event;
KeySym keysym,sym;
Atom atm1,atm2;
Visual *vis;
XImage *image;

unsigned long mask;
char **flist,**mlist,*def;
int mcount,fontnum;
XIM ime;
XIMStyle style;
XIC ic;
Status stts;
#endif
#endif

void closegraph_(void),initpalette(void),BitBlt_full(void),terminator(int),
     cleardevice_(char,int,int,int,int),rectangle_(char,int,int,int,int,int,int),
     delay_(long),beep(long),kbhit_(void),restore_3(void),check_rcount(void),
     use_subroop(void),keydowns_f2(void),bitblt(char,int,int,int,int,int,int),
     arrayreset(void),field(void),
     axes_eye(char,char,char,dbl,dbl,dbl,char*,char*,char*),
     line_eye_(dbl,dbl,dbl,dbl,dbl,dbl,int),mallocs(void),frees(void),getsum(void),
     initeye(void),set_O(int,int),clearZbuf(void),set_Z(char),
     projection(dbl,dbl,dbl,int *,int *),puts_(int,int,int,char *);
unsigned char subroop(void);
int initgraph_(void),fourfloor_fiveceil(dbl),random_(int),cag_r(long),
/* here B5 */                                 /* add int */
    getplane(int,int,int,int,int,int,int,int,int,int,int);
/* here A5 */                             /* add int */
    putpixel(int,int,int,int,int,int,int,int,int,int,INT);
/* here A5 */                              /* add int */
INT pfunc(int,int,int,int,int,int,int,int,int,int,int,INT),
/* here A5 */                             /* add int */
    getpixel(int,int,int,int,int,int,int,int,int,int),
/* here A5 */                           /* add int */
    rpixel(int,int,int,int,int,int,int,int,int,int);

#if !WX
COLORREF PALETTE(int color);
LRESULT CALLBACK wndproc_by_kbhit_(HWND,UINT,WPARAM,LPARAM);
int wndproc_filer(HWND,UINT,WPARAM,LPARAM);
#else
#if Xlib
int wndproc_filer(void);
XIMStyle InputStyle(XIM);
XIC InputContext(XIM,XIMStyle,XFontSet,Window);
#endif
#endif



int main(int argc,unsigned char **argv)
{
char str[32];
int i,xs,ys,dlt,xo,yo,cubes;
long mytime,oldtime,nowtime,dtime;
long long vall;
dbl x,y,z,val;

if(DMS<3 || DMS>DMS_max) return 1;
if(_1st__<0 || _1st__>1) return 1;
if(_2nd__<0 || _2nd__>2) return 1;
if(_1st__==1 && _2nd__==0 && DMS>3){
printf(" _1st__==1 && _2nd__==0 && DMS>3\n");
return 1;
}




fieldflag=-1;

if(!Xlib){
```

```
GRPH=0;
if(argc>1 && strcmp(argv[1],"0")==0) {if(argc==2) argc=1;else argc=2;}
else     {if(argc==1) argc=1;else argc=2;}  /* dr (s) */
}
else if(argc>1 && strcmp(argv[1],"0")==0){
GRPH=0;
if(argc==2) argc=1;else argc=2;
}
else{
GRPH=1;
}
WB=0;
refill=1;
sn_=0;

if(initgraph_()==1) return 1;

#if !WX || Xlib
if(GRPH){
rho=5000*1;if(Xlib==1) th=90-atan(1/sqrt(2))*TODEG;else th=70;ph=-40;
dscr=5000;

initeye();
set_Z(1);
if(Xlib==1) dlt=150;else dlt=0;
if(DMS==3) {xo=80+250;yo=205+150;}
else       {xo=80+dlt;yo=205+dlt;}
set_O(0+xo,0+yo);

     if(Xlib==1 && GRPH) sn_=-1;
else if(Xlib==2 && GRPH/* && cnt==CNT-1*/) sn_=1;
}

cleardevice_(1,0,0,XRESO,YRESO);
BitBlt_full();
#endif

printf(" DMS:%dd/%dd PBC:%d CPMAX:%d delta_x:%d RESO:%d\n",DMS,DMS_max,PBC,CPMAX,delta_x,RESO);

if(argc>1) {time(&mytime);srand((unsigned int)mytime);}
else
srand(/*1*/6-1);

xt=RESO-1;
yt=RESO-1;
zt=RESO-1;
if(DMS<4) ut=0;else ut=RESO-1;
if(DMS<5) vt=0;else vt=RESO-1;
if(DMS<6) wt=0;else wt=RESO-1;
if(DMS<7) rt=0;else rt=RESO-1;
if(DMS<8) st=0;else st=RESO-1;
if(DMS<9) tt=0;else tt=RESO-1;
if(DMS<10) ot=0;else ot=RESO-1;
/* here A6 *//* add*/
/*if(DMS<11) pt=0;else pt=RESO-1;
```

```
*/

mallocs();
for(i=1;i<DMS_max;i++) jpow[i]=pow(RESO,i);
arrayreset();
/*if(!PBC){
cubes=2+4+0+0;
vall=pow(RESO,DMS-1)*2+pow(RESO-2,DMS-1)*cubes;
printf(" %lld\n",vall);
}*/

printf(" \n");

/*999*/
while(1){
/*printf(" sn_:%d GRPH:%d\n",sn_,GRPH);*/

time(&nowtime);
strcpy(str,ctime(&nowtime));
i=0;
while(1){
if(str[i]==':') break;
i++;
}
strncpy(str,&str[i-2],5);
str[5]='\0';
printf(" %s\n",str);

if(DMS>3 && Xlib==2 && GRPH/* && cnt==CNT-1*/){
cleardevice_(1,0,0,XRESO,YRESO);

clearZbuf();
axes_eye(1+0,1+0,1+0,110,170,100,"x","y","z");
projection(0,0,0,&xs,&ys);
puts_(16,xs-1.2*UdX,ys-0.3*UdY,"O");

if(DMS==5) val=4;
else val=6.8;

clearZbuf();
set_O(0+xo,0+yo);
projection((x=0),(y=val*RESO*PS),(z=0),&xs,&ys);
set_O(0+xs,0+ys);
axes_eye(1+0,1+0,1+0,110,170,100,"x","y","z");
puts_(16,xs-1.2*UdX,ys-0.3*UdY,"O");

clearZbuf();
set_O(0+xo,0+yo);
projection((x=val*RESO*PS),(y=0),(z=0),&xs,&ys);
set_O(0+xs,0+ys);
axes_eye(1+0,1+0,1+0,110,170,100,"x","y","z");
puts_(16,xs-1.2*UdX,ys-0.3*UdY,"O");

clearZbuf();
set_O(0+xo,0+yo);
projection((x=val*RESO*PS),(y=val*RESO*PS),(z=0),&xs,&ys);
set_O(0+xs,0+ys);
axes_eye(1+0,1+0,1+0,110,170,100,"x","y","z");
puts_(16,xs-1.2*UdX,ys-0.3*UdY,"O");

clearZbuf();
set_O(0+xo,0+yo);

BitBlt_full();
}
else if(Xlib==1 && GRPH){
cleardevice_(1,0,0,XRESO,YRESO);
clearZbuf();
axes_eye(1+0,1+0,1+0,110+165,170+165,100+165,"x","y","z");
BitBlt_full();
}

time(&oldtime);

if(refill==0) break;
field();
if(refill==0) break;
#if !PBC
if(cnt==0) getsum();
#endif
printf(" sum:%lld\n",sum);
cag_r(oldtime);
check_rcount();

time(&nowtime);
dtime=nowtime-oldtime;
if(dtime<0){
dtime=2147483647-oldtime+nowtime;
printf(" %.2f [Om]\n",dtime/60.);
}
else{
x=dtime/60.;
printf(" %.2f [m]\n",x);
if(tmp3>-1) printf(" ratio:%.2f\n",x/tmp3);
}

printf(" \n");
if(refill<=0) break;

if(GRPH){
beep(50);

delay_(6000);
if(pauseflag==1) {pauseflag=0;use_subroop();}
}/**if(GRPH)**/
}/**while(1)**/

closegraph_();

return 0;
```

```
}/** main **/



void fprintf_(char flag,int i,int plane,int algo,char *str)
{
FILE *fp_;

if(!flag || cnt<CNT) return;

if(flag==1){
#if 0
if(algo==-1) fprintf(fp_," i:%d %d %d\n",i,-1,-1);
#else
fp_=fopen("Cpage.bin","ab");
if(i==0)
fprintf(fp_," %lld %lld i:%d plane:%d algo:%d%s\n",cnt,pcount[0]+1,i,plane,algo%2,str);
else if(i==CPMAX-1)
fprintf(fp_," %lld %lld i:%d plane:%d algo:%d%s\n\n",cnt,pcount[0],i,plane,algo%2,str);
else
fprintf(fp_," %lld %lld i:%d plane:%d algo:%d%s\n",cnt,pcount[0],i,plane,algo%2,str);
fclose(fp_);
#endif
}
else if(flag==2){
if(i==0)
printf(" %lld %lld i:%d plane:%d algo:%d%s\n",cnt,pcount[0]+1,i,plane,algo%2,str);
else if(i==CPMAX-1)
printf(" %lld %lld i:%d plane:%d algo:%d%s\n\n",cnt,pcount[0],i,plane,algo%2,str);
else
printf(" %lld %lld i:%d plane:%d algo:%d%s\n",cnt,pcount[0],i,plane,algo%2,str);
}
else if(flag==3){
fp_=fopen("Cpage.bin","ab");
fprintf(fp_," %d %d %d %s\n",i,plane,algo,str);
fclose(fp_);
}
}/** fprintf_ **/



#if !WX
void ls_image(char flag,char *file,int x,int y,int dx,int dy)
{
unsigned long xsize,ysize,size;
unsigned long width,height,imagesize;
unsigned long bits,bytesPerPixel,lineSizeDW,lineSize;
HDC hdce,hdc;
HBITMAP hbitmape;
BITMAPFILEHEADER bfh;
BITMAPINFOHEADER bih;
BYTE *gdata;
FILE *fpo,*fpi;

if(flag<=3){                            /* save */
if((fpo=fopen(file,"wb"))==NULL) {printf("Can't open a file.\n");return;}
```

```
width=dx;
height=dy;

bits=/*16*/24/*32*/;
bytesPerPixel=bits/8;
lineSizeDW=bytesPerPixel*width;
lineSizeDW=ICEIL(lineSizeDW,sizeof(long));
lineSize=lineSizeDW*sizeof(long);
imagesize=lineSize*height;

bfh.bfType=0x4d42;                  /* "BM" */
bfh.bfSize=54+imagesize;
bfh.bfReserved1=0;
bfh.bfOffBits=54;
bfh.bfReserved2=0;

bih.biSize=40;
bih.biWidth=width;
bih.biHeight=height;
bih.biPlanes=1;
bih.biBitCount=bits;
bih.biCompression=0;
bih.biSizeImage=imagesize;
bih.biXPelsPerMeter=0;
bih.biYPelsPerMeter=0;
bih.biClrUsed=0;
bih.biClrImportant=0;

if(flag<=1)
/*hdce=CreateCompatibleDC(hdctmp2)*/;
else if(flag==2)
hdce=CreateCompatibleDC(hdctmp1);
else{
hdc=CreateDC("DISPLAY",NULL,NULL,NULL);
hdce=CreateCompatibleDC(hdc);
}

hbitmape=CreateDIBSection(hdce,(LPBITMAPINFO)&bih,DIB_RGB_COLORS,&gdata,NULL,0);
SelectObject(hdce,hbitmape);

if(flag<=1)
/*BitBlt(hdce,0,0,dx,dy,hdctmp2,x,y,SRCCOPY)*/;
else if(flag==2)
BitBlt(hdce,0,0,dx,dy,hdctmp1,x,y,SRCCOPY);
else
BitBlt(hdce,0,0,dx,dy,hdc,x,y,SRCCOPY);

size=bih.biSizeImage;

fwrite(&bfh,14,1,fpo);
fwrite(&bih,40,1,fpo);
fwrite(gdata,size,1,fpo);

fclose(fpo);
```

```
if(flag==3) DeleteDC(hdc);
DeleteDC(hdce);
DeleteObject(hbitmape);

printf(" SAVE\n");
}
else{                                /* load */
if((fpi=fopen(file,"rb"))==NULL) {printf("Can't open the file.\n");return;}

fread(&bfh,14,1,fpi);
if(bfh.bfType!=0x4d42) {fclose(fpi);printf("Not BM.\n");return;}
fread(&bih,40,1,fpi);

fseek(fpi,bfh.bfOffBits,0);
size=bih.biSizeImage;
gdata=(BYTE *)malloc(size);
fread(gdata,size,1,fpi);

/*StretchDIBits(hdctmp2,x,y,bih.biWidth,bih.biHeight,0,0,bih.biWidth,bih.biHeight,
              gdata,(LPBITMAPINFO)&bih,DIB_RGB_COLORS,SRCCOPY);*/

fclose(fpi);
free(gdata);
}
}/** ls_image **/
#else
#if Xlib
void ls_image(char flag,char *file,int x,int y,int dx,int dy)
{
unsigned long xsize,ysize,size;
unsigned long unitbytes,width,height,bits_per_pixel,bytes_per_line;
unsigned long i,j,k,k_,knew,oddbytes,dksum;
int c0,c1,c2;
unsigned long heightdiv2,dbx;
unsigned char *buf_,*buf,*bf,*swap;
FILE *fpo,*fpi;

typedef struct {
unsigned char bfType[2];
unsigned long bfSize;
unsigned short bfReserved1;
unsigned short bfOffBits;
unsigned long bfReserved2;
} bfhset;
bfhset bfh;

typedef struct {
unsigned long biSize;
unsigned long biWidth;
unsigned long biHeight;
unsigned short biPlanes;
unsigned short biBitCount;
unsigned long biCompression;
unsigned long biSizeImage;
unsigned long biXPelsPerMeter;
unsigned long biYPelsPerMeter;
unsigned long biClrUsed;
unsigned long biClrImportant;
} bihset;
bihset bih;

if(flag<=3){                        /* save */
if(depth==16)      {unitbytes=2;printf(" 16bpp\n");}
else if(depth==24) {unitbytes=4;printf(" 24bpp\n");}
else               {printf("Depth unsuitable.\n");return;}

if((fpo=fopen(file,"wb"))==NULL) {printf("Can't open a file.\n");return;}

if(flag<=1)
/*image=XGetImage(d,pmap2,x,y,dx,dy,AllPlanes,ZPixmap)*/;
else if(flag==2)
image=XGetImage(d,pmap1,x,y,dx,dy,AllPlanes,ZPixmap);
else
image=XGetImage(d,rw,x,y,dx,dy,AllPlanes,ZPixmap);

width=image->width;
height=image->height;

oddbytes=(XRESO*3)%4;
if(oddbytes==0){
buf=(unsigned char *)malloc(1L*XRESO*3*YRESO);
swap=(unsigned char *)malloc(XRESO*3);
}
else{
buf=(unsigned char *)malloc(1L*(XRESO*3+(4-oddbytes))*YRESO);
swap=(unsigned char *)malloc(XRESO*3+(4-oddbytes));
}

if((oddbytes=(width*3)%4)==0){
size=width*height;
for(i=0;i<size;i++){
if(unitbytes==4){                   /* 4:24bpp, 2:16bpp */
buf[i*3+0]=image->data[i*4+0];
buf[i*3+1]=image->data[i*4+1];
buf[i*3+2]=image->data[i*4+2];
}
else{
c0=image->data[i*2+1];
c1=image->data[i*2+0];
                                    /* red, green, blue */
buf[i*3+2]=((c0 >> 3) & 31)*255/31;
buf[i*3+1]=(((c0 & 0x07) << 2) | ((c1 >> 6) & 0x03))*255/31;
buf[i*3+0]=(c1 & 31)*255/31;
}
}

bytes_per_line=width*3;
}/**if(oddbytes)**/
else{
k=0;k_=0;dksum=0;
for(j=0;j<height;j++){
for(i=0;i<width;i++){
if(unitbytes==4){
buf[k*3+0+dksum]=image->data[k_*4+0];
buf[k*3+1+dksum]=image->data[k_*4+1];
buf[k*3+2+dksum]=image->data[k_*4+2];
}
else{
c0=image->data[k_*2+1];
c1=image->data[k_*2+0];
                                    /* red, green, blue */
buf[k*3+2+dksum]=((c0 >> 3) & 31)*255/31;
buf[k*3+1+dksum]=(((c0 & 0x07) << 2) | ((c1 >> 6) & 0x03))*255/31;
buf[k*3+0+dksum]=(c1 & 31)*255/31;
}

k++;k_++;
}

if(unitbytes==2) k_+=(width*3)%2;   /* 16bpp */

knew=k*3+0+dksum;
for(i=0;i<4-oddbytes;i++){
buf[knew]=0;

knew++;
}

dksum+=(4-oddbytes);
}/**for(j)**/

bytes_per_line=width*3+(4-oddbytes);
}/**else(oddbytes)*/

/*printf(" size=%ld\n",bytes_per_line*height);
printf(" %d %d %d\n",width,height,width*height*3);*/

strcpy(bfh.bfType,"BM");
/*bfh.bfSize=bytes_per_line*height/65536;*/
bfh.bfSize=54+bytes_per_line*height;
bfh.bfReserved1=0;
bfh.bfOffBits=54;
bfh.bfReserved2=0;

bih.biSize=40;
bih.biWidth=width;
bih.biHeight=height;
bih.biPlanes=1;
bih.biBitCount=8*3;                 /* 24bpp */
bih.biCompression=0;
bih.biSizeImage=bytes_per_line*height;
bih.biXPelsPerMeter=2925;
bih.biYPelsPerMeter=2925;
bih.biClrUsed=0;
bih.biClrImportant=0;
```

```
size=bih.biSizeImage;
heightdiv2=height/2;
dbx=bytes_per_line;
for(i=0;i<heightdiv2;i++){
memmove(swap,&buf[i*dbx],dbx);
memmove(&buf[i*dbx],&buf[size-(i+1)*dbx],dbx);
memmove(&buf[size-(i+1)*dbx],swap,dbx);
}

fwrite(&bfh,14,1,fpo);
fwrite(&bih,40,1,fpo);
size=bih.biSizeImage;
fwrite(buf,size,1,fpo);

fclose(fpo);
free(buf);
free(swap);

printf(" SAVE\n");
}
else{                                /* load */
if(depth==16)      {printf("16bpp\n");}
else if(depth==24) {printf("24bpp\n");}
else               {printf("Depth unsuitable.\n");return;}

if((fpi=fopen(file,"rb"))==NULL) {printf("Can't open the file.\n");return;}

fread(&bfh,14,1,fpi);
if(strncmp(bfh.bfType,"BM",2)!=0) {fclose(fpi);printf("Not BM.\n");return;}
fread(&bih,40,1,fpi);

fseek(fpi,bfh.bfOffBits,0);
size=bih.biSizeImage;
buf_=(unsigned char *)malloc(size);
fread(buf_,size,1,fpi);

fclose(fpi);

width=bih.biWidth;
height=bih.biHeight;
bits_per_pixel=bih.biBitCount;
bytes_per_line=bih.biSizeImage/bih.biHeight;

oddbytes=(width*3)%4;
if(oddbytes==0)
swap=(unsigned char *)malloc(width*3);
else
swap=(unsigned char *)malloc(width*3+(4-oddbytes));

size=bih.biSizeImage;
heightdiv2=height/2;
dbx=bytes_per_line;
for(i=0;i<heightdiv2;i++){
memmove(swap,&buf_[i*dbx],dbx);
memmove(&buf_[i*dbx],&buf_[size-(i+1)*dbx],dbx);
memmove(&buf_[size-(i+1)*dbx],swap,dbx);
}

buf=(unsigned char *)malloc(width*height*3);
bf=(unsigned char *)malloc(width*height*4);

if((oddbytes=(width*3)%4)==0){
size=width*height;
for(i=0;i<size;i++){
buf[i*3+0]=buf_[i*3+0];
buf[i*3+1]=buf_[i*3+1];
buf[i*3+2]=buf_[i*3+2];
}
}/**if(oddbytes)**/
else{
k=0;k_=0;dksum=0;
for(j=0;j<height;j++){
for(i=0;i<width;i++){
buf[k*3+0]=buf_[k_*3+0+dksum];
buf[k*3+1]=buf_[k_*3+1+dksum];
buf[k*3+2]=buf_[k_*3+2+dksum];

k++;k_++;
}

dksum+=(4-oddbytes);
}/**for(j)**/
}/**if(oddbytes)**/

if(depth==16){
size=width*height;
for(i=0;i<size;i++){
/*c0=buf[i*3+0]*31/255;
c1=buf[i*3+1]*31/255;
c2=buf[i*3+2]*31/255;

buf[i*3+0]=0;
buf[i*3+1]=(c0<<3) | ((c1>>2) & 0x07);
buf[i*3+2]=(((c1 & 0x03) << 6) | c2)+32;*/

c0=buf[i*3+2]*31/255;                    /* bmp */
c1=buf[i*3+1]*31/255;
c2=buf[i*3+0]*31/255;

buf[i*3+2]=0;
buf[i*3+1]=(c0<<3) | ((c1>>2) & 0x07);
buf[i*3+0]=(((c1 & 0x03) << 6) | c2)+32;
}
}/**if(depth)**/

size=width*height;
for(i=0;i<size;i++){
bf[i*4+0]=buf[i*3+0];
bf[i*4+1]=buf[i*3+1];
```

```
bf[i*4+2]=buf[i*3+2];
}

vis=DefaultVisual(d,screen);

image=XCreateImage(d,vis,depth,ZPixmap,0,bf,width,height,32,width*4);

image->byte_order=LSBFirst;
image->bitmap_bit_order=LSBFirst;
image->bits_per_pixel=8*4;

/*XPutImage(d,pmap2,gcdisplay,image,0,0,x,y,width,height);*/

free(buf_);
free(buf);
free(bf);
free(swap);
}
}/** ls_image **/
#endif
#endif


void use_subroop(void)
{
char function_old,charflag_old;

usflag=1;

function_old=function;function=2;
charflag_old=charflag;

yorn=subroop();

function=function_old;
charflag=charflag_old;
}/** use_subroop **/


unsigned char subroop(void)
{
charflag=1;

while(1){
kbhit_();
if(charflag==0  || refill==0) return charcode;
}
}/** subroop **/


#if !WX
void keydowns_f2(void)
{
int dy;
```

```
if(GKS(VK_ESCAPE)<0 || GKS(VK_PAUSE)<0) charflag=0;
else if(GKS('S')<0){
ls_image(2,"ss.bmp",0,0,XRESO,YRESO);
beep(300);
}
}/** keydowns_f2 **/
#else
#if Xlib
void keydowns_f2(void)
{
int dy;

if(GKS(XK_Escape)<0 || GKS(XK_Pause)<0) charflag=0;
else if(GKS('S')<0 || GKS('s')<0){
ls_image(2,"ss.bmp",0,0,/*XRESO*/800,/*YRESO*/400);
beep(300);
}
}/** keydowns_f2 **/
#endif
#endif


void restore_in_PAINT(void)
{
#if !WX
ValidateRect(hwnd,NULL);
#endif

bitblt(1,0,0,XRESO,YRESO,0,0);
}/** restore_in_PAINT **/


#if WX
#if Xlib
void initsysfont(int type)
{
if(type==0){                        /* small */
strcpy(fs1,fs2[0]);
strcat(fs1,"-*-*-*-*-*-*");         /* scalable */

font_fs=XCreateFontSet(d,fs1,&mlist,&mcount,&def);
XFontsOfFontSet(font_fs,&info,&flist);
asct=(*info)->ascent;
}
else if(type==-1){                  /* medium (math) */
strcpy(fs1,fs2[1]);
strcat(fs1,"-*-*-*-*-*-*");         /* scalable */

font_fs=XCreateFontSet(d,fs1,&mlist,&mcount,&def);
XFontsOfFontSet(font_fs,&info,&flist);
asct=(*info)->ascent;
}
else if(type==1){                   /* medium */
strcpy(fs1,fs2[2]);
strcat(fs1,"-*-*-*-*-*-*");         /* scalable */
```

```
font_fs=XCreateFontSet(d,fs1,&mlist,&mcount,&def);
XFontsOfFontSet(font_fs,&info,&flist);
asct=(*info)->ascent;
}
else{                               /* large */
strcpy(fs1,fs2[3]);
strcat(fs1,"-*-*-*-*-*-*");         /* scalable */

font_fs=XCreateFontSet(d,fs1,&mlist,&mcount,&def);

if(mcount>0)
font_fs=XCreateFontSet(d,"-*-*-medium-r-normal--14-*",&mlist,&mcount,&def);

XFontsOfFontSet(font_fs,&info,&flist);
asct=(*info)->ascent;
}
}/** initsysfont **/
#endif
#endif



/* here A4 */                                                /* add int p */
INT pfunc(int RW,int x,int y,int z,int u,int v,int w,int r,int s,int t,int o,
          INT pcolor)
{
unsigned char dlt;
long long dist,bytes;
static short count=0,INTsize;
static unsigned char byte_clear[8],byte_write[8],val[2];
static INT val_INT;

#if PAL==0
if(!count){
INTsize=sizeof(INT);

count++;
}

/* here A7 *//* add +p*jpow[10] */
dist=x+y*jpow[1]+z*jpow[2]+u*jpow[3]+v*jpow[4]+w*jpow[5]+r*jpow[6]+s*jpow[7]+
     t*jpow[8]+o*jpow[9];  /* bytes */

if(!RW){                          /* read */
if(!pixelflag){
return pixel[dist];
}
else{
dist*=INTsize;
fseek(fp_pixel,dist,0);
fread(&val_INT,1,INTsize,fp_pixel);
return val_INT;
}
}
else{                               /* write */
```

```
if(!pixelflag){
pixel[dist]=pcolor;
}
else{
dist*=INTsize;
fseek(fp_pixel,dist,0);
val_INT=pcolor;
fwrite(&val_INT,1,INTsize,fp_pixel);
}
return 0;
}
#else
if(!count){
byte_clear[0]=255-1;
byte_clear[1]=255-2;
byte_clear[2]=255-4;
byte_clear[3]=255-8;
byte_clear[4]=255-16;
byte_clear[5]=255-32;
byte_clear[6]=255-64;
byte_clear[7]=255-128;

byte_write[0]=1;
byte_write[1]=2;
byte_write[2]=4;
byte_write[3]=8;
byte_write[4]=16;
byte_write[5]=32;
byte_write[6]=64;
byte_write[7]=128;

count++;
}

/* here A7 *//* add +p*jpow[10] */
dist=x+y*jpow[1]+z*jpow[2]+u*jpow[3]+v*jpow[4]+w*jpow[5]+r*jpow[6]+s*jpow[7]+
     t*jpow[8]+o*jpow[9];  /* bits */
bytes=dist/8;dlt=dist%8;

if(!pixelflag){
val[0]=pixel[bytes];
}
else{
fseek(fp_pixel,bytes,0);
fread(&val[0],1,1,fp_pixel);
}

if(!RW){                          /* read */
if(dlt) {val[0]=val[0] >> dlt;}
val[0]=val[0] & 1;
return val[0];
}
else{                                /* write */
val[0]=val[0] & byte_clear[dlt];
if(pcolor) val[0]=val[0] | byte_write[dlt];
if(!pixelflag){
pixel[bytes]=val[0];
}
else{
fseek(fp_pixel,bytes,0);
fwrite(&val[0],1,1,fp_pixel);
}
return 0;
}
#endif
}/** pfunc **/


void mallocs(void)
{
long long bytes_max,bytes_max_;

bytes_max=pow(RESO,DMS);

#if !M_or_D
#if PAL==0
pixel=(/*unsigned char*/INT *)malloc(sizeof(/*unsigned char*/INT)*bytes_max);
if(pixel!=NULL) printf(" mallocs(0):%lld [b]\n",bytes_max);
else            printf(" pixel:NULL\n");
#else
bytes_max_=bytes_max/8+1;
pixel=(/*unsigned char*/INT *)malloc(sizeof(/*unsigned char*/INT)*bytes_max_);
if(pixel!=NULL) printf(" mallocs(1):%lld [b]\n",bytes_max_);
else            printf(" pixel:NULL\n");
#endif
#else
pixel=NULL;
#endif

if(pixel!=NULL){
pixelflag=0;
}
else{
pixelflag=1;
if((fp_pixel=FOPEN("pixel.bin","w+b"))==NULL) exit(0);
if(!M_or_D) printf(" memory -> disk\n");
else        printf(" disk\n");
}
}/** mallocs **/


void frees(void)
{
if(!pixelflag) free(pixel);
else          {fclose(fp_pixel);unlink("pixel.bin");}
}/** frees **/


int initgraph_(void)
{
```

```
#if !WX
WNDCLASS wndclass;

hinstance=GetModuleHandle(NULL);

wndclass.hInstance    =hinstance;
wndclass.lpszClassName="CAGCLASS";
wndclass.lpszMenuName =NULL;
wndclass.lpfnWndProc  =wndproc_by_kbhit_;
wndclass.style        =0;
wndclass.hIcon        =LoadIcon(hinstance,"MYICON");
wndclass.hCursor      =LoadCursor(NULL,IDC_ARROW);
wndclass.cbClsExtra   =0;
wndclass.cbWndExtra   =0;
if(WB==0)
wndclass.hbrBackground=GetStockObject(WHITE_BRUSH);
else
wndclass.hbrBackground=GetStockObject(BLACK_BRUSH);

if(RegisterClass(&wndclass)==0) return 1;

hwnd=CreateWindow("CAGCLASS"," CAG",
                  /*WS_POPUP,*/
                  WS_OVERLAPPED | WS_CAPTION | WS_SYSMENU | WS_MINIMIZEBOX,
                  300,0,/*XRESO*/1024,YRESO,
                  NULL,NULL,hinstance,NULL);
if(hwnd==NULL) {MessageBox(NULL,"Memory space is not left.","CAG",MB_OK);return 1;}

SetWindowPos(hwnd,HWND_TOP,0,0,0,0,SWP_NOMOVE | SWP_NOSIZE);
ShowWindow(hwnd,SW_SHOWDEFAULT);

hdcdisplay=GetDC(hwnd);

hbitmap1=CreateCompatibleBitmap(hdcdisplay,XRESO,YRESO);
hdctmp1=CreateCompatibleDC(hdcdisplay);  /* text, dialog, menu */
SelectObject(hdctmp1,hbitmap1);
SetBkMode(hdcdisplay,TRANSPARENT);
SetBkMode(hdctmp1,TRANSPARENT);
SetBkColor(hdcdisplay,PALETTE(bfset[WB].back));
SetBkColor(hdctmp1,PALETTE(bfset[WB].back));
#else
#if Xlib
if((d=XOpenDisplay(""))==NULL) return 1;
screen=DefaultScreen(d);
cmap=DefaultColormap(d,screen);

rw=DefaultRootWindow(d);
ww=XCreateSimpleWindow(d,rw,300,0,/*XRESO*/1024,YRESO,0,
                       irgb[bfset[WB].back].pixel,
                       irgb[bfset[WB].back].pixel);
sh.flags=PPosition | PSize;
sh.x=0;sh.y=0;
sh.width=XRESO;sh.height=YRESO;
XSetStandardProperties(d,ww,appliname,appliname,None,argv_,argc_,&sh);
```

```
atm1=XInternAtom(d,"WM_PROTOCOLS",False);
atm2=XInternAtom(d,"WM_DELETE_WINDOW",False);
XSetWMProtocols(d,ww,&atm2,1);

XSelectInput(d,ww,KeyPressMask | ButtonPressMask | PointerMotionMask | ExposureMask |
                  StructureNotifyMask | SubstructureNotifyMask);
if(GRPH) {XMapWindow(d,ww);XFlush(d);}

gcdisplay=XCreateGC(d,ww,0,NULL);

depth=DefaultDepth(d,screen);
pmap1=XCreatePixmap(d,ww,XRESO,YRESO,depth);  /* text, dialog, menu */

cursor=XCreateFontCursor(d,XC_arrow);
XDefineCursor(d,ww,cursor);

XSetLineAttributes(d,gcdisplay,/*2*/1,LineSolid,CapButt,JoinMiter);

initsysfont(0);
#endif
#endif

initpalette();
signal(SIGINT,terminator);

return 0;
}/** initgraph_ **/


void terminator(int sig)
{
signal(sig,SIG_IGN);

printf(" Ctrl+C\n");
if(fieldflag) exit(0);
else          refill=0;
}/** terminator **/


void closegraph_(void)
{
int i;
char str[32],buf[10];

for(i=0;i<CPMAX;i++){
strcpy(str,"rtn[");
#if !WX
strcat(str,itoa(i,buf,10));
#else
strcat(str,gcvt(i,8,buf));
#endif
strcat(str,"].bin");
unlink(str);
}
```

```
frees();

#if !WX
DeleteObject(hbitmap1);
DeleteDC(hdctmp1);

/*EndPaint(hwnd,&paintstruct);*/
ReleaseDC(hwnd,hdcdisplay);
DestroyWindow(hwnd);
/*UnregisterClass("CAGCLASS",hinstance);*/
#else
#if Xlib
/*XFreeFontSet(d,font_fs);*/
XFreeCursor(d,cursor);
XFreePixmap(d,pmap1);
XFreeGC(d,gcdisplay);

XFreeColormap(d,cmap);
XDestroyWindow(d,ww);XFlush(d);
XCloseDisplay(d);
#endif
#endif
}/** closegraph_ **/


void initpalette(void)
{
#if !WX || Xlib
int i,i_;
dbl dlt;

irgb[0].red=0;irgb[0].green=0;irgb[0].blue=0;

irgb[9].red=0;irgb[9].green=0;irgb[9].blue=255;  /* blue */
irgb[10].red=0;irgb[10].green=255;irgb[10].blue=0;  /* green */
irgb[11].red=0;irgb[11].green=255;irgb[11].blue=255;  /* cyan */
irgb[12].red=255;irgb[12].green=0;irgb[12].blue=0;  /* red */
irgb[13].red=255;irgb[13].green=0;irgb[13].blue=255;  /* magenta */
irgb[14].red=255;irgb[14].green=255;irgb[14].blue=0;  /* yellow */

irgb[15].red=255;irgb[15].green=255;irgb[15].blue=255;

for(i=1;i<7;i++){                   /* 1 -> 6 */
if(irgb[9+(i-1)].red==255)
irgb[i].red=irgb[9+(i-1)].red-24*1;
if(irgb[9+(i-1)].green==255)
irgb[i].green=irgb[9+(i-1)].green-24*1;
if(irgb[9+(i-1)].blue==255)
irgb[i].blue=irgb[9+(i-1)].blue-24*1;
}

for(i=7;i<9;i++){                   /* 7, 8 */
irgb[i].red=128+32*(8-i);
irgb[i].green=irgb[i].red;
irgb[i].blue=irgb[i].red;
}

dlt=255./(VGACOLORS-15);

for(i=16;i<VGACOLORS;i++){
i_=VGACOLORS-i;
irgb[i].red=dlt*i_;
irgb[i].green=irgb[i].red;
irgb[i].blue=irgb[i].red;
}
#endif

#if WX
#if Xlib
for(i=0;i<VGACOLORS;i++){
irgb[i].red=ff_fc(irgb[i].red*TRANS);
irgb[i].green=ff_fc(irgb[i].green*TRANS);
irgb[i].blue=ff_fc(irgb[i].blue*TRANS);
}

for(i=0;i<VGACOLORS;i++)
XAllocColor(d,cmap,&irgb[i]);

XParseColor(d,cmap,"cyan",&c);
XAllocColor(d,cmap,&c);
#endif
#endif
}/** initpalette **/


void BitBlt_full(void)
{
bitblt(1,0,0,XRESO,YRESO,0,0);
}/** BitBlt_full **/



#if !WX
void bitblt(char flag,int x,int y,int xsize,int ysize,int x_,int y_)
{
BitBlt(hdcdisplay,x_,y_,xsize,ysize,
       hdctmp1,x,y,SRCCOPY);
}/** bitblt **/
#else
void bitblt(char flag,int x,int y,int xsize,int ysize,int x_,int y_)
{
#if Xlib
XCopyArea(d,pmap1,ww,gcdisplay,x,y,xsize,ysize,
                              x_,y_);

XFlush(d);
#endif
}/** bitblt **/
#endif
```

```
#if !WX
void cleardevice_(char flag,int x,int y,int xsize,int ysize)
{
PatBlt(hdctmp1,x,y,xsize,ysize,bfset[WB].back_);
}/** cleardevice_ **/
#else
#if Xlib
void cleardevice_(char hdc,int x,int y,int xsize,int ysize)
{
XSetForeground(d,gcdisplay,irgb[bfset[WB].back].pixel);

XFillRectangle(d,pmap1,gcdisplay,x,y,xsize,ysize);
}/** cleardevice_ **/
#endif
#endif



#if !WX
COLORREF PALETTE(int color)
{
return RGB(irgb[color].red,irgb[color].green,irgb[color].blue);
}/** PALETTE **/
#endif



#if !WX
void kbhit_(void)
{
MSG msg;

if(PeekMessage(&msg,NULL,0,0,PM_REMOVE)){
TranslateMessage(&msg);
DispatchMessage(&msg);
}
}/** kbhit_ */
#else
#if Xlib
int GKS(KeySym XK)
{
if(keysym==XK) return -1;
else return 0;
}/** GKS **/



int GKS_(long ModkeyMask)
{
if((event.xkey.state & ModkeyMask)>0) return -1;
else return 0;
}/** GKS_ **/
#endif



void kbhit_(void)
{
#if Xlib
```

```
int i;

if(XPending(d)){
XNextEvent(d,&event);

if(event.type==ClientMessage &&
   event.xclient.message_type==atm1 && event.xclient.data.l[0]==atm2){
printf(" Close\n");

if(fieldflag) exit(0);
else          refill=0;
}
else{
wndproc_filer();
}
}
#endif
}/** kbhit_ */
#endif



#if !WX
LRESULT CALLBACK  wndproc_by_kbhit_(HWND hwnd,UINT umsg,WPARAM wparam,LPARAM lparam)
{
if(wndproc_filer(hwnd,umsg,wparam,lparam)!=0) return 1;

return DefWindowProc(hwnd,umsg,wparam,lparam);
}/** wndproc_by_kbhit_ **/
#endif



#if !WX
int wndproc_filer(HWND hwnd,UINT umsg,WPARAM wparam,LPARAM lparam)
{
if(umsg==WM_KEYDOWN){
/*************************** menu keydowns -> ****************************/
/*************************** <- menu keydowns ****************************/

/*************************** dialog keydowns -> ****************************/
/*************************** <- dialog keydowns ****************************/

if(function==2){
keydowns_f2();
return 1;
}

if(usflag==1) usflag=0;

     if(GKS(VK_ESCAPE)<0 || GKS(VK_PAUSE)<0) refill=0;
else if(GKS_(VK_SHIFT)<0) pauseflag=1;

return 1;
}/**else if(umsg)**/
else if(umsg==WM_SYSKEYDOWN){
}/**else if(umsg)**/
```

```
else if(umsg==WM_CLOSE){
if(fieldflag) exit(0);
else          refill=0;

return 1;
}/**else if(umsg)**/
else if(umsg==WM_PAINT){
restore_in_PAINT();

return 1;
}/**else if(umsg)**/
else if(umsg==WM_LBUTTONDOWN){

return 1;
}/**else if(umsg)**/
else if(umsg==WM_RBUTTONDOWN){

return 1;
}/**else if(umsg)**/

return 0;
}/** wndproc_filer **/
#else
#if Xlib
int wndproc_filer(void)
{
int length;
/*static */char buf[10];

length=XLookupString((XKeyEvent *)&event,buf,10,&keysym,NULL);
buf[length]='\0';

if(event.type==KeyPress){
/************************** menu keydowns -> ****************************/
/************************** <- menu keydowns ****************************/

/************************** dialog keydowns -> ****************************/
/************************** <- dialog keydowns ****************************/

if(function==2){
keydowns_f2();
return 1;
}

if(usflag==1) usflag=0;

     if(GKS(XK_Escape)<0 || GKS(XK_Pause)<0) refill=0;
else if(GKS(XK_Shift_L)<0 || GKS(XK_Shift_R)<0) pauseflag=1;

return 1;
}/**if(event.type)**/
else if(event.type==Expose){
restore_in_PAINT();

return 1;
}/**else if(event.type)**/
else if(event.type==ButtonPress){

return 1;
}/**else if(event.type)**/

return 0;
}/** wndproc_filer **/
#endif
#endif



void delay_(long millisecond)
{
long oldtime,nowtime,dtime=0,old;
dbl i=CLOCKS_PER_SEC,j;

j=millisecond;
millisecond=j*(i/1000.);
oldtime=clock();

while(1){
kbhit_();
if(pauseflag==1 && refill==0) {pauseflag=0;refill=1;break;}
if(refill==0) break;

old=dtime;
nowtime=clock();dtime=nowtime-oldtime;
if(dtime>=millisecond) break;
if(dtime<0){
millisecond-=old;
oldtime=0;
}
}
}/** delay_ **/



void beep(long millisecond)
{
#if !WX
Beep(888,millisecond);
#endif
}/** beep **/



int ff_fc(dbl val_d)
{
int val_i,val;

val_i=floor(val_d);
val=(val_d-val_i<0.5)?val_i:val_i+1;

return val;
}/** ff_fc **/
```

```
long max_(long a,long b)
{
return (a>b)?a:b;
}/** max_ **/


long min_(long a,long b)
{
return (a<b)?a:b;
}/** min_ **/


#if !WX
void puts_(int h,int x,int y,char *str)
{
HFONT hfont;

hfont=CreateFont(h,h/2,0,0,
                 FW_NORMAL,/*TRUE*/0,0,0,
                 DEFAULT_CHARSET,OUT_DEFAULT_PRECIS,
                 CLIP_DEFAULT_PRECIS,DEFAULT_QUALITY,
                 FIXED_PITCH | FF_ROMAN/*FF_MODERN*/,NULL);

SetTextColor(hdctmp1,RGB(0,0,0));
TextOut(hdctmp1,x,y,str,strlen(str));

DeleteObject(hfont);
}/** puts_ **/
#else
void puts_(int h,int x,int y,char *str)
{
#if Xlib
XSetForeground(d,gcdisplay,irgb[0].pixel);

XmbDrawString(d,pmap1,font_fs,gcdisplay,x,y+XDSDY,str,strlen(str));
#endif
}/** puts_ **/
#endif


void axes_eye(char flagx,char flagy,char flagz,dbl lenx,dbl leny,dbl lenz,
              char *xname,char *yname,char *zname)
{
int xs,ys,dx,dy,ds,ds0=15,dys=8;

if(flagx>0) line_eye_(0,0,0,lenx,0,0,0);
if(flagy>0) line_eye_(0,0,0,0,leny,0,0);
if(flagz>0) line_eye_(0,0,0,0,0,lenz,0);

projection(lenx,0,0,&xs,&ys);
dx=xs-dO[0];
dy=ys-dO[1];
ds=sqrt(dx*dx+dy*dy);
if(ds)
```

```
puts_(16,xs+dx*ds0/ds-dys/2,ys+dy*ds0/ds-dys,xname);
else
puts_(16,xs+ds0,ys-dys,xname);

projection(0,leny,0,&xs,&ys);
dx=xs-d0[0];
dy=ys-d0[1];
ds=sqrt(dx*dx+dy*dy);
if(ds)
puts_(16,xs+dx*ds0/ds-dys/2,ys+dy*ds0/ds-dys,yname);
else
puts_(16,xs+ds0,ys-dys,yname);

projection(0,0,lenz,&xs,&ys);
dx=xs-d0[0];
dy=ys-d0[1];
ds=sqrt(dx*dx+dy*dy);
if(ds)
puts_(16,xs+dx*ds0/ds-dys/2,ys+dy*ds0/ds-dys,zname);
else
puts_(16,xs+ds0,ys-dys,zname);
}/** axes_eye **/



#if !WX
int ppixel(int nx,int ny,int pcolor)
{
if(!sn_) return 1;
if((nx<0)||(nx>XRESO-1)||(ny<0)||(ny>YRESO-1)) return 1;

if(Trianflag){
if(zg==9)  pcolor=8;
else       pcolor=15;
}
else if(pcolor>VGACOLORS-1) pcolor=0;

SetPixelV(hdcdisplay,nx,ny,PALETTE(pcolor));
SetPixelV(hdctmp1,nx,ny,PALETTE(pcolor));

return 0;
}/** ppixel **/
#else
int ppixel(int nx,int ny,int pcolor)
{
#if Xlib
if(!sn_) return 1;
if((nx<0)||(nx>XRESO-1)||(ny<0)||(ny>YRESO-1)) return 1;

if(Trianflag){
if(zg==9)  pcolor=8;
else       pcolor=15;
}
else if(pcolor>VGACOLORS-1) pcolor=0;
XSetForeground(d,gcdisplay,irgb[pcolor].pixel);
```

```
XDrawPoint(d,ww,gcdisplay,nx,ny);
XDrawPoint(d,pmap1,gcdisplay,nx,ny);
#endif

return 0;
}/** ppixel **/
#endif


void restore_work(char flag)
{
int i;

if(flag==0){
if(idx){
for(i=idx-1;i>-1;i--){
work[stack[i].x][stack[i].y].p=0;
}

idx=0;
}
}
else{
if(idx_Rect){
for(i=idx_Rect-1;i>-1;i--){
work_Rect[stack_Rect[i].x][stack_Rect[i].y]=0;
}

idx_Rect=0;
}
}
}/** restore_work **/


void hline(int left,int right,int y,int color)
{
int i,xs,ys;
dbl DX,DY,Dz,X,Y,Z,len,val;

DX=work[right][y].x-work[left][y].x;
DY=work[right][y].y-work[left][y].y;
Dz=work[right][y].z-work[left][y].z;

for(i=left+1;i<=right-1;i++){
xs=i+dO[0]-Zxdiv;
ys=y+dO[1]-Zydiv;

if(Zflag==0) ppixel(xs,ys,color);
else{
/*xb=x-dO[0]+Zx/2;
yb=y-dO[1]+Zy/2;*/
if(i<0 || i>Zx-1 || y<0 || y>Zy-1) ;
else{
val=1.*(i-left)/(right-left);
X=work[left][y].x+DX*val;
Y=work[left][y].y+DY*val;
Z=work[left][y].z+Dz*val;
len=sqrt(pow(xE-X,2)+pow(yE-Y,2)+pow(zE-Z,2));

if(len<Zbuf[i][y] || Zflag==2){
if(Zflag==1) Zbuf[i][y]=len;
if(work_Rect[i][y]==0) ppixel(xs,ys,color);
}

}
}
}/**for(i)**/
}/** hline **/



void line_eye_(dbl xd1,dbl yd1,dbl zd1,dbl xd2,dbl yd2,dbl zd2,int color)
{
int x1,y1,x2,y2,dx,dy,x,y;
int c,d,e,sx,sy;
int putflag=1,putcount=0;
int xb,yb;
dbl DX,DY,Dz,X,Y,Z,len,val,ds,tmp0,tmp1,tmp2;

projection(xd1,yd1,zd1,&x1,&y1);
projection(xd2,yd2,zd2,&x2,&y2);

xg=x2;yg=y2;

dx=x2-x1;
if(dx<0){
dx=x1;dy=y1;
x1=x2;y1=y2;
x2=dx;y2=dy;

tmp0=xd1;tmp1=yd1;tmp2=zd1;
xd1=xd2;yd1=yd2;zd1=zd2;
xd2=tmp0;yd2=tmp1;zd2=tmp2;
}

DX=xd2-xd1;
DY=yd2-yd1;
Dz=zd2-zd1;




dx=abs(x2-x1);
dy=abs(y2-y1);
ds=sqrt(dx*dx+dy*dy);

if(dx==0 && dy==0){
/* x1,y1 */
if(Zflag==0) ppixel(x1,y1,color);
else{
xb=x1-dO[0]+Zxdiv;
yb=y1-dO[1]+Zydiv;
```

```
if(xb<0 || xb>Zx-1 || yb<0 || yb>Zy-1) ;
else{
val=0;
X=xd1+DX*val;
Y=yd1+DY*val;
Z=zd1+Dz*val;
len=sqrt(pow(xE-X,2)+pow(yE-Y,2)+pow(zE-Z,2));

if(len<Zbuf[xb][yb] || Zflag==2){
if(Zflag==1) Zbuf[xb][yb]=len;
if(color==0){
ppixel(x1,y1,color);
}
else{
if(work_Rect[xb][yb]==0) {ppixel(x1,y1,color);work_Rect[xb][yb]=2;}
}
}
else if(color==0 && work_Rect[xb][yb]==2) ppixel(x1,y1,color);

stack[idx].x=xb;stack[idx].y=yb;idx++;
if(color==0) work_Rect[xb][yb]=1;
stack_Rect[idx_Rect].x=xb;stack_Rect[idx_Rect].y=yb;idx_Rect++;
work[xb][yb].p=1;
work[xb][yb].x=X;
work[xb][yb].y=Y;
work[xb][yb].z=Z;
}
}

return;
}

if(x1<=x2) sx=1;
else       sx=-1;
if(y1<=y2) sy=1;
else       sy=-1;

x=x1;
y=y1;

if(dx>=dy){
c=2*dy;d=2*(dy-dx);e=c-dx;

while(1){
if(putperiod==0) ;
else{
if(putcount==putperiod){
putcount=0;
if(putflag==1) putflag=0;else putflag=1;
}
putcount++;
}

if(putperiod==0 || putflag==1) ;else goto next_dx;
if(Zflag==0) ppixel(x,y,color);
else{
xb=x-d0[0]+Zxdiv;
yb=y-d0[1]+Zydiv;
if(xb<0 || xb>Zx-1 || yb<0 || yb>Zy-1) ;
else{
val=sqrt(pow(x-x1,2)+pow(y-y1,2))/ds;
X=xd1+DX*val;
Y=yd1+DY*val;
Z=zd1+Dz*val;
len=sqrt(pow(xE-X,2)+pow(yE-Y,2)+pow(zE-Z,2));

if(len<Zbuf[xb][yb] || Zflag==2){
if(Zflag==1) Zbuf[xb][yb]=len;
if(color==0){
ppixel(x,y,color);
}
else{
if(work_Rect[xb][yb]==0) {ppixel(x,y,color);work_Rect[xb][yb]=2;}
}
}
else if(color==0 && work_Rect[xb][yb]==2) ppixel(x,y,color);

stack[idx].x=xb;stack[idx].y=yb;idx++;
if(color==0) work_Rect[xb][yb]=1;
stack_Rect[idx_Rect].x=xb;stack_Rect[idx_Rect].y=yb;idx_Rect++;
work[xb][yb].p=1;
work[xb][yb].x=X;
work[xb][yb].y=Y;
work[xb][yb].z=Z;
}
}



next_dx:

if(e<0) e+=c;
else    {e+=d;y+=sy;}

x+=sx;
if(sx>=0) {if(x>x2) break;}
else      {if(x<x2) break;}
}
}/**if(dx,dy)**/
else{
c=2*dx;d=2*(dx-dy);e=c-dy;

while(1){
if(putperiod==0) ;
else{
if(putcount==putperiod){
putcount=0;
if(putflag==1) putflag=0;else putflag=1;
}
putcount++;
}

if(putperiod==0 || putflag==1) ;else goto next_dy;



if(Zflag==0) ppixel(x,y,color);
else{
xb=x-dO[0]+Zxdiv;
yb=y-dO[1]+Zydiv;
if(xb<0 || xb>Zx-1 || yb<0 || yb>Zy-1) ;
else{
val=sqrt(pow(x-x1,2)+pow(y-y1,2))/ds;
X=xd1+DX*val;
Y=yd1+DY*val;
Z=zd1+Dz*val;
len=sqrt(pow(xE-X,2)+pow(yE-Y,2)+pow(zE-Z,2));

if(len<Zbuf[xb][yb] || Zflag==2){
if(Zflag==1) Zbuf[xb][yb]=len;
if(color==0){
ppixel(x,y,color);
}
else{
if(work_Rect[xb][yb]==0) {ppixel(x,y,color);work_Rect[xb][yb]=2;}
}
}
else if(color==0 && work_Rect[xb][yb]==2) ppixel(x,y,color);

stack[idx].x=xb;stack[idx].y=yb;idx++;
if(color==0) work_Rect[xb][yb]=1;
stack_Rect[idx_Rect].x=xb;stack_Rect[idx_Rect].y=yb;idx_Rect++;
work[xb][yb].p=1;
work[xb][yb].x=X;
work[xb][yb].y=Y;
work[xb][yb].z=Z;
}
}




next_dy:

if(e<0) e+=c;
else    {e+=d;x+=sx;}

y+=sy;
if(sy>=0) {if(y>y2) break;}
else      {if(y<y2) break;}
}
}/**else(dx,dy)**/
}/** line_eye_ **/
```

```
void line_thph_eye(dbl th,dbl ph,dbl len,dbl x1,dbl y1,dbl z1,
                   int color)
{
char flag=1;
dbl x2,y2,z2;

if(len==0) return;
if(len<0) {flag=0;len=-len;}

x2=x1+len*sin(th*TORAD)*cos(ph*TORAD);
y2=y1+len*sin(th*TORAD)*sin(ph*TORAD);
z2=z1+len*cos(th*TORAD);

if(color>=0) line_eye_(x1,y1,z1,x2,y2,z2,color);

if(flag==1){
/*xg=x2;yg=y2;zg=z2;*/
tmp0=x2;tmp1=y2;tmp2=z2;
}
}/** line_thph_eye **/


void Trian(char flag,dbl x1,dbl y1,dbl z1,dbl x2,dbl y2,dbl z2,
           dbl x3,dbl y3,dbl z3,int color1,int color2,int color3,int color)
{
int i;
int ymin,ymax,y,xmin,xmax,xmin_,xmax_;

restore_work(0);
if(Rectflag==0) restore_work(1);

if(flag==0){
zg=color;

if(color3<0) {Trianflag=1;color3=0;}else Trianflag=0;
line_eye_(x1,y1,z1,x2,y2,z2,color3);
if(color1<0) {Trianflag=1;color1=0;}else Trianflag=0;
line_eye_(x2,y2,z2,x3,y3,z3,color1);
if(color2<0) {Trianflag=1;color2=0;}else Trianflag=0;
line_eye_(x3,y3,z3,x1,y1,z1,color2);
Trianflag=0;
}
else{
line_eye_(x1,y1,z1,x2,y2,z2,color3);
line_eye_(x2,y2,z2,x3,y3,z3,color1);
line_eye_(x3,y3,z3,x1,y1,z1,color2);
return;
}

if(idx==0) return;

i=0;ymin=Zy;
while(1){
```

```
if(stack[i].y<ymin) ymin=stack[i].y;

i++;if(i==idx) break;
}

i=0;ymax=-1;
while(1){
if(stack[i].y>ymax) ymax=stack[i].y;

i++;if(i==idx) break;
}

for(y=ymin;y<=ymax;y++){
i=0;xmin=Zx;
while(1){
if(stack[i].y==y && stack[i].x<xmin) xmin=stack[i].x;

i++;if(i==idx) break;
}

i=0;xmax=-1;
while(1){
if(stack[i].y==y && stack[i].x>xmax) xmax=stack[i].x;

i++;if(i==idx) break;
}

i=xmin;
while(1){
if(work[i+1][y].p==0) {xmin_=i;break;}

i++;if(i==Zx-1) break;
}

i=xmax;
while(1){
if(work[i-1][y].p==0) {xmax_=i;break;}

i--;if(i==0) break;
}

if(xmax_>=xmin_+2){
if(/*ig!=15*/1) hline(xmin_,xmax_,y,color);
}
}/**for(y)**/
}/** Trian **/



void Rect(dbl x1,dbl y1,dbl z1,dbl x2,dbl y2,dbl z2,
          dbl x3,dbl y3,dbl z3,dbl x4,dbl y4,dbl z4,int color)
{
restore_work(1);

Rectflag=1;
if(color<8 || color==9){
```

```
Trian(0,x3,y3,z3,x2,y2,z2,x1,y1,z1,-1,color,-1,color);
Trian(0,x1,y1,z1,x3,y3,z3,x4,y4,z4,-1,-1,color,color);
}
else{
Trian(0,x3,y3,z3,x2,y2,z2,x1,y1,z1,0,color,0,color);
Trian(0,x1,y1,z1,x3,y3,z3,x4,y4,z4,0,0,color,color);
}
Rectflag=0;
}/** Rect **/


void polygon(dbl vx[],dbl vy[],dbl vz[],int color)
{
Rect(vx[0],vy[0],vz[0],vx[1],vy[1],vz[1],vx[2],vy[2],vz[2],vx[3],vy[3],vz[3],color);
}/** polygon **/


void pixel_3d(int nx,int ny,int nz,int color)
{
dbl vx[4],vy[4],vz[4];

if(!GRPH) return;
if(color<0) return;

if(1){
/* -z */
vx[0]=nx;vy[0]=ny;vz[0]=nz;
vx[1]=nx+PS;vy[1]=ny;vz[1]=nz;
vx[2]=nx+PS;vy[2]=ny+PS;vz[2]=nz;
vx[3]=nx;vy[3]=ny+PS;vz[3]=nz;
polygon(vx,vy,vz,color);

/* +z */
vx[0]=nx;vy[0]=ny;vz[0]=nz+PS;
vx[1]=nx+PS;vy[1]=ny;vz[1]=nz+PS;
vx[2]=nx+PS;vy[2]=ny+PS;vz[2]=nz+PS;
vx[3]=nx;vy[3]=ny+PS;vz[3]=nz+PS;
polygon(vx,vy,vz,color);

/* -y */
vx[0]=nx;vy[0]=ny;vz[0]=nz;
vx[1]=nx;vy[1]=ny;vz[1]=nz+PS;
vx[2]=nx+PS;vy[2]=ny;vz[2]=nz+PS;
vx[3]=nx+PS;vy[3]=ny;vz[3]=nz;
polygon(vx,vy,vz,color);

/* +y */
vx[0]=nx;vy[0]=ny+PS;vz[0]=nz;
vx[1]=nx;vy[1]=ny+PS;vz[1]=nz+PS;
vx[2]=nx+PS;vy[2]=ny+PS;vz[2]=nz+PS;
vx[3]=nx+PS;vy[3]=ny+PS;vz[3]=nz;
polygon(vx,vy,vz,color);
}

if(1){
```

```
/* -x */
vx[0]=nx;vy[0]=ny;vz[0]=nz;
vx[1]=nx;vy[1]=ny+PS;vz[1]=nz;
vx[2]=nx;vy[2]=ny+PS;vz[2]=nz+PS;
vx[3]=nx;vy[3]=ny;vz[3]=nz+PS;
polygon(vx,vy,vz,color);

/* +x */
vx[0]=nx+PS;vy[0]=ny;vz[0]=nz;
vx[1]=nx+PS;vy[1]=ny+PS;vz[1]=nz;
vx[2]=nx+PS;vy[2]=ny+PS;vz[2]=nz+PS;
vx[3]=nx+PS;vy[3]=ny;vz[3]=nz+PS;

polygon(vx,vy,vz,color);
}

/*bitblt(1,min_x,min_y,max_x-min_x+1,max_y-min_y+1,min_x,min_y);*/
}/** pixel_3d **/



dbl det(dbl a11,dbl a12,dbl a13,dbl a21,dbl a22,dbl a23,
           dbl a31,dbl a32,dbl a33)
{
dbl val;

val=a11*a22*a33+a12*a23*a31+a13*a32*a21-a13*a22*a31-a12*a21*a33-a11*a32*a23;

return val;
}/** det **/



void initeye(void)
{
dbl sinth,costh,sinph,cosph;

if(th==0) th=360;

th*=TORAD;ph*=TORAD;
sinth=sin(th);costh=cos(th);
sinph=sin(ph);cosph=cos(ph);

mx[0][0]=-sinph;
mx[1][0]=cosph;
mx[2][0]=0;

mx[0][1]=-costh*cosph;
mx[1][1]=-costh*sinph;
mx[2][1]=sinth;

mx[0][2]=-sinth*cosph;
mx[1][2]=-sinth*sinph;
mx[2][2]=-costh;

mx[3][0]=0;
mx[3][1]=0;
```

```
mx[3][2]=rho;

xE=rho*sinth*cosph;
yE=rho*sinth*sinph;
zE=rho*costh;

DET=det(mx[0][0],mx[1][0],mx[2][0],mx[0][1],mx[1][1],mx[2][1],mx[0][2],mx[1][2],
        mx[2][2]);
}/** initeye **/


void set_O(int x,int y)
{
dO[0]=x;dO[1]=y;
}/** set_O **/


void projection(dbl x,dbl y,dbl z,int *xs,int *ys)
{
/*int dx,dy;*/
dbl xe,ye,ze;

/*dx=XRESO/2;dy=YRESO/2;*/

xe=mx[0][0]*x+mx[1][0]*y;
ye=mx[0][1]*x+mx[1][1]*y+mx[2][1]*z;
ze=mx[0][2]*x+mx[1][2]*y+mx[2][2]*z+mx[3][2];

*xs=ff_fc(dscr*xe/ze)+/*dx*/dO[0];
*ys=ff_fc(-dscr*ye/ze)+/*dy*/dO[1];
}/** projection **/


void set_Z(char flag)
{
Zflag=flag;
}/** set_Z **/


void clearZbuf(void)
{
int i,j;

for(j=0;j<Zy;j++)
for(i=0;i<Zx;i++){
Zbuf[i][j]=rho*5;
work_Rect[i][j]=0;
work[i][j].p=0;
}

idx=0;
idx_Rect=0;
}/** clearZbuf **/
```

```
void check_rcount(void)
{
int j;
long long val,sum;

if(GRPH>0){
for(j=0;j<CPMAX;j++)
printf(" %lld %lld\n",cnt,pcount[j]);
}
else{
val=pcount[0];
for(j=1;j<CPMAX;j++){
if(pcount[j]!=val) {beep(1000);refill=-1;break;}
}

if(refill==0){
}
else if(refill==-1){
printf(" %dd %lld %lld %d:%lld\n",DMS,cnt,val,j,pcount[j]);
}
else if(refill<=-2){
printf(" %dd %d %lld %lld\n",DMS,refill,cnt,val);
}
else{
#if PBC
printf(" %ddi %dCP ok:%lld %lld\n",DMS,CPMAX,cnt,val);
#else
printf(" %ddf %dCP ok:%lld %lld\n",DMS,CPMAX,cnt,val);
#endif
}
}

end:
if(refill<0){
for(j=0;j<CPMAX;j++) printf(" %lld\n",pcount[j]);
}

sum=0;
for(j=0;j<CPMAX;j++) sum+=pcount[j];
printf(" %lld\n",sum);
}/** check_rcount **/


void arrayreset(void)
{
/* here A3 */  /* add p */
int x,y,z,u,v,w,r,s,t,o;
int _t[DMS_limit];

_t[0]=xt;
_t[1]=yt;
_t[2]=zt;
_t[3]=ut;
_t[4]=vt;
_t[5]=wt;
```

```
_t[6]=rt;
_t[7]=st;
_t[8]=tt;
_t[9]=ot;
/* here A7 *//* add*/
/*_t[10]=pt;*/

/* here A8 *//* add*/
/*p=0;
while(1){
*/

o=0;
while(1){

t=0;
while(1){

s=0;
while(1){

r=0;
while(1){

w=0;
while(1){

v=0;
while(1){

u=0;
while(1){

z=0;
while(1){

y=0;
while(1){

x=0;
while(1){
/* here A9 */      /* add p */
pfunc(1,x,y,z,u,v,w,r,s,t,o,wall);

x++;
if(x==_t[0]+1) break;
}

y++;
if(y==_t[1]+1) break;
}

z++;
if(z==_t[2]+1) break;
}
u++;
if(u==_t[3]+1) break;
}

v++;
if(v==_t[4]+1) break;
}

w++;
if(w==_t[5]+1) break;
}

r++;
if(r==_t[6]+1) break;
}

s++;
if(s==_t[7]+1) break;
}

t++;
if(t==_t[8]+1) break;
}

o++;
if(o==_t[9]+1) break;
}

/* here A8 *//* add*/
/*p++;
if(p==_t[10]+1) break;
}
*/
}/** arrayreset **/


#if !PBC
void getsum(void)
{
/* here A3 */  /* add p */
int x,y,z,u,v,w,r,s,t,o;

/* here A8 *//* add*/
/*for(p=0;p<=pt;p++)
*/
for(o=0;o<=ot;o++)
for(t=0;t<=tt;t++)
for(s=0;s<=st;s++)
for(r=0;r<=rt;r++)
for(w=0;w<=wt;w++)
for(v=0;v<=vt;v++)
for(u=0;u<=ut;u++)
for(z=0;z<=zt;z++)
for(y=0;y<=yt;y++)
for(x=0;x<=xt;x++){
/* here A9 */         /* add p */
if(pfunc(0,x,y,z,u,v,w,r,s,t,o,-1)==fcolor) sum++;
}
}/** getsum **/
#endif


void field(void)
{
/* here A3 */    /* add p */
int i,x,y,z,u,v,w,r,s,t,o,/*fcolor=16,*/old,ds1[DMS_limit],ds2[DMS_limit];
int val;

fieldflag=1;
old=sn_;sn_=0;
val=SMALLER;                        /* SMALLER */

/*99*/
#if PBC
/* all */
/* here A8 *//* add*/
/*for(p=0;p<=pt;p++)
*/
for(o=0;o<=ot;o++)
for(t=0;t<=tt;t++)
for(s=0;s<=st;s++)
for(r=0;r<=rt;r++)
for(w=0;w<=wt;w++)
for(v=0;v<=vt;v++)
for(u=0;u<=ut;u++)
for(z=0;z<=zt;z++)
for(y=0;y<=yt;y++)
for(x=0;x<=xt;x++){
/* here A9 */       /* add p */
putpixel(x,y,z,u,v,w,r,s,t,o,fcolor);
if(PBC && cnt==0) sum++;
}
#else
if(1){
/* lower */                         /* full size */
for(i=0;i<DMS_limit;i++) {ds1[i]=0;ds2[i]=0;}
/* 0 */
     if(DMS==3) ds2[2]=zt;
     if(DMS<=4) ds2[3]=ut;
     if(DMS<=5) ds2[4]=vt;
     if(DMS<=6) ds2[5]=wt;
     if(DMS<=7) ds2[6]=rt;
     if(DMS<=8) ds2[7]=st;
     if(DMS<=9) ds2[8]=tt;
     if(DMS<=10) ds2[9]=ot;
/* here B6 *//* add*/
/*     if(DMS<=11) ds2[10]=pt;
*/
/* here B8 *//* add*/
```

```
/*for(p=0+ds1[10];p<=pt-ds2[10];p++)
*/
for(o=0+ds1[9];o<=ot-ds2[9];o++)
for(t=0+ds1[8];t<=tt-ds2[8];t++)
for(s=0+ds1[7];s<=st-ds2[7];s++)
for(r=0+ds1[6];r<=rt-ds2[6];r++)
for(w=0+ds1[5];w<=wt-ds2[5];w++)
for(v=0+ds1[4];v<=vt-ds2[4];v++)
for(u=0+ds1[3];u<=ut-ds2[3];u++)
for(z=0+ds1[2];z<=zt-ds2[2];z++)
for(y=0+ds1[1];y<=yt-ds2[1];y++)
for(x=0+ds1[0];x<=xt-ds2[0];x++)
/* here B9 */       /* add p */
putpixel(x,y,z,u,v,w,r,s,t,o,fcolor);

/* upper */                         /* full size */
for(i=0;i<DMS_limit;i++) {ds1[i]=0;ds2[i]=0;}
/* ?t */
     if(DMS==3) ds1[2]=zt;
     if(DMS<=4) ds1[3]=ut;
     if(DMS<=5) ds1[4]=vt;
     if(DMS<=6) ds1[5]=wt;
     if(DMS<=7) ds1[6]=rt;
     if(DMS<=8) ds1[7]=st;
     if(DMS<=9) ds1[8]=tt;
     if(DMS<=10) ds1[9]=ot;
/* here B6 *//* add*/
/*     if(DMS<=11) ds1[10]=pt;
*/
/* here B8 *//* add*/
/*for(p=0+ds1[10];p<=pt-ds2[10];p++)
*/
for(o=0+ds1[9];o<=ot-ds2[9];o++)
for(t=0+ds1[8];t<=tt-ds2[8];t++)
for(s=0+ds1[7];s<=st-ds2[7];s++)
for(r=0+ds1[6];r<=rt-ds2[6];r++)
for(w=0+ds1[5];w<=wt-ds2[5];w++)
for(v=0+ds1[4];v<=vt-ds2[4];v++)
for(u=0+ds1[3];u<=ut-ds2[3];u++)
for(z=0+ds1[2];z<=zt-ds2[2];z++)
for(y=0+ds1[1];y<=yt-ds2[1];y++)
for(x=0+ds1[0];x<=xt-ds2[0];x++)
/* here B9 */       /* add p */
putpixel(x,y,z,u,v,w,r,s,t,o,fcolor);
}/**if()**/

/* the following are smaller size if val=1 */
/* the following are full size if val=0 */

if(DMS>3){
/* xy_ */
for(i=0;i<DMS_limit;i++) if(i<DMS) {ds1[i]=val;ds2[i]=val;}else{ds1[i]=0;ds2[i]=0;}
/* 0 */
z=0;          /* z */
/* here B8 *//* add*/
```

```
/*for(p=0+ds1[10];p<=pt-ds2[10];p++)
*/
for(o=0+ds1[9];o<=ot-ds2[9];o++)
for(t=0+ds1[8];t<=tt-ds2[8];t++)
for(s=0+ds1[7];s<=st-ds2[7];s++)
for(r=0+ds1[6];r<=rt-ds2[6];r++)
for(w=0+ds1[5];w<=wt-ds2[5];w++)
for(v=0+ds1[4];v<=vt-ds2[4];v++)
for(u=0+ds1[3];u<=ut-ds2[3];u++)
for(y=0+ds1[1];y<=yt-ds2[1];y++)
for(x=0+ds1[0];x<=xt-ds2[0];x++)
/* here B9 */       /* add p */
putpixel(x,y,z,u,v,w,r,s,t,o,fcolor);

for(i=0;i<DMS_limit;i++) if(i<DMS) {ds1[i]=val;ds2[i]=val;}else{ds1[i]=0;ds2[i]=0;}
/* ?t */
z=zt;          /* z */
/* here B8 *//* add*/
/*for(p=0+ds1[10];p<=pt-ds2[10];p++)
*/
for(o=0+ds1[9];o<=ot-ds2[9];o++)
for(t=0+ds1[8];t<=tt-ds2[8];t++)
for(s=0+ds1[7];s<=st-ds2[7];s++)
for(r=0+ds1[6];r<=rt-ds2[6];r++)
for(w=0+ds1[5];w<=wt-ds2[5];w++)
for(v=0+ds1[4];v<=vt-ds2[4];v++)
for(u=0+ds1[3];u<=ut-ds2[3];u++)
for(y=0+ds1[1];y<=yt-ds2[1];y++)
for(x=0+ds1[0];x<=xt-ds2[0];x++)
/* here B9 */       /* add p */
putpixel(x,y,z,u,v,w,r,s,t,o,fcolor);
}/**if()**/

if(1){
/* xz_ */
for(i=0;i<DMS_limit;i++) if(i<DMS) {ds1[i]=val;ds2[i]=val;}else{ds1[i]=0;ds2[i]=0;}
y=0;
/* here B8 *//* add*/
/*for(p=0+ds1[10];p<=pt-ds2[10];p++)
*/
for(o=0+ds1[9];o<=ot-ds2[9];o++)
for(t=0+ds1[8];t<=tt-ds2[8];t++)
for(s=0+ds1[7];s<=st-ds2[7];s++)
for(r=0+ds1[6];r<=rt-ds2[6];r++)
for(w=0+ds1[5];w<=wt-ds2[5];w++)
for(v=0+ds1[4];v<=vt-ds2[4];v++)
for(u=0+ds1[3];u<=ut-ds2[3];u++)
for(z=0+ds1[2];z<=zt-ds2[2];z++)
for(x=0+ds1[0];x<=xt-ds2[0];x++)
/* here B9 */       /* add p */
putpixel(x,y,z,u,v,w,r,s,t,o,fcolor);

/* xz_ */
for(i=0;i<DMS_limit;i++) if(i<DMS) {ds1[i]=val;ds2[i]=val;}else{ds1[i]=0;ds2[i]=0;}
y=yt;
```

```
/* here B8 *//* add*/
/*for(p=0+ds1[10];p<=pt-ds2[10];p++)
*/
for(o=0+ds1[9];o<=ot-ds2[9];o++)
for(t=0+ds1[8];t<=tt-ds2[8];t++)
for(s=0+ds1[7];s<=st-ds2[7];s++)
for(r=0+ds1[6];r<=rt-ds2[6];r++)
for(w=0+ds1[5];w<=wt-ds2[5];w++)
for(v=0+ds1[4];v<=vt-ds2[4];v++)
for(u=0+ds1[3];u<=ut-ds2[3];u++)
for(z=0+ds1[2];z<=zt-ds2[2];z++)
for(x=0+ds1[0];x<=xt-ds2[0];x++)
/* here B9 */       /* add p */
putpixel(x,y,z,u,v,w,r,s,t,o,fcolor);
}/**if()**/

if(1){
/* yz_ */
for(i=0;i<DMS_limit;i++) if(i<DMS) {ds1[i]=val;ds2[i]=val;}else{ds1[i]=0;ds2[i]=0;}
x=0;
/* here B8 *//* add*/
/*for(p=0+ds1[10];p<=pt-ds2[10];p++)
*/
for(o=0+ds1[9];o<=ot-ds2[9];o++)
for(t=0+ds1[8];t<=tt-ds2[8];t++)
for(s=0+ds1[7];s<=st-ds2[7];s++)
for(r=0+ds1[6];r<=rt-ds2[6];r++)
for(w=0+ds1[5];w<=wt-ds2[5];w++)
for(v=0+ds1[4];v<=vt-ds2[4];v++)
for(u=0+ds1[3];u<=ut-ds2[3];u++)
for(z=0+ds1[2];z<=zt-ds2[2];z++)
for(y=0+ds1[1];y<=yt-ds2[1];y++)
/* here B9 */       /* add p */
putpixel(x,y,z,u,v,w,r,s,t,o,fcolor);

/* yz_ */
for(i=0;i<DMS_limit;i++) if(i<DMS) {ds1[i]=val;ds2[i]=val;}else{ds1[i]=0;ds2[i]=0;}
x=xt;
/* here B8 *//* add*/
/*for(p=0+ds1[10];p<=pt-ds2[10];p++)
*/
for(o=0+ds1[9];o<=ot-ds2[9];o++)
for(t=0+ds1[8];t<=tt-ds2[8];t++)
for(s=0+ds1[7];s<=st-ds2[7];s++)
for(r=0+ds1[6];r<=rt-ds2[6];r++)
for(w=0+ds1[5];w<=wt-ds2[5];w++)
for(v=0+ds1[4];v<=vt-ds2[4];v++)
for(u=0+ds1[3];u<=ut-ds2[3];u++)
for(z=0+ds1[2];z<=zt-ds2[2];z++)
for(y=0+ds1[1];y<=yt-ds2[1];y++)
/* here B9 */       /* add p */
putpixel(x,y,z,u,v,w,r,s,t,o,fcolor);
}/**if()**/
#endif
```

```
sn_=old;
fieldflag=0;
}/** field **/



/* here A4 */                                              /* add int p */
int putpixel(int x,int y,int z,int u,int v,int w,int r,int s,int t,int o,
             INT pcolor)
{
int plane,VA,VZ,V0,V1,du,dv,val;

/*if((x<0)||(x>xt)||(y<0)||(y>yt)||(z<0)||(z>zt)||(u<0)||(u>ut)||(v<0)||(v>vt)||
   (w<0)||(w>wt)||(r<0)||(r>rt)||(s<0)||(s>st)||(t<0)||(t>tt)||(o<0)||(o>ot)){
return 1;
}*/
if(!GRPH) goto end;
if(fieldflag) goto end;

if(sn_==-1){
if(u==ut && v==vt && w==0 && r==0 && s==0 && t==0 && o==0){
pixel_3d(x*PS,y*PS,z*PS,pcolor);
printf(" %lld %d x:%ld y:%ld z:%ld u:%ld v:%ld w:%ld r:%ld s:%ld t:%ld o:%ld\n",
pcount[ig],ig,x,y,z,u,v,w,r,s,t,o);
}
}
else if(sn_==1 && pcount[ig]>=978 && pcount[ig]<=988){
pixel_3d(x*PS,y*PS,z*PS,pcolor);
printf(" %lld %d x:%ld y:%ld z:%ld u:%ld v:%ld w:%ld r:%ld s:%ld t:%ld o:%ld\n",
pcount[ig],ig,x,y,z,u,v,w,r,s,t,o);
}
else{
if(DMS==5 || DMS==10){
if(pcount[ig]<978) goto end;
else if(pcount[ig]>988){
ls_image(2,"fig4_npbc.bmp",0,0,800,400);
GRPH=0;goto end;
}
}
else{
GRPH=0;goto end;
}

#if 0
if(!fieldflag && Xlib==2 && GRPH){
if(DMS==10) {VA=1;VZ=1;}
else        {VA=1;VZ=0;}            /* 5d only */

/* here B9 */               /* add p */
plane=getplane(1,x,y,z,u,v,w,r,s,t,o);

printf(" %lld %d\n",pcount[ig],ig);
/*if(plane>=2 && plane<=5) printf(" plane:2~5\n");
else printf(" plane:%d\n",plane);*/

/* PBC:1, DMS:5 */
```

```
#if CUBE==-1
V0=delta_x;
V1=/*RESO-1-delta_x*/3*delta_x+1;
du=1;
dv=1;
val=4;
/* here A10 */                                        /* add && p==0 */
     if(u==V0+du && v==V0+dv && w==0 && r==0 && s==0 && t==0 && o==0){
pixel_3d((x+0*RESO)*PS,(y+0)*PS,z*PS,ig);use_subroop();
}
/* here A10 */                                        /* add && p==0 */
else if(u==V1+du && v==V0+dv && w==0 && r==0 && s==0 && t==0 && o==0){
pixel_3d((x+0*RESO)*PS,(y+val*RESO)*PS,z*PS,ig);use_subroop();
}
/* here A10 */                                        /* add && p==0 */
else if(u==V0+du && v==V1+dv && w==0 && r==0 && s==0 && t==0 && o==0){
pixel_3d((x+val*RESO)*PS,(y+0*RESO)*PS,z*PS,ig);use_subroop();
}
/* here A10 */                                        /* add && p==0 */
else if(u==V1+du && v==V1+dv && w==0 && r==0 && s==0 && t==0 && o==0){
pixel_3d((x+val*RESO)*PS,(y+val*RESO)*PS,z*PS,ig);use_subroop();
}
/* PBC:0, DMS:5 */
#elif CUBE==0
/* xyU(xyuv) */
/* here B10 */                                         /* add && p==VA */
     if(plane==6 && v==VA && w==VA && r==VA && s==VA && t==VA && o==VA){
pixel_3d((x+2*RESO)*PS,(y+0)*PS,u*PS,ig);use_subroop();
}
/* here B10 */                                         /* add && p==VA */
else if(plane==7 && v==VA && w==VA && r==VA && s==VA && t==VA && o==VA){
pixel_3d((x+2*RESO)*PS,(y+2*RESO)*PS,u*PS,ig);use_subroop();
}
#elif CUBE==1
/* xyV(xyuv) */
/* here B10 */                                         /* add && p==VA */
     if(plane==6 && u==VA && w==VZ && r==VZ && s==VZ && t==VZ && o==VZ){
pixel_3d((x+0*RESO)*PS,(y+0)*PS,v*PS,ig);use_subroop();
}
/* here B10 */                                         /* add && p==VA */
     if(plane==7 && u==VA && w==VZ && r==VZ && s==VZ && t==VZ && o==VZ){
pixel_3d((x+0*RESO)*PS,(y+4*RESO)*PS,v*PS,ig);use_subroop();
}
#elif CUBE==2
/* xzU(xzuv) */
/* here B10 */                                         /* add && p==VA */
     if(plane==2 && v==VA && w==VZ && r==VZ && s==VZ && t==VZ && o==VZ){
pixel_3d((x+0*RESO)*PS,(z+0)*PS,u*PS,ig);use_subroop();
}
/* here B10 */                                         /* add && p==VA */
else if(plane==3 && v==VA && w==VZ && r==VZ && s==VZ && t==VZ && o==VZ){
pixel_3d((x+0*RESO)*PS,(z+4*RESO)*PS,u*PS,ig);use_subroop();
}
/* yzU(yzuv) */
/* here B10 */                                         /* add && p==VA */
```

```
else if(plane==4 && v==VA && w==VZ && r==VZ && s==VZ && t==VZ && o==VZ){
pixel_3d((y+4*RESO)*PS,(z+0)*PS,u*PS,ig);use_subroop();
}
/* here B10 */                                       /* add && p==VA */
else if(plane==5 && v==VA && w==VZ && r==VZ && s==VZ && t==VZ && o==VZ){
pixel_3d((y+4*RESO)*PS,(z+4*RESO)*PS,u*PS,ig);use_subroop();
}
#elif CUBE==3
/* xzV(xzuv) */
/* here B10 */                                       /* add && p==VA */
     if(plane==2 && u==VA && w==VA && r==VA && s==VA && t==VA && o==VA){
pixel_3d((x+2*RESO)*PS,(z+0)*PS,v*PS,ig);use_subroop();
}
/* here B10 */                                       /* add && p==VA */
else if(plane==3 && u==VA && w==VA && r==VA && s==VA && t==VA && o==VA){
pixel_3d((x+2*RESO)*PS,(z+2*RESO)*PS,v*PS,ig);use_subroop();
}
/* yzV(yzuv) */
/* here B10 */                                       /* add && p==VA */
else if(plane==4 && u==VA && w==VA && r==VA && s==VA && t==VA && o==VA){
pixel_3d((y+4*RESO)*PS,(z+0)*PS,v*PS,ig);use_subroop();
}
/* here B10 */                                       /* add && p==VA */
else if(plane==5 && u==VA && w==VA && r==VA && s==VA && t==VA && o==VA){
pixel_3d((y+4*RESO)*PS,(z+2*RESO)*PS,v*PS,ig);use_subroop();
}
#endif

}/**if(,,,)**/
#endif

}

end:
/* here A9 */      /* add p */
pfunc(1,x,y,z,u,v,w,r,s,t,o,pcolor);

return 0;
}/** putpixel **/



/* here A4 */                                                /* add int p */
INT getpixel(int x,int y,int z,int u,int v,int w,int r,int s,int t,int o)
{
/* here A7 */                /* add P=p; */
X=x;Y=y;Z=z;U=u;V=v;W=w;R=r;S=s;T=t;O=o;
/* here A7 */                                  /* add P_=Np; */
X_=Nx;Y_=Ny;Z_=Nz;U_=Nu;V_=Nv;W_=Nw;R_=Nr;S_=Ns;T_=Nt;O_=No;

/* here A10 */                                      /* add ||(p<0)||(p>pt) */
if((x<0)||(x>xt)||(y<0)||(y>yt)||(z<0)||(z>zt)||(u<0)||(u>ut)||(v<0)||(v>vt)||
   (w<0)||(w>wt)||(r<0)||(r>rt)||(s<0)||(s>st)||(t<0)||(t>tt)||(o<0)||(o>ot)){
return wall;
}
```

```
/* here A9 */             /* add p */
return pfunc(0,x,y,z,u,v,w,r,s,t,o,-1);
}/** getpixel **/


int jump(int i)
{
#if PBC
if(nx[i]==-1)        {nx[i]=RESO-1;nx_[i]=RESO;}
else if(nx[i]==RESO) {nx[i]=0;nx_[i]=-1;}
if(ny[i]==-1)        {ny[i]=RESO-1;ny_[i]=RESO;}
else if(ny[i]==RESO) {ny[i]=0;ny_[i]=-1;}
if(nz[i]==-1)        {nz[i]=RESO-1;nz_[i]=RESO;}
else if(nz[i]==RESO) {nz[i]=0;nz_[i]=-1;}
if(nu[i]==-1)        {nu[i]=RESO-1;nu_[i]=RESO;}
else if(nu[i]==RESO) {nu[i]=0;nu_[i]=-1;}
if(nv[i]==-1)        {nv[i]=RESO-1;nv_[i]=RESO;}
else if(nv[i]==RESO) {nv[i]=0;nv_[i]=-1;}
if(nw[i]==-1)        {nw[i]=RESO-1;nw_[i]=RESO;}
else if(nw[i]==RESO) {nw[i]=0;nw_[i]=-1;}
if(nr[i]==-1)        {nr[i]=RESO-1;nr_[i]=RESO;}
else if(nr[i]==RESO) {nr[i]=0;nr_[i]=-1;}
if(ns[i]==-1)        {ns[i]=RESO-1;ns_[i]=RESO;}
else if(ns[i]==RESO) {ns[i]=0;ns_[i]=-1;}
if(nt[i]==-1)        {nt[i]=RESO-1;nt_[i]=RESO;}
else if(nt[i]==RESO) {nt[i]=0;nt_[i]=-1;}
if(no[i]==-1)        {no[i]=RESO-1;no_[i]=RESO;}
else if(no[i]==RESO) {no[i]=0;no_[i]=-1;}
/* here A6 *//* add*/
/*if(np[i]==-1)        {np[i]=RESO-1;np_[i]=RESO;}
else if(np[i]==RESO) {np[i]=0;np_[i]=-1;}
*/
#endif
}/** jump **/


int connect_nxpm(int nxpm)
{
#if PBC
if(nxpm==-1)        return RESO-1;
else if(nxpm==RESO) return 0;
else                return nxpm;
#else
return nxpm;
#endif
}/** connect_nxpm **/


int connect_nypm(int nypm)
{
#if PBC
if(nypm==-1)        return RESO-1;
else if(nypm==RESO) return 0;
else                return nypm;
#else
```

```
return nypm;
#endif
}/** connect_nypm **/


int connect_nzpm(int nzpm)
{
#if PBC
if(nzpm==-1)        return RESO-1;
else if(nzpm==RESO) return 0;
else                return nzpm;
#else
return nzpm;
#endif
}/** connect_nzpm **/


int connect_nupm(int nupm)
{
#if PBC
if(nupm==-1)        return RESO-1;
else if(nupm==RESO) return 0;
else                return nupm;
#else
return nupm;
#endif
}/** connect_nupm **/


int connect_nvpm(int nvpm)
{
#if PBC
if(nvpm==-1)        return RESO-1;
else if(nvpm==RESO) return 0;
else                return nvpm;
#else
return nvpm;
#endif
}/** connect_nvpm **/


int connect_nwpm(int nwpm)
{
#if PBC
if(nwpm==-1)        return RESO-1;
else if(nwpm==RESO) return 0;
else                return nwpm;
#else
return nwpm;
#endif
}/** connect_nwpm **/


int connect_nrpm(int nrpm)
{
```

```
#if PBC
if(nrpm==-1)        return RESO-1;
else if(nrpm==RESO) return 0;
else                return nrpm;
#else
return nrpm;
#endif
}/** connect_nrpm **/


int connect_nspm(int nspm)
{
#if PBC
if(nspm==-1)        return RESO-1;
else if(nspm==RESO) return 0;
else                return nspm;
#else
return nspm;
#endif
}/** connect_nspm **/


int connect_ntpm(int ntpm)
{
#if PBC
if(ntpm==-1)        return RESO-1;
else if(ntpm==RESO) return 0;
else                return ntpm;
#else
return ntpm;
#endif
}/** connect_ntpm **/


int connect_nopm(int nopm)
{
#if PBC
if(nopm==-1)        return RESO-1;
else if(nopm==RESO) return 0;
else                return nopm;
#else
return nopm;
#endif
}/** connect_nopm **/


/* here A11 *//* add*/
/*int connect_nppm(int nppm)
{
#if PBC
if(nppm==-1)        return RESO-1;
else if(nppm==RESO) return 0;
else                return nppm;
#else
return nppm;
```

```
#endif
}*//** connect_nppm **/



int random_(int n)
{
int val;

val=(int)((rand()/(RAND_MAX+1.))*n);

return val;
}/** random_ **/


int fen(char *str,int i,int jmax)
{
int val;

if(i==jmax+1) val=0;
else if(i==-1) val=jmax;
else val=i;

if(strcmp(str,"X")==0) return enX[val];
else if(strcmp(str,"Y")==0) return enY[val];
else if(strcmp(str,"Z")==0) return enZ[val];
else if(strcmp(str,"U")==0) return enU[val];
else if(strcmp(str,"V")==0) return enV[val];
else if(strcmp(str,"W")==0) return enW[val];
else if(strcmp(str,"R")==0) return enR[val];
else if(strcmp(str,"S")==0) return enS[val];
else if(strcmp(str,"T")==0) return enT[val];
else if(strcmp(str,"O")==0) return enO[val];
/* here A6 *//* add*/
/*else if(strcmp(str,"P")==0) return enP[val];
*/

else if(strcmp(str,"X_")==0) return enX_[val];
else if(strcmp(str,"Y_")==0) return enY_[val];
else if(strcmp(str,"Z_")==0) return enZ_[val];
else if(strcmp(str,"U_")==0) return enU_[val];
else if(strcmp(str,"V_")==0) return enV_[val];
else if(strcmp(str,"W_")==0) return enW_[val];
else if(strcmp(str,"R_")==0) return enR_[val];
else if(strcmp(str,"S_")==0) return enS_[val];
else if(strcmp(str,"T_")==0) return enT_[val];
else if(strcmp(str,"O_")==0) return enO_[val];
/* here A6 *//* add*/
/*else if(strcmp(str,"P_")==0) return enP_[val];
*/
}/** fen **/


/* here A4 */                                                   /* add int p */
INT rpixel(int x,int y,int z,int u,int v,int w,int r,int s,int t,int o){
```

```
/* here A10 */                                        /* add ||(p<0)||(p>pt) */
if((x<0)||(x>xt)||(y<0)||(y>yt)||(z<0)||(z>zt)||(u<0)||(u>ut)||(v<0)||(v>vt)||
   (w<0)||(w>wt)||(r<0)||(r>rt)||(s<0)||(s>st)||(t<0)||(t>tt)||(o<0)||(o>ot)){
return wall;
}

/* here A9 */             /* add p */
return pfunc(0,x,y,z,u,v,w,r,s,t,o,-1);
}/** rpixel **/


void set_colors(void)
{
int i;

for(i=0;i<CPMAX;i++){
if(PAL) acolor[i]=wall;
else{
/*if(i<16) */acolor[i]=i;
/*else     acolor[i]=i+1;*/
}
}
}/** set_colors **/


#if PBC
void set_seedpoints(void)
{
/* here A3 *//* add q, mp */
int i,j,k,l,m,n,o,p,mx,my,mz,mu,mv,mw,mr,ms,mt,mo;
/* here A3 */                                   /* add base_p */
int xyz,bases,base_u,base_v,base_w,base_r,base_s,base_t,base_o;
int iend=DMS_limit-3;
int num[DMS_limit],coor[DMS_limit],db[DMS_limit];

i=0;
while(1){
db[i]=0;

i++;if(i==iend) break;
}

xyz=pow(cpwidth,3);
db[0]=xyz;                           /* 2*2*2 */
i=0;
while(1){
if(db[i]==CPMAX/cpwidth) break;
if(db[i]>CPMAX/cpwidth) {db[i]=0;break;}

i++;if(i==iend) break;
db[i]=db[i-1]*cpwidth;
}

i=0;
while(1){
```

```
if(db[i]>0) {num[i]=cpwidth-1;coor[i]=1;}
else        {num[i]=0;coor[i]=0;}

i++;if(i==iend) break;
}

/* here A8 *//* add*/
/*base_p=0;
for(q=0;q<=num[7];q++){
mp=q%cpwidth;
*/

base_o=0;
for(p=0;p<=num[6];p++){
mo=p%cpwidth;

base_t=0;
for(o=0;o<=num[5];o++){
mt=o%cpwidth;

base_s=0;
for(n=0;n<=num[4];n++){
ms=n%cpwidth;

base_r=0;
for(m=0;m<=num[3];m++){
mr=m%cpwidth;

base_w=0;
for(l=0;l<=num[2];l++){
mw=l%cpwidth;

base_v=0;
for(k=0;k<=num[1];k++){
mv=k%cpwidth;

base_u=0;
for(j=0;j<=num[0];j++){
mu=j%cpwidth;

for(i=0;i<xyz;i++){
my=i%cpwidth;
mx=(i%(cpwidth*cpwidth))/cpwidth;   /* 2, 4 or 16, 4 */
mz=i/(cpwidth*cpwidth);
/* here A7 *//* add base_p+ */
bases=base_o+base_t+base_s+base_r+base_w+base_v+base_u;
nax[bases+i]=delta_x+mx*(2*delta_x+1);
nay[bases+i]=delta_x+my*(2*delta_x+1);
naz[bases+i]=delta_x+mz*(2*delta_x+1);
nau[bases+i]=(delta_x+mu*(2*delta_x+1))*coor[0];
nav[bases+i]=(delta_x+mv*(2*delta_x+1))*coor[1];
naw[bases+i]=(delta_x+mw*(2*delta_x+1))*coor[2];
nar[bases+i]=(delta_x+mr*(2*delta_x+1))*coor[3];
nas[bases+i]=(delta_x+ms*(2*delta_x+1))*coor[4];
nat[bases+i]=(delta_x+mt*(2*delta_x+1))*coor[5];
nao[bases+i]=(delta_x+mo*(2*delta_x+1))*coor[6];
/* here A7 *//* add*/
/*nap[bases+i]=(delta_x+mp*(2*delta_x+1))*coor[7];
*/
}/**for(i)**/

base_u+=db[0];
}/**for(j)**/

base_v+=db[1];
}/**for(k)**/

base_w+=db[2];
}/**for(l)**/

base_r+=db[3];
}/**for(m)**/

base_s+=db[4];
}/**for(n)**/

base_t+=db[5];
}/**for(o)**/

base_o+=db[6];
}/**for(p)**/

/* here A8 *//* add*/
/*base_p+=db[7];
}*//**for(q)**/
}/** set_seedpoints **/
#elif _1st__==0 || DMS==3
void set_seedpoints(void)
{
/* here B3 */               /* add mp */
int i,j,mx,my,mz,mu,mv,mw,mr,ms,mt,mo,ds2;

if(CPMAX==2){
for(i=0;i<4;i++){
/* lower,upper */
my=i%cpwidth;
mx=(i%(cpwidth*cpwidth))/cpwidth;
mz=/*RESO-1-1*/0;
mu=0;
mv=0;
mw=0;
mr=0;
ms=0;
mt=0;
mo=0;
/* here B7 *//* add*/
/*mp=0;
*/

/* here B7 *//* add*/
```

```
/*nap[i]=mp;
*/
nax[i]=delta_x+mx*(2*delta_x+1);nay[i]=delta_x+my*(2*delta_x+1);naz[i]=mz;nau[i]=mu;
nav[i]=mv;naw[i]=mw;nar[i]=mr;nas[i]=ms;nat[i]=mt;nao[i]=mo;
}

i=0;j=0;
/* here B7 *//* add*/
/*nap[i]=nap[j];
*/
nax[i]=nax[j];nay[i]=nay[j];naz[i]=naz[j];nau[i]=nau[j];nav[i]=nav[j];naw[i]=naw[j];
nar[i]=nar[j];nas[i]=nas[j];nat[i]=nat[j];nao[i]=nao[j];
i=1;j=3;
/* here B7 *//* add*/
/*nap[i]=nap[j];
*/
nax[i]=nax[j];nay[i]=nay[j];naz[i]=naz[j];nau[i]=nau[j];nav[i]=nav[j];naw[i]=naw[j];
nar[i]=nar[j];nas[i]=nas[j];nat[i]=nat[j];nao[i]=nao[j];
}
else if(CPMAX==4){
for(i=0;i<4;i++){
/* lower */
my=i%cpwidth;
mx=(i%(cpwidth*cpwidth))/cpwidth;
mz=/*RESO-1-1*/0;
mu=0;
mv=0;
mw=0;
mr=0;
ms=0;
mt=0;
mo=0;
/* here B7 *//* add*/
/*mp=0;
*/

/* here B7 *//* add*/
/*nap[i]=mp;
*/
nax[i]=delta_x+mx*(2*delta_x+1);nay[i]=delta_x+my*(2*delta_x+1);naz[i]=mz;nau[i]=mu;
nav[i]=mv;naw[i]=mw;nar[i]=mr;nas[i]=ms;nat[i]=mt;nao[i]=mo;
}

for(i=0;i<4;i++){
/* here B7 *//* add*/
/*p[i]=nap[i];
*/
x[i]=nax[i];y[i]=nay[i];z[i]=naz[i];u[i]=nau[i];v[i]=nav[i];w[i]=naw[i];r[i]=nar[i];
s[i]=nas[i];t[i]=nat[i];o[i]=nao[i];
}

i=0;j=0;
/* here B7 *//* add*/
/*nap[i]=p[j];
*/
```

```
nax[i]=x[j];nay[i]=y[j];naz[i]=z[j];nau[i]=u[i];nav[i]=v[j];naw[i]=w[j];nar[i]=r[j];
nas[i]=s[j];nat[i]=t[j];nao[i]=o[j];
i=1;j=3;
/* here B7 *//* add*/
/*nap[i]=p[j];
*/
nax[i]=x[j];nay[i]=y[j];naz[i]=z[j];nau[i]=u[i];nav[i]=v[j];naw[i]=w[j];nar[i]=r[j];
nas[i]=s[j];nat[i]=t[j];nao[i]=o[j];
if(0){
/* lower, upper */
i=2;j=1;
nax[i]=x[j];nay[i]=y[j];naz[i]=RESO-1;nau[i]=RESO-1;
i=3;j=2;
nax[i]=x[j];nay[i]=y[j];naz[i]=RESO-1;nau[i]=RESO-1;
}
else{
/* lower */
i=2;j=1;
/* here B7 *//* add*/
/*nap[i]=p[j];
*/
nax[i]=x[j];nay[i]=y[j];naz[i]=z[j];nau[i]=u[j];nav[i]=v[j];naw[i]=w[j];nar[i]=r[j];
nas[i]=s[j];nat[i]=t[j];nao[i]=o[j];
i=3;j=2;
/* here B7 *//* add*/
/*nap[i]=p[j];
*/
nax[i]=x[j];nay[i]=y[j];naz[i]=z[j];nau[i]=u[j];nav[i]=v[j];naw[i]=w[j];nar[i]=r[j];
nas[i]=s[j];nat[i]=t[j];nao[i]=o[j];
}
}
else if(CPMAX==8){
ds2=0;

for(i=0;i<8/2;i++){
my=i%cpwidth;
mx=(i%(cpwidth*cpwidth))/cpwidth;
mz=0+ds2;
mu=0;
mv=0;
mw=0;
mr=0;
ms=0;
mt=0;
mo=0;
/* here B7 *//* add*/
/*mp=0;
*/

/* here B7 *//* add*/
/*nap[i]=mp;
*/
nax[i]=delta_x+mx*(2*delta_x+1);nay[i]=delta_x+my*(2*delta_x+1);naz[i]=mz;nau[i]=mu;
nav[i]=mv;naw[i]=mw;nar[i]=mr;nas[i]=ms;nat[i]=mt;nao[i]=mo;
}
```

```
for(i=0;i<8/2;i++){
/* here B7 *//* add*/
/*p[i]=nap[i];
*/
x[i]=nax[i];y[i]=nay[i];z[i]=naz[i];u[i]=nau[i];v[i]=nav[i];w[i]=naw[i];r[i]=nar[i];
s[i]=nas[i];t[i]=nat[i];o[i]=nao[i];
}

/*mz=0+ds2;mv=0;*/

i=0;j=0;
/* here B7 *//* add*/
/*nap[i]=p[j];
*/
nax[i]=x[j];nay[i]=y[j];naz[i]=z[j];nau[i]=u[i];nav[i]=v[j];naw[i]=w[j];nar[i]=r[j];
nas[i]=s[j];nat[i]=t[j];nao[i]=o[j];
i=1;j=3;
/* here B7 *//* add*/
/*nap[i]=p[j];
*/
nax[i]=x[j];nay[i]=y[j];naz[i]=z[j];nau[i]=u[i];nav[i]=v[j];naw[i]=w[j];nar[i]=r[j];
nas[i]=s[j];nat[i]=t[j];nao[i]=o[j];
i=2;j=1;
/* here B7 *//* add*/
/*nap[i]=p[j];
*/
nax[i]=x[j];nay[i]=y[j];naz[i]=z[j];nau[i]=u[i];nav[i]=v[j];naw[i]=w[j];nar[i]=r[j];
nas[i]=s[j];nat[i]=t[j];nao[i]=o[j];
i=3;j=2;
/* here B7 *//* add*/
/*nap[i]=p[j];
*/
nax[i]=x[j];nay[i]=y[j];naz[i]=z[j];nau[i]=u[i];nav[i]=v[j];naw[i]=w[j];nar[i]=r[j];
nas[i]=s[j];nat[i]=t[j];nao[i]=o[j];

if(1) mz=RESO-1-ds2;
if(0) mu=RESO-1;
if(0) mv=RESO-1;
if(0) mw=RESO-1;
if(0) mr=RESO-1;
if(0) ms=RESO-1;
if(0) mt=RESO-1;
if(0) mo=RESO-1;

i=4;j=0;
nax[i]=x[j];nay[i]=y[j];naz[i]=mz;nau[i]=mu;nav[i]=mv;naw[i]=mw;nar[i]=mr;nas[i]=ms;
/* here B7 *//* add*/
/*nap[i]=mp;
*/
nat[i]=mt;nao[i]=mo;
i=5;j=3;
/* here B7 *//* add*/
/*nap[i]=mp;
*/
```

```
nax[i]=x[j];nay[i]=y[j];naz[i]=mz;nau[i]=mu;nav[i]=mv;naw[i]=mw;nar[i]=mr;nas[i]=ms;
nat[i]=mt;nao[i]=mo;
i=6;j=1;
/* here B7 *//* add*/
/*nap[i]=mp;
*/
nax[i]=x[j];nay[i]=y[j];naz[i]=mz;nau[i]=mu;nav[i]=mv;naw[i]=mw;nar[i]=mr;nas[i]=ms;
nat[i]=mt;nao[i]=mo;
i=7;j=2;
/* here B7 *//* add*/
/*nap[i]=mp;
*/
nax[i]=x[j];nay[i]=y[j];naz[i]=mz;nau[i]=mu;nav[i]=mv;naw[i]=mw;nar[i]=mr;nas[i]=ms;
nat[i]=mt;nao[i]=mo;
}
}/** set_seedpoints **/
#else
void set_seedpoints(void)
{
int i,val[DMS_max-1];

/* smaller version */
val[0]=2;
val[1]=1;                             /* z */
if(DMS>=4)  val[2]=1;else val[2]=0;  /* u */
if(DMS>=5)  val[3]=1;else val[3]=0;  /* v */
if(DMS>=6)  val[4]=1;else val[4]=0;  /* w */
if(DMS>=7)  val[5]=1;else val[5]=0;  /* r */
if(DMS>=8)  val[6]=1;else val[6]=0;  /* s */
if(DMS>=9)  val[7]=1;else val[7]=0;  /* t */
if(DMS>=10) val[8]=1;else val[8]=0;  /* o */
/* here B6 *//* add*/
/*if(DMS>=11) val[9]=1;else val[9]=0;*/  /* p */

i=0;nax[i]=xt-val[0];nay[i]=0;naz[i]=0+val[1];nau[i]=0+val[2];nav[i]=0+val[3];
    naw[i]=0+val[4];nar[i]=0+val[5];nas[i]=0+val[6];nat[i]=0+val[7];nao[i]=0+val[8];
/* here B7 *//* add nap[i]=0+val[9]; */
i=1;nax[i]=0+val[0];nay[i]=yt;naz[i]=0+val[1];nau[i]=0+val[2];nav[i]=0+val[3];
    naw[i]=0+val[4];nar[i]=0+val[5];nas[i]=0+val[6];nat[i]=0+val[7];nao[i]=0+val[8];
/* here B7 *//* add nap[i]=0+val[9]; */
if(CPMAX>=4){
i=2;nax[i]=0;nay[i]=0+val[0];naz[i]=0+val[1];nau[i]=0+val[2];nav[i]=0+val[3];
    naw[i]=0+val[4];nar[i]=0+val[5];nas[i]=0+val[6];nat[i]=0+val[7];nao[i]=0+val[8];
/* here B7 *//* add nap[i]=0+val[9]; */
i=3;nax[i]=xt;nay[i]=yt-val[0];naz[i]=0+val[1];nau[i]=0+val[2];nav[i]=0+val[3];
    naw[i]=0+val[4];nar[i]=0+val[5];nas[i]=0+val[6];nat[i]=0+val[7];nao[i]=0+val[8];
/* here B7 *//* add nap[i]=0+val[9]; */

i=4;nax[i]=0;nay[i]=yt-val[0];naz[i]=zt-val[1];nau[i]=0+val[2];nav[i]=0+val[3];
    naw[i]=0+val[4];nar[i]=0+val[5];nas[i]=0+val[6];nat[i]=0+val[7];nao[i]=0+val[8];
/* here B7 *//* add nap[i]=0+val[9]; */
i=5;nax[i]=xt;nay[i]=0+val[0];naz[i]=zt-val[1];nau[i]=0+val[2];nav[i]=0+val[3];
    naw[i]=0+val[4];nar[i]=0+val[5];nas[i]=0+val[6];nat[i]=0+val[7];nao[i]=0+val[8];
/* here B7 *//* add nap[i]=0+val[9]; */
i=6;nax[i]=xt-val[0];nay[i]=yt;naz[i]=zt-val[1];nau[i]=0+val[2];nav[i]=0+val[3];
```

```
    naw[i]=0+val[4];nar[i]=0+val[5];nas[i]=0+val[6];nat[i]=0+val[7];nao[i]=0+val[8];
/* here B7 *//* add nap[i]=0+val[9]; */
i=7;nax[i]=0+val[0];nay[i]=0;naz[i]=zt-val[1];nau[i]=0+val[2];nav[i]=0+val[3];
    naw[i]=0+val[4];nar[i]=0+val[5];nas[i]=0+val[6];nat[i]=0+val[7];nao[i]=0+val[8];
/* here B7 *//* add nap[i]=0+val[9]; */
}
}/** set_seedpoints **/
#endif


int CW(int pos,int jmax)
{
int j,count,val[2];

count=0;
for(j=pos;;){
/* here A9 *//* add fen("P",j,jmax) */
val[0]=rpixel(fen("X",j,jmax),
              fen("Y",j,jmax),
              fen("Z",j,jmax),
              fen("U",j,jmax),
              fen("V",j,jmax),
              fen("W",j,jmax),
              fen("R",j,jmax),
              fen("S",j,jmax),
              fen("T",j,jmax),
              fen("O",j,jmax)
             );

/* here A9 *//* add fen("P",j-1,jmax) */
val[1]=rpixel(fen("X",j-1,jmax),
              fen("Y",j-1,jmax),
              fen("Z",j-1,jmax),
              fen("U",j-1,jmax),
              fen("V",j-1,jmax),
              fen("W",j-1,jmax),
              fen("R",j-1,jmax),
              fen("S",j-1,jmax),
              fen("T",j-1,jmax),
              fen("O",j-1,jmax)
             );

if(val[0]!=ca && val[1]==ca){
/* here A7 *//* add P=fen("P",j-1,jmax); */
X=fen("X",j-1,jmax);
Y=fen("Y",j-1,jmax);
Z=fen("Z",j-1,jmax);
U=fen("U",j-1,jmax);
V=fen("V",j-1,jmax);
W=fen("W",j-1,jmax);
R=fen("R",j-1,jmax);
S=fen("S",j-1,jmax);
T=fen("T",j-1,jmax);
O=fen("O",j-1,jmax);
```

```
/* here A7 *//* add P_=fen("P_",j-1,jmax); */
X_=fen("X_",j-1,jmax);
Y_=fen("Y_",j-1,jmax);
Z_=fen("Z_",j-1,jmax);
U_=fen("U_",j-1,jmax);
V_=fen("V_",j-1,jmax);
W_=fen("W_",j-1,jmax);
R_=fen("R_",j-1,jmax);
S_=fen("S_",j-1,jmax);
T_=fen("T_",j-1,jmax);
O_=fen("O_",j-1,jmax);

return 0;
}

j--;if(j<0) j=jmax;
count++;
if(count==jmax+1) {printf(" ?CW\n");return 1;}
}/**for()**/
}/** CW **/



int CCW(int pos,int jmax)
{
int j,count,val[2];

count=0;
for(j=pos;;){
/* here A9 *//* add fen("P",j,jmax) */
val[0]=rpixel(fen("X",j,jmax),
              fen("Y",j,jmax),
              fen("Z",j,jmax),
              fen("U",j,jmax),
              fen("V",j,jmax),
              fen("W",j,jmax),
              fen("R",j,jmax),
              fen("S",j,jmax),
              fen("T",j,jmax),
              fen("O",j,jmax)
             );

/* here A9 *//* add fen("P",j+1,jmax) */
val[1]=rpixel(fen("X",j+1,jmax),
              fen("Y",j+1,jmax),
              fen("Z",j+1,jmax),
              fen("U",j+1,jmax),
              fen("V",j+1,jmax),
              fen("W",j+1,jmax),
              fen("R",j+1,jmax),
              fen("S",j+1,jmax),
              fen("T",j+1,jmax),
              fen("O",j+1,jmax)
             );

if(val[0]!=ca && val[1]==ca){
```

```
/* here A7 *//* add P=fen("P",j+1,jmax); */
X=fen("X",j+1,jmax);
Y=fen("Y",j+1,jmax);
Z=fen("Z",j+1,jmax);
U=fen("U",j+1,jmax);
V=fen("V",j+1,jmax);
W=fen("W",j+1,jmax);
R=fen("R",j+1,jmax);
S=fen("S",j+1,jmax);
T=fen("T",j+1,jmax);
O=fen("O",j+1,jmax);

/* here A7 *//* add P_=fen("P_",j+1,jmax); */
X_=fen("X_",j+1,jmax);
Y_=fen("Y_",j+1,jmax);
Z_=fen("Z_",j+1,jmax);
U_=fen("U_",j+1,jmax);
V_=fen("V_",j+1,jmax);
W_=fen("W_",j+1,jmax);
R_=fen("R_",j+1,jmax);
S_=fen("S_",j+1,jmax);
T_=fen("T_",j+1,jmax);
O_=fen("O_",j+1,jmax);

return 0;
}

j++;if(j>jmax) j=0;
count++;
if(count==jmax+1) {printf(" ?CCW\n");return 1;}
}/**for()**/
}/** CCW **/


void stt_en(int j,int i)
{
/* here A7 */                  /* add enP[j]=p[i] */
  enX[j]=x[i];enY[j]=y[i];enZ[j]=z[i];enU[j]=u[i];enV[j]=v[i];enW[j]=w[i];
  enR[j]=r[i];enS[j]=s[i];enT[j]=t[i];enO[j]=o[i];
/* here A7 */                        /* add enP_[j]=p_[i] */
  enX_[j]=x_[i];enY_[j]=y_[i];enZ_[j]=z_[i];enU_[j]=u_[i];enV_[j]=v_[i];enW_[j]=w_[i];
  enR_[j]=r_[i];enS_[j]=s_[i];enT_[j]=t_[i];enO_[j]=o_[i];
}/** stt_en **/


int en_xy(void)
{
int i,j;

j=-1;

j++;i=1;stt_en(j,i);
j++;i=3;stt_en(j,i);
j++;i=2;stt_en(j,i);
j++;i=4;stt_en(j,i);
```

```
return j;
}/** en_xy **/


int en_yz(void)
{
int i,j;

j=-1;

/* 2 => 3 */
/* 3 => 2 */
j++;i=/*2*/3;stt_en(j,i);
j++;i=5;stt_en(j,i);
j++;i=4;stt_en(j,i);
j++;i=6;stt_en(j,i);

return j;
}/** en_yz **/


int en_zx(void)
{
int i,j;

j=-1;

j++;i=5;stt_en(j,i);
j++;i=1;stt_en(j,i);
j++;i=6;stt_en(j,i);
j++;i=/*3*/2;stt_en(j,i);

return j;
}/** en_zx **/


int en_xu(void)
{
int i,j;

j=-1;

j++;i=1;stt_en(j,i);
j++;i=7;stt_en(j,i);
j++;i=/*3*/2;stt_en(j,i);
j++;i=8;stt_en(j,i);

return j;
}/** en_xu **/


int en_yu(void)
{
int i,j;
```

```
j=-1;

j++;i=/*2*/3;stt_en(j,i);
j++;i=7;stt_en(j,i);
j++;i=4;stt_en(j,i);
j++;i=8;stt_en(j,i);

return j;
}/** en_yu **/


int en_zu(void)
{
int i,j;

j=-1;

j++;i=5;stt_en(j,i);
j++;i=7;stt_en(j,i);
j++;i=6;stt_en(j,i);
j++;i=8;stt_en(j,i);

return j;
}/** en_zu **/


int en_xv(void)
{
int i,j;

j=-1;

j++;i=1;stt_en(j,i);
j++;i=9;stt_en(j,i);
j++;i=/*3*/2;stt_en(j,i);
j++;i=10;stt_en(j,i);

return j;
}/** en_xv **/


int en_yv(void)
{
int i,j;

j=-1;

j++;i=/*2*/3;stt_en(j,i);
j++;i=9;stt_en(j,i);
j++;i=4;stt_en(j,i);
j++;i=10;stt_en(j,i);

return j;
}/** en_yv **/
```

```
int en_zv(void)
{
int i,j;

j=-1;

j++;i=5;stt_en(j,i);
j++;i=9;stt_en(j,i);
j++;i=6;stt_en(j,i);
j++;i=10;stt_en(j,i);

return j;
}/** en_zv **/


int en_uv(void)
{
int i,j;

j=-1;

j++;i=7;stt_en(j,i);
j++;i=9;stt_en(j,i);
j++;i=8;stt_en(j,i);
j++;i=10;stt_en(j,i);

return j;
}/** en_uv **/


int en_xw(void)
{
int i,j;

j=-1;

j++;i=1;stt_en(j,i);
j++;i=11;stt_en(j,i);
j++;i=/*3*/2;stt_en(j,i);
j++;i=12;stt_en(j,i);

return j;
}/** en_xw **/


int en_yw(void)
{
int i,j;

j=-1;

j++;i=/*2*/3;stt_en(j,i);
j++;i=11;stt_en(j,i);
```

```
j++;i=4;stt_en(j,i);
j++;i=12;stt_en(j,i);

return j;
}/** en_yw **/


int en_zw(void)
{
int i,j;

j=-1;

j++;i=5;stt_en(j,i);
j++;i=11;stt_en(j,i);
j++;i=6;stt_en(j,i);
j++;i=12;stt_en(j,i);

return j;
}/** en_zw **/


int en_uw(void)
{
int i,j;

j=-1;

j++;i=7;stt_en(j,i);
j++;i=11;stt_en(j,i);
j++;i=8;stt_en(j,i);
j++;i=12;stt_en(j,i);

return j;
}/** en_uw **/


int en_vw(void)
{
int i,j;

j=-1;

j++;i=9;stt_en(j,i);
j++;i=11;stt_en(j,i);
j++;i=10;stt_en(j,i);
j++;i=12;stt_en(j,i);

return j;
}/** en_vw **/


int en_xr(void)
{
int i,j;
```

```
j=-1;

j++;i=1;stt_en(j,i);
j++;i=13;stt_en(j,i);
j++;i=/*3*/2;stt_en(j,i);
j++;i=14;stt_en(j,i);

return j;
}/** en_xr **/


int en_yr(void)
{
int i,j;

j=-1;

j++;i=/*2*/3;stt_en(j,i);
j++;i=13;stt_en(j,i);
j++;i=4;stt_en(j,i);
j++;i=14;stt_en(j,i);

return j;
}/** en_yr **/


int en_zr(void)
{
int i,j;

j=-1;

j++;i=5;stt_en(j,i);
j++;i=13;stt_en(j,i);
j++;i=6;stt_en(j,i);
j++;i=14;stt_en(j,i);

return j;
}/** en_zr **/


int en_ur(void)
{
int i,j;

j=-1;

j++;i=7;stt_en(j,i);
j++;i=13;stt_en(j,i);
j++;i=8;stt_en(j,i);
j++;i=14;stt_en(j,i);

return j;
}/** en_ur **/
```

```
int en_vr(void)
{
int i,j;

j=-1;

j++;i=9;stt_en(j,i);
j++;i=13;stt_en(j,i);
j++;i=10;stt_en(j,i);
j++;i=14;stt_en(j,i);

return j;
}/** en_vr **/


int en_wr(void)
{
int i,j;

j=-1;

j++;i=11;stt_en(j,i);
j++;i=13;stt_en(j,i);
j++;i=12;stt_en(j,i);
j++;i=14;stt_en(j,i);

return j;
}/** en_wr **/


int en_xs(void)
{
int i,j;

j=-1;

j++;i=1;stt_en(j,i);
j++;i=15;stt_en(j,i);
j++;i=/*3*/2;stt_en(j,i);
j++;i=16;stt_en(j,i);

return j;
}/** en_xs **/


int en_ys(void)
{
int i,j;

j=-1;

j++;i=/*2*/3;stt_en(j,i);
j++;i=15;stt_en(j,i);
```

```
j++;i=4;stt_en(j,i);
j++;i=16;stt_en(j,i);

return j;
}/** en_ys **/


int en_zs(void)
{
int i,j;

j=-1;

j++;i=5;stt_en(j,i);
j++;i=15;stt_en(j,i);
j++;i=6;stt_en(j,i);
j++;i=16;stt_en(j,i);

return j;
}/** en_zs **/


int en_us(void)
{
int i,j;

j=-1;

j++;i=7;stt_en(j,i);
j++;i=15;stt_en(j,i);
j++;i=8;stt_en(j,i);
j++;i=16;stt_en(j,i);

return j;
}/** en_us **/


int en_vs(void)
{
int i,j;

j=-1;

j++;i=9;stt_en(j,i);
j++;i=15;stt_en(j,i);
j++;i=10;stt_en(j,i);
j++;i=16;stt_en(j,i);

return j;
}/** en_vs **/


int en_ws(void)
{
int i,j;
```

```
j=-1;

j++;i=11;stt_en(j,i);
j++;i=15;stt_en(j,i);
j++;i=12;stt_en(j,i);
j++;i=16;stt_en(j,i);

return j;
}/** en_ws **/


int en_rs(void)
{
int i,j;

j=-1;

j++;i=13;stt_en(j,i);
j++;i=15;stt_en(j,i);
j++;i=14;stt_en(j,i);
j++;i=16;stt_en(j,i);

return j;
}/** en_rs **/


int en_xt(void)
{
int i,j;

j=-1;

j++;i=1;stt_en(j,i);
j++;i=17;stt_en(j,i);
j++;i=/*3*/2;stt_en(j,i);
j++;i=18;stt_en(j,i);

return j;
}/** en_xt **/


int en_yt(void)
{
int i,j;

j=-1;

j++;i=/*2*/3;stt_en(j,i);
j++;i=17;stt_en(j,i);
j++;i=4;stt_en(j,i);
j++;i=18;stt_en(j,i);

return j;
}/** en_yt **/
```

```
int en_zt(void)
{
int i,j;

j=-1;

j++;i=5;stt_en(j,i);
j++;i=17;stt_en(j,i);
j++;i=6;stt_en(j,i);
j++;i=18;stt_en(j,i);

return j;
}/** en_zt **/


int en_ut(void)
{
int i,j;

j=-1;

j++;i=7;stt_en(j,i);
j++;i=17;stt_en(j,i);
j++;i=8;stt_en(j,i);
j++;i=18;stt_en(j,i);

return j;
}/** en_ut **/


int en_vt(void)
{
int i,j;

j=-1;

j++;i=9;stt_en(j,i);
j++;i=17;stt_en(j,i);
j++;i=10;stt_en(j,i);
j++;i=18;stt_en(j,i);

return j;
}/** en_vt **/


int en_wt(void)
{
int i,j;

j=-1;

j++;i=11;stt_en(j,i);
j++;i=17;stt_en(j,i);
```

```
j++;i=12;stt_en(j,i);
j++;i=18;stt_en(j,i);

return j;
}/** en_wt **/


int en_rt(void)
{
int i,j;

j=-1;

j++;i=13;stt_en(j,i);
j++;i=17;stt_en(j,i);
j++;i=14;stt_en(j,i);
j++;i=18;stt_en(j,i);

return j;
}/** en_rt **/


int en_st(void)
{
int i,j;

j=-1;

j++;i=15;stt_en(j,i);
j++;i=17;stt_en(j,i);
j++;i=16;stt_en(j,i);
j++;i=18;stt_en(j,i);

return j;
}/** en_st **/


int en_xo(void)
{
int i,j;

j=-1;

j++;i=1;stt_en(j,i);
j++;i=19;stt_en(j,i);
j++;i=/*3*/2;stt_en(j,i);
j++;i=20;stt_en(j,i);

return j;
}/** en_xo **/


int en_yo(void)
{
int i,j;
```

```
j=-1;

j++;i=/*2*/3;stt_en(j,i);
j++;i=19;stt_en(j,i);
j++;i=4;stt_en(j,i);
j++;i=20;stt_en(j,i);

return j;
}/** en_yo **/


int en_zo(void)
{
int i,j;

j=-1;

j++;i=5;stt_en(j,i);
j++;i=19;stt_en(j,i);
j++;i=6;stt_en(j,i);
j++;i=20;stt_en(j,i);

return j;
}/** en_zo **/


int en_uo(void)
{
int i,j;

j=-1;

j++;i=7;stt_en(j,i);
j++;i=19;stt_en(j,i);
j++;i=8;stt_en(j,i);
j++;i=20;stt_en(j,i);

return j;
}/** en_uo **/


int en_vo(void)
{
int i,j;

j=-1;

j++;i=9;stt_en(j,i);
j++;i=19;stt_en(j,i);
j++;i=10;stt_en(j,i);
j++;i=20;stt_en(j,i);

return j;
}/** en_vo **/
```

```
int en_wo(void)
{
int i,j;

j=-1;

j++;i=11;stt_en(j,i);
j++;i=19;stt_en(j,i);
j++;i=12;stt_en(j,i);
j++;i=20;stt_en(j,i);

return j;
}/** en_wo **/


int en_ro(void)
{
int i,j;

j=-1;

j++;i=13;stt_en(j,i);
j++;i=19;stt_en(j,i);
j++;i=14;stt_en(j,i);
j++;i=20;stt_en(j,i);

return j;
}/** en_ro **/


int en_so(void)
{
int i,j;

j=-1;

j++;i=15;stt_en(j,i);
j++;i=19;stt_en(j,i);
j++;i=16;stt_en(j,i);
j++;i=20;stt_en(j,i);

return j;
}/** en_so **/


int en_to(void)
{
int i,j;

j=-1;

j++;i=17;stt_en(j,i);
j++;i=19;stt_en(j,i);
```

```
j++;i=18;stt_en(j,i);
j++;i=20;stt_en(j,i);

return j;
}/** en_to **/




/* here A11 *//* change 0 into 1 */
#if 0
/*
en_xp():i=1,  i=25, i=2,  i=26
en_yp():i=3,  i=21, i=4,  i=22
en_zp():i=5,  i=21, i=6, i=22
en_up():i=7, i=21, i=8, i=22
en_vp():i=9, i=21, i=10, i=22
en_wp():i=11, i=21, i=12, i=22
en_rp():i=13, i=21, i=14, i=22
en_sp():i=15, i=21, i=16, i=22
en_tp():i=17, i=21, i=18, i=22
en_op():i=19, i=21, i=20, i=22
*/
int en_xp(void){
int i,j;

j=-1;

j++;i=1;stt_en(j,i);
j++;i=21;stt_en(j,i);
j++;i=/*3*/2;stt_en(j,i);
j++;i=22;stt_en(j,i);

return j;
}/** en_xp **/



int en_yp(void){
int i,j;

j=-1;

j++;i=/*2*/3;stt_en(j,i);
j++;i=21;stt_en(j,i);
j++;i=4;stt_en(j,i);
j++;i=22;stt_en(j,i);

return j;
}/** en_yp **/



int en_zp(void){
int i,j;

j=-1;
```

```
j++;i=5;stt_en(j,i);
j++;i=21;stt_en(j,i);
j++;i=6;stt_en(j,i);
j++;i=22;stt_en(j,i);

return j;
}/** en_zp **/


int en_up(void){
int i,j;

j=-1;

j++;i=7;stt_en(j,i);
j++;i=21;stt_en(j,i);
j++;i=8;stt_en(j,i);
j++;i=22;stt_en(j,i);

return j;
}/** en_up **/


int en_vp(void){
int i,j;

j=-1;

j++;i=9;stt_en(j,i);
j++;i=21;stt_en(j,i);
j++;i=10;stt_en(j,i);
j++;i=22;stt_en(j,i);

return j;
}/** en_vp **/


int en_wp(void){
int i,j;

j=-1;

j++;i=11;stt_en(j,i);
j++;i=21;stt_en(j,i);
j++;i=12;stt_en(j,i);
j++;i=22;stt_en(j,i);

return j;
}/** en_wp **/


int en_rp(void){
int i,j;

j=-1;
```

```
j++;i=13;stt_en(j,i);
j++;i=21;stt_en(j,i);
j++;i=14;stt_en(j,i);
j++;i=22;stt_en(j,i);

return j;
}/** en_rp **/


int en_sp(void){
int i,j;

j=-1;

j++;i=15;stt_en(j,i);
j++;i=21;stt_en(j,i);
j++;i=16;stt_en(j,i);
j++;i=22;stt_en(j,i);

return j;
}/** en_sp **/


int en_tp(void){
int i,j;

j=-1;

j++;i=17;stt_en(j,i);
j++;i=21;stt_en(j,i);
j++;i=18;stt_en(j,i);
j++;i=22;stt_en(j,i);

return j;
}/** en_tp **/


int en_op(void){
int i,j;

j=-1;

j++;i=19;stt_en(j,i);
j++;i=21;stt_en(j,i);
j++;i=20;stt_en(j,i);
j++;i=22;stt_en(j,i);

return j;
}/** en_op **/
#endif


/* here B4 */                                                        /* add int p */
int getplane(int flag,int x,int y,int z,int u,int v,int w,int r,int s,int t,int o)
```

```
{
int plane=-1,ds1,ds2,ds3;

ds1=0;ds2=0;ds3=0;

/* xy:xyzu */
/* here B10 *//* replace o==0 with p==0 */
     if((_V==0)&&(x>=0+ds1 && x<=xt-ds1)&&(y>=0+ds1 && y<=yt-ds1)&&
        (z>=0+ds1 && z<=zt-ds1)&&
        (u>=0+ds1 && u<=ut-ds1)&&(v>=0+ds1 && v<=vt-ds1)&&
        (w>=0+ds1 && w<=wt-ds1)&&(r>=0+ds1 && r<=rt-ds1)&&
        (s>=0+ds1 && s<=st-ds1)&&(t>=0+ds1 && t<=tt-ds1)
/* here B10 *//* add &&(o>=0+ds1 && o<=ot-ds1) */
       ) plane=0;
/* here B10 *//* replace o==ot with p==pt */
else if((_V==_VT)&&(x>=0+ds1 && x<=xt-ds1)&&(y>=0+ds1 && y<=yt-ds1)&&
        (z>=0+ds1 && z<=zt-ds1)&&
        (u>=0+ds1 && u<=ut-ds1)&&(v>=0+ds1 && v<=vt-ds1)&&
        (w>=0+ds1 && w<=wt-ds1)&&(r>=0+ds1 && r<=rt-ds1)&&
        (s>=0+ds1 && s<=st-ds1)&&(t>=0+ds1 && t<=tt-ds1)
/* here B10 *//* add &&(o>=0+ds1 && o<=ot-ds1) */
       ) plane=1;

else if(flag){
/* zx_:zxuv */
     if((y==0)&&(x>=0+ds1 && x<=xt-ds1)&&(z>=0+ds1 && z<=zt-ds1)&&
        (u>=0+ds1 && u<=ut-ds1)&&(v>=0+ds1 && v<=vt-ds1)&&
        (w>=0+ds1 && w<=wt-ds1)&&(r>=0+ds1 && r<=rt-ds1)&&
        (s>=0+ds1 && s<=st-ds1)&&(t>=0+ds1 && t<=tt-ds1)&&
        (o>=0+ds1 && o<=ot-ds1)
/* here B10 *//* add &&(p>=0+ds1 && p<=pt-ds1) */
        ) plane=2;
else if((y==yt)&&(x>=0+ds1 && x<=xt-ds1)&&(z>=0+ds1 && z<=zt-ds1)&&
        (u>=0+ds1 && u<=ut-ds1)&&(v>=0+ds1 && v<=vt-ds1)&&
        (w>=0+ds1 && w<=wt-ds1)&&(r>=0+ds1 && r<=rt-ds1)&&
        (s>=0+ds1 && s<=st-ds1)&&(t>=0+ds1 && t<=tt-ds1)&&
        (o>=0+ds1 && o<=ot-ds1)
/* here B10 *//* add &&(p>=0+ds1 && p<=pt-ds1) */
        ) plane=3;
/* yz_:yzuv */
else if((x==0)&&(y>=0+ds1 && y<=yt-ds1)&&(z>=0+ds1 && z<=zt-ds1)&&
        (u>=0+ds1 && u<=ut-ds1)&&(v>=0+ds1 && v<=vt-ds1)&&
        (w>=0+ds1 && w<=wt-ds1)&&(r>=0+ds1 && r<=rt-ds1)&&
        (s>=0+ds1 && s<=st-ds1)&&(t>=0+ds1 && t<=tt-ds1)&&
        (o>=0+ds1 && o<=ot-ds1)
/* here B10 *//* add &&(p>=0+ds1 && p<=pt-ds1) */
        ) plane=4;
else if((x==xt)&&(y>=0+ds1 && y<=yt-ds1)&&(z>=0+ds1 && z<=zt-ds1)&&
        (u>=0+ds1 && u<=ut-ds1)&&(v>=0+ds1 && v<=vt-ds1)&&
        (w>=0+ds1 && w<=wt-ds1)&&(r>=0+ds1 && r<=rt-ds1)&&
        (s>=0+ds1 && s<=st-ds1)&&(t>=0+ds1 && t<=tt-ds1)&&
        (o>=0+ds1 && o<=ot-ds1)
/* here B10 *//* add &&(p>=0+ds1 && p<=pt-ds1) */
        ) plane=5;
#if 1
/* xy_:xyuv */
else if((z==0)&&(x>=0+ds1 && x<=xt-ds1)&&(y>=0+ds1 && y<=yt-ds1)&&
        (u>=0+ds1 && u<=ut-ds1)&&(v>=0+ds1 && v<=vt-ds1)&&
        (w>=0+ds1 && w<=wt-ds1)&&(r>=0+ds1 && r<=rt-ds1)&&
        (s>=0+ds1 && s<=st-ds1)&&(t>=0+ds1 && t<=tt-ds1)&&
        (o>=0+ds1 && o<=ot-ds1)
/* here B10 *//* add &&(p>=0+ds1 && p<=pt-ds1) */
        ) plane=6;
else if((z==zt)&&(x>=0+ds1 && x<=xt-ds1)&&(y>=0+ds1 && y<=yt-ds1)&&
        (u>=0+ds1 && u<=ut-ds1)&&(v>=0+ds1 && v<=vt-ds1)&&
        (w>=0+ds1 && w<=wt-ds1)&&(r>=0+ds1 && r<=rt-ds1)&&
        (s>=0+ds1 && s<=st-ds1)&&(t>=0+ds1 && t<=tt-ds1)&&
        (o>=0+ds1 && o<=ot-ds1)
/* here B10 *//* add &&(p>=0+ds1 && p<=pt-ds1) */
        ) plane=7;
#endif
}

return plane;
}/** getplane **/



#if CHK_H && CPMAX>1
#if PBC
void check_hole(int begin,int i)
{
int j,ds1;
static int k;
static long pcnt[CPMAX];

if(!begin) {if(k<0) return;}
else       {k=0;return;}
ds1=delta_x-0;

for(j=0;j<CPMAX;j++){
/* here A10 *//* add && np[i]==nap[j]-ds1 */
if(nx[i]==nax[j]-ds1 && ny[i]==nay[j]-ds1 && nz[i]==naz[j]-ds1 && nu[i]==nau[j]-ds1 &&
   nv[i]==nav[j]-ds1 && nw[i]==naw[j]-ds1 && nr[i]==nar[j]-ds1 && ns[i]==nas[j]-ds1 &&
   nt[i]==nat[j]-ds1 && no[i]==nao[j]-ds1) goto next;
}

return;

next:
pcnt[k]=pcount[i];
k++;
if(k==CPMAX){
k=1;
while(1){
if(pcnt[k]!=pcnt[0]) {refill=-2;break;}

k++;if(k==CPMAX) {k=-1;break;}
}
}
```

```
}/** check_hole **/
#else
void check_hole(int begin,int i)
{
char pflag=0;
/* here B3 */           /* add PT */
int j,ds1,ZT,UT,VT,WT,RT,ST,TT,OT;
static int k;
static long pcnt[CPMAX];

if(!begin) {if(k<0) return;}
else       {k=0;return;}
ds1=1;

/* here B7 */                /* add PT=pt; */
UT=ut;VT=vt;WT=wt;RT=rt;ST=st;TT=tt;OT=ot;

# if CPMAX<=4
/* here B10 */                       /* add && np[i]==PT */
     if(nx[i]==0+2 && ny[i]==0 && nz[i]==zt-ds1 && nu[i]==UT && nv[i]==VT && nw[i]==WT &&
        nr[i]==RT && ns[i]==ST && nt[i]==TT && no[i]==OT)
{if(pflag) printf(" i:%d %lld\n",i,pcount[i]);}

/* here B10 */                       /* add && np[i]==PT */
else if(nx[i]==xt-2 && ny[i]==yt && nz[i]==zt-ds1 && nu[i]==UT && nv[i]==VT && nw[i]==WT &&
        nr[i]==RT && ns[i]==ST && nt[i]==TT && no[i]==OT)
{if(pflag) printf(" i:%d %lld\n",i,pcount[i]);}

# if CPMAX==4
/* here B10 */                       /* add && np[i]==PT */
else if(nx[i]==xt && ny[i]==0+2 && nz[i]==zt-ds1 && nu[i]==UT && nv[i]==VT && nw[i]==WT &&
        nr[i]==RT && ns[i]==ST && nt[i]==TT && no[i]==OT)
{if(pflag) printf(" i:%d %lld\n",i,pcount[i]);}

/* here B10 */                       /* add && np[i]==PT */
else if(nx[i]==0 && ny[i]==yt-2 && nz[i]==zt-ds1 && nu[i]==UT && nv[i]==VT && nw[i]==WT &&
        nr[i]==RT && ns[i]==ST && nt[i]==TT && no[i]==OT)
{if(pflag) printf(" i:%d %lld\n",i,pcount[i]);}
#endif

else return;
#elif CPMAX==8
/* here B10 */                       /* add && np[i]==PT */
     if(nx[i]==0+2 && ny[i]==0 && nz[i]==0+ds1 && nu[i]==UT && nv[i]==VT && nw[i]==WT &&
        nr[i]==RT && ns[i]==ST && nt[i]==TT && no[i]==OT)
{if(pflag) printf(" i:%d %lld\n",i,pcount[i]);}
/* here B10 */                       /* add && np[i]==PT */
else if(nx[i]==xt-2 && ny[i]==0 && nz[i]==zt-ds1 && nu[i]==UT && nv[i]==VT && nw[i]==WT &&
        nr[i]==RT && ns[i]==ST && nt[i]==TT && no[i]==OT)
{if(pflag) printf(" i:%d %lld\n",i,pcount[i]);}

/* here B10 */                       /* add && np[i]==PT */
else if(nx[i]==xt && ny[i]==0+2 && nz[i]==0+ds1 && nu[i]==UT && nv[i]==VT && nw[i]==WT &&
        nr[i]==RT && ns[i]==ST && nt[i]==TT && no[i]==OT)
{if(pflag) printf(" i:%d %lld\n",i,pcount[i]);}
```

```
/* here B10 */                          /* add && np[i]==PT */
else if(nx[i]==xt && ny[i]==yt-2 && nz[i]==zt-ds1 && nu[i]==UT && nv[i]==VT && nw[i]==WT &&
        nr[i]==RT && ns[i]==ST && nt[i]==TT && no[i]==OT)
{if(pflag) printf(" i:%d %lld\n",i,pcount[i]);}

/* here B10 */                          /* add && np[i]==PT */
else if(nx[i]==xt-2 && ny[i]==yt && nz[i]==0+ds1 && nu[i]==UT && nv[i]==VT && nw[i]==WT &&
        nr[i]==RT && ns[i]==ST && nt[i]==TT && no[i]==OT)
{if(pflag) printf(" i:%d %lld\n",i,pcount[i]);}
/* here B10 */                          /* add && np[i]==PT */
else if(nx[i]==0+2 && ny[i]==yt && nz[i]==zt-ds1 && nu[i]==UT && nv[i]==VT && nw[i]==WT &&
        nr[i]==RT && ns[i]==ST && nt[i]==TT && no[i]==OT)
{if(pflag) printf(" i:%d %lld\n",i,pcount[i]);}

/* here B10 */                          /* add && np[i]==PT */
else if(nx[i]==0 && ny[i]==yt-2 && nz[i]==0+ds1 && nu[i]==UT && nv[i]==VT && nw[i]==WT &&
        nr[i]==RT && ns[i]==ST && nt[i]==TT && no[i]==OT)
{if(pflag) printf(" i:%d %lld\n",i,pcount[i]);}
/* here B10 */                          /* add && np[i]==PT */
else if(nx[i]==0 && ny[i]==0+2 && nz[i]==zt-ds1 && nu[i]==UT && nv[i]==VT && nw[i]==WT &&
        nr[i]==RT && ns[i]==ST && nt[i]==TT && no[i]==OT)
{if(pflag) printf(" i:%d %lld\n",i,pcount[i]);}

else return;
#endif

/*return;*/
pcnt[k]=pcount[i];
k++;
if(k==CPMAX){
k=1;
while(1){
if(pcnt[k]!=pcnt[0]) {refill=-2;break;}

k++;if(k==CPMAX) {k=-1;break;}
}

end:{}
}
}/** check_hole **/
#endif
#endif



int pm_algo(int algo)
{
if(algo%2==0) algo++;else algo--;

return algo;
}/** pm_algo **/



int pm_algo_en(char flag,int algo,int algo_)
{
int val,rot;
```

```
#if 1
val=algo%2;
if(flag==/*0*/1){
if(algo_%2==0) rot=0;else rot=1;    /* return algo(algo%2!=algo_%2) */
}
else{
if(algo_%2==0) rot=1;else rot=0;    /* return algo(algo%2==algo_%2) */
}

/* CW => CCW, CCW => CW */
if(rot==0){         /* is CW */
if(val==0) algo++;  /* to CCW */
}
else{               /* is CCW */
if(val==1) algo--;  /* to CW */
}
#endif

return algo;
}/** pm_algo_en **/



int PM_algo_en(int flag,int algo)
{
#if 1
/* here B10 */   /* add || flag==45 */
if(flag==0 || flag==1 || flag==3 || flag==6 || flag==10 || flag==15 || flag==21 ||
   flag==28 || flag==36){   /* delta_flag=2,3,4,5,6,7,8,9,... */
/* 01100101 */
     /*if(ig==0) algo=pm_algo_en(0,algo,ui);
else */if(ig==1) algo=pm_algo_en(/*0*/1,algo,ui);
else if(ig==2) algo=pm_algo_en(/*0*/1,algo,ui);
else if(ig==3) algo=pm_algo_en(/*1*/0,algo,ui);

else if(ig==4) algo=pm_algo_en(/*1*/0,algo,ui);
else if(ig==5) algo=pm_algo_en(/*0*/1,algo,ui);
else if(ig==6) algo=pm_algo_en(/*1*/0,algo,ui);
else if(ig==7) algo=pm_algo_en(/*0*/1,algo,ui);
}/**else if(flag)**/
else{
/* 01010110 */
     /*if(ig==0) algo=pm_algo_en(0,algo,ui);
else */if(ig==1) algo=pm_algo_en(/*0*/1,algo,ui);
else if(ig==2) algo=pm_algo_en(/*1*/0,algo,ui);
else if(ig==3) algo=pm_algo_en(/*0*/1,algo,ui);

else if(ig==4) algo=pm_algo_en(/*1*/0,algo,ui);
else if(ig==5) algo=pm_algo_en(/*0*/1,algo,ui);
else if(ig==6) algo=pm_algo_en(/*0*/1,algo,ui);
else if(ig==7) algo=pm_algo_en(/*1*/0,algo,ui);
}/**else(flag)**/
#elif 0
if(flag==0 || flag==1 || flag==3 || flag==6 || flag==10 || flag==15 || flag==21 ||
   flag==28 || flag==36){
```

```
/* swapped */
/* 01010110 */
     /*if(ig==0) algo=pm_algo_en(0,algo,ui);
else */if(ig==1) algo=pm_algo_en(/*0*/1,algo,ui);
else if(ig==2) algo=pm_algo_en(/*1*/0,algo,ui);
else if(ig==3) algo=pm_algo_en(/*0*/1,algo,ui);

else if(ig==4) algo=pm_algo_en(/*1*/0,algo,ui);
else if(ig==5) algo=pm_algo_en(/*0*/1,algo,ui);
else if(ig==6) algo=pm_algo_en(/*0*/1,algo,ui);
else if(ig==7) algo=pm_algo_en(/*1*/0,algo,ui);
}/**else if(flag)**/
else{
/* 01100101 */
/* swapped */
     /*if(ig==0) algo=pm_algo_en(0,algo,ui);
else */if(ig==1) algo=pm_algo_en(/*0*/1,algo,ui);
else if(ig==2) algo=pm_algo_en(/*0*/1,algo,ui);
else if(ig==3) algo=pm_algo_en(/*1*/0,algo,ui);

else if(ig==4) algo=pm_algo_en(/*1*/0,algo,ui);
else if(ig==5) algo=pm_algo_en(/*0*/1,algo,ui);
else if(ig==6) algo=pm_algo_en(/*1*/0,algo,ui);
else if(ig==7) algo=pm_algo_en(/*0*/1,algo,ui);
}/**else(flag)**/
#endif

return algo;
}/** PM_algo_en **/



/* here A4 */                 /* add int n9_,int n9 */
int Su(int n1_,int n1,int n2_,int n2,int n3_,int n3,int n4_,int n4,int n5_,int n5,
       int n6_,int n6,int n7_,int n7,int n8_,int n8)
{
/* here A10 */ /* add && n9_==n9 */
if(n1_==n1 && n2_==n2 && n3_==n3 && n4_==n4 && n5_==n5 &&
   n6_==n6 && n7_==n7 && n8_==n8)
     return 1;
else return 0;
}/** Su **/



int C(int c1,int c3,int c2,int c4)
{
if(c1==ca || c3==ca || c2==ca || c4==ca) return 1;
else return 0;
}/** C **/



int set_ens_1(int algo)
{
if(algo/2==0){
/* xy */
/* here A9 */                                                /* add Np_,Np */
```

```
if(C(c1,c3,c2,c4)&& Su(Nz_,Nz,Nu_,Nu,Nv_,Nv,Nw_,Nw,Nr_,Nr,Ns_,Ns,Nt_,Nt,No_,No)){
jmax=en_xy();                      /* xy */
}
else if(C(c3,c4,c5,c6)&& Su(Nx_,Nx,Nu_,Nu,Nv_,Nv,Nw_,Nw,Nr_,Nr,Ns_,Ns,Nt_,Nt,No_,No)){
jmax=en_yz();                      /* yz */
}
else if(C(c1,c2,c5,c6)&& Su(Ny_,Ny,Nu_,Nu,Nv_,Nv,Nw_,Nw,Nr_,Nr,Ns_,Ns,Nt_,Nt,No_,No)){
jmax=en_zx();                      /* zx */
}

else if(C(c1,c2,c7,c8)&& Su(Ny_,Ny,Nz_,Nz,Nv_,Nv,Nw_,Nw,Nr_,Nr,Ns_,Ns,Nt_,Nt,No_,No)){
jmax=en_xu();                      /* xu */
}
else if(C(c3,c4,c7,c8)&& Su(Nx_,Nx,Nz_,Nz,Nv_,Nv,Nw_,Nw,Nr_,Nr,Ns_,Ns,Nt_,Nt,No_,No)){
jmax=en_yu();                      /* yu */
}
else if(C(c5,c6,c7,c8)&& Su(Nx_,Nx,Ny_,Ny,Nv_,Nv,Nw_,Nw,Nr_,Nr,Ns_,Ns,Nt_,Nt,No_,No)){
jmax=en_zu();                      /* zu */
}

else if(C(c1,c2,c9,c10)&& Su(Ny_,Ny,Nz_,Nz,Nu_,Nu,Nw_,Nw,Nr_,Nr,Ns_,Ns,Nt_,Nt,No_,No)){
jmax=en_xv();                      /* xv */
}
else if(C(c3,c4,c9,c10)&& Su(Nx_,Nx,Nz_,Nz,Nu_,Nu,Nw_,Nw,Nr_,Nr,Ns_,Ns,Nt_,Nt,No_,No)){
jmax=en_yv();                      /* yv */
}
else if(C(c5,c6,c9,c10)&& Su(Nx_,Nx,Ny_,Ny,Nu_,Nu,Nw_,Nw,Nr_,Nr,Ns_,Ns,Nt_,Nt,No_,No)){
jmax=en_zv();                      /* zv */
}
else if(C(c7,c8,c9,c10)&& Su(Nx_,Nx,Ny_,Ny,Nz_,Nz,Nw_,Nw,Nr_,Nr,Ns_,Ns,Nt_,Nt,No_,No)){
jmax=en_uv();                      /* uv */
}

else if(C(c1,c2,c11,c12)&& Su(Ny_,Ny,Nz_,Nz,Nu_,Nu,Nv_,Nv,Nr_,Nr,Ns_,Ns,Nt_,Nt,No_,No)){
jmax=en_xw();                      /* xw */
}
else if(C(c3,c4,c11,c12)&& Su(Nx_,Nx,Nz_,Nz,Nu_,Nu,Nv_,Nv,Nr_,Nr,Ns_,Ns,Nt_,Nt,No_,No)){
jmax=en_yw();                      /* yw */
}
else if(C(c5,c6,c11,c12)&& Su(Nx_,Nx,Ny_,Ny,Nu_,Nu,Nv_,Nv,Nr_,Nr,Ns_,Ns,Nt_,Nt,No_,No)){
jmax=en_zw();                      /* zw */
}
else if(C(c7,c8,c11,c12)&& Su(Nx_,Nx,Ny_,Ny,Nz_,Nz,Nv_,Nv,Nr_,Nr,Ns_,Ns,Nt_,Nt,No_,No)){
jmax=en_uw();                      /* uw */
}
else if(C(c9,c10,c11,c12)&& Su(Nx_,Nx,Ny_,Ny,Nz_,Nz,Nu_,Nu,Nr_,Nr,Ns_,Ns,Nt_,Nt,No_,No)){
jmax=en_vw();                      /* vw */
}

else if(C(c1,c2,c13,c14)&& Su(Ny_,Ny,Nz_,Nz,Nu_,Nu,Nv_,Nv,Nw_,Nw,Ns_,Ns,Nt_,Nt,No_,No)){
jmax=en_xr();                      /* xr */
}
else if(C(c3,c4,c13,c14)&& Su(Nx_,Nx,Nz_,Nz,Nu_,Nu,Nv_,Nv,Nw_,Nw,Ns_,Ns,Nt_,Nt,No_,No)){
jmax=en_yr();                      /* yr */
}
```

```
else if(C(c5,c6,c13,c14)&& Su(Nx_,Nx,Ny_,Ny,Nu_,Nu,Nv_,Nv,Nw_,Nw,Ns_,Ns,Nt_,Nt,No_,No)){
jmax=en_zr();                         /* zr */
}
else if(C(c7,c8,c13,c14)&& Su(Nx_,Nx,Ny_,Ny,Nz_,Nz,Nv_,Nv,Nw_,Nw,Ns_,Ns,Nt_,Nt,No_,No)){
jmax=en_ur();                         /* ur */
}
else if(C(c9,c10,c13,c14)&& Su(Nx_,Nx,Ny_,Ny,Nz_,Nz,Nu_,Nu,Nw_,Nw,Ns_,Ns,Nt_,Nt,No_,No)){
jmax=en_vr();                         /* vr */
}
else if(C(c11,c12,c13,c14)&& Su(Nx_,Nx,Ny_,Ny,Nz_,Nz,Nu_,Nu,Nv_,Nv,Ns_,Ns,Nt_,Nt,No_,No)){
jmax=en_wr();                         /* wr */
}

else if(C(c1,c2,c15,c16)&& Su(Ny_,Ny,Nz_,Nz,Nu_,Nu,Nv_,Nv,Nw_,Nw,Nr_,Nr,Nt_,Nt,No_,No)){
jmax=en_xs();                         /* xs */
}
else if(C(c3,c4,c15,c16)&& Su(Nx_,Nx,Nz_,Nz,Nu_,Nu,Nv_,Nv,Nw_,Nw,Nr_,Nr,Nt_,Nt,No_,No)){
jmax=en_ys();                         /* ys */
}
else if(C(c5,c6,c15,c16)&& Su(Nx_,Nx,Ny_,Ny,Nu_,Nu,Nv_,Nv,Nw_,Nw,Nr_,Nr,Nt_,Nt,No_,No)){
jmax=en_zs();                         /* zs */
}
else if(C(c7,c8,c15,c16)&& Su(Nx_,Nx,Ny_,Ny,Nz_,Nz,Nv_,Nv,Nw_,Nw,Nr_,Nr,Nt_,Nt,No_,No)){
jmax=en_us();                         /* us */
}
else if(C(c9,c10,c15,c16)&& Su(Nx_,Nx,Ny_,Ny,Nz_,Nz,Nu_,Nu,Nw_,Nw,Nr_,Nr,Nt_,Nt,No_,No)){
jmax=en_vs();                         /* vs */
}
else if(C(c11,c12,c15,c16)&& Su(Nx_,Nx,Ny_,Ny,Nz_,Nz,Nu_,Nu,Nv_,Nv,Nr_,Nr,Nt_,Nt,No_,No)){
jmax=en_ws();                         /* ws */
}
else if(C(c13,c14,c15,c16)&& Su(Nx_,Nx,Ny_,Ny,Nz_,Nz,Nu_,Nu,Nv_,Nv,Nw_,Nw,Nt_,Nt,No_,No)){
jmax=en_rs();                         /* rs */
}

else if(C(c1,c2,c17,c18)&& Su(Ny_,Ny,Nz_,Nz,Nu_,Nu,Nv_,Nv,Nw_,Nw,Nr_,Nr,Ns_,Ns,No_,No)){
jmax=en_xt();                         /* xt */
}
else if(C(c3,c4,c17,c18)&& Su(Nx_,Nx,Nz_,Nz,Nu_,Nu,Nv_,Nv,Nw_,Nw,Nr_,Nr,Ns_,Ns,No_,No)){
jmax=en_yt();                         /* yt */
}
else if(C(c5,c6,c17,c18)&& Su(Nx_,Nx,Ny_,Ny,Nu_,Nu,Nv_,Nv,Nw_,Nw,Nr_,Nr,Ns_,Ns,No_,No)){
jmax=en_zt();                         /* zt */
}
else if(C(c7,c8,c17,c18)&& Su(Nx_,Nx,Ny_,Ny,Nz_,Nz,Nv_,Nv,Nw_,Nw,Nr_,Nr,Ns_,Ns,No_,No)){
jmax=en_ut();                         /* ut */
}
else if(C(c9,c10,c17,c18)&& Su(Nx_,Nx,Ny_,Ny,Nz_,Nz,Nu_,Nu,Nw_,Nw,Nr_,Nr,Ns_,Ns,No_,No)){
jmax=en_vt();                         /* vt */
}
else if(C(c11,c12,c17,c18)&& Su(Nx_,Nx,Ny_,Ny,Nz_,Nz,Nu_,Nu,Nv_,Nv,Nr_,Nr,Ns_,Ns,No_,No)){
jmax=en_wt();                         /* wt */
}
else if(C(c13,c14,c17,c18)&& Su(Nx_,Nx,Ny_,Ny,Nz_,Nz,Nu_,Nu,Nv_,Nv,Nw_,Nw,Ns_,Ns,No_,No)){
jmax=en_rt();                         /* rt */
}
else if(C(c15,c16,c17,c18)&& Su(Nx_,Nx,Ny_,Ny,Nz_,Nz,Nu_,Nu,Nv_,Nv,Nw_,Nw,Nr_,Nr,No_,No)){
jmax=en_st();                         /* st */
}

/* o sector */
else if(C(c1,c2,c19,c20)&& Su(Ny_,Ny,Nz_,Nz,Nu_,Nu,Nv_,Nv,Nw_,Nw,Nr_,Nr,Ns_,Ns,Nt_,Nt)){
jmax=en_xo();                         /* xo */
}
else if(C(c3,c4,c19,c20)&& Su(Nx_,Nx,Nz_,Nz,Nu_,Nu,Nv_,Nv,Nw_,Nw,Nr_,Nr,Ns_,Ns,Nt_,Nt)){
jmax=en_yo();                         /* yo */
}
else if(C(c5,c6,c19,c20)&& Su(Nx_,Nx,Ny_,Ny,Nu_,Nu,Nv_,Nv,Nw_,Nw,Nr_,Nr,Ns_,Ns,Nt_,Nt)){
jmax=en_zo();                         /* zo */
}
else if(C(c7,c8,c19,c20)&& Su(Nx_,Nx,Ny_,Ny,Nz_,Nz,Nv_,Nv,Nw_,Nw,Nr_,Nr,Ns_,Ns,Nt_,Nt)){
jmax=en_uo();                         /* uo */
}
else if(C(c9,c10,c19,c20)&& Su(Nx_,Nx,Ny_,Ny,Nz_,Nz,Nu_,Nu,Nw_,Nw,Nr_,Nr,Ns_,Ns,Nt_,Nt)){
jmax=en_vo();                         /* vo */
}
else if(C(c11,c12,c19,c20)&& Su(Nx_,Nx,Ny_,Ny,Nz_,Nz,Nu_,Nu,Nv_,Nv,Nr_,Nr,Ns_,Ns,Nt_,Nt)){
jmax=en_wo();                         /* wo */
}
else if(C(c13,c14,c19,c20)&& Su(Nx_,Nx,Ny_,Ny,Nz_,Nz,Nu_,Nu,Nv_,Nv,Nw_,Nw,Ns_,Ns,Nt_,Nt)){
jmax=en_ro();                         /* ro */
}
else if(C(c15,c16,c19,c20)&& Su(Nx_,Nx,Ny_,Ny,Nz_,Nz,Nu_,Nu,Nv_,Nv,Nw_,Nw,Nr_,Nr,Nt_,Nt)){
jmax=en_so();                         /* so */
}
else if(C(c17,c18,c19,c20)&& Su(Nx_,Nx,Ny_,Ny,Nz_,Nz,Nu_,Nu,Nv_,Nv,Nw_,Nw,Nr_,Nr,Ns_,Ns)){
jmax=en_to();                         /* to */
}

/* here A6 *//* change 0 into 1 */
#if 0
/* p sector */
else if(C(c1,c2,c21,c22)&& Su(Ny_,Ny,Nz_,Nz,Nu_,Nu,Nv_,Nv,Nw_,Nw,Nr_,Nr,Ns_,Ns,Nt_,Nt,No_,No)){
jmax=en_xp();                         /* xp */
}
else if(C(c3,c4,c21,c22)&& Su(Nx_,Nx,Nz_,Nz,Nu_,Nu,Nv_,Nv,Nw_,Nw,Nr_,Nr,Ns_,Ns,Nt_,Nt,No_,No)){
jmax=en_yp();                         /* yp */
}
else if(C(c5,c6,c21,c22)&& Su(Nx_,Nx,Ny_,Ny,Nu_,Nu,Nv_,Nv,Nw_,Nw,Nr_,Nr,Ns_,Ns,Nt_,Nt,No_,No)){
jmax=en_zp();                         /* zp */
}
else if(C(c7,c8,c21,c22)&& Su(Nx_,Nx,Ny_,Ny,Nz_,Nz,Nv_,Nv,Nw_,Nw,Nr_,Nr,Ns_,Ns,Nt_,Nt,No_,No)){
jmax=en_up();                         /* up */
}
else if(C(c9,c10,c21,c22)&& Su(Nx_,Nx,Ny_,Ny,Nz_,Nz,Nu_,Nu,Nw_,Nw,Nr_,Nr,Ns_,Ns,Nt_,Nt,No_,No)){
jmax=en_vp();                         /* vp */
}
else if(C(c11,c12,c21,c22)&& Su(Nx_,Nx,Ny_,Ny,Nz_,Nz,Nu_,Nu,Nv_,Nv,Nr_,Nr,Ns_,Ns,Nt_,Nt,No_,No)){
jmax=en_wp();                         /* wp */
}
```

```
else if(C(c13,c14,c21,c22)&& Su(Nx_,Nx,Ny_,Ny,Nz_,Nz,Nu_,Nu,Nv_,Nv,Nw_,Nw,Ns_,Ns,Nt_,Nt,No_,No)){
jmax=en_rp();                     /* rp */
}
else if(C(c15,c16,c21,c22)&& Su(Nx_,Nx,Ny_,Ny,Nz_,Nz,Nu_,Nu,Nv_,Nv,Nw_,Nw,Nr_,Nr,Nt_,Nt,No_,No)){
jmax=en_sp();                     /* sp */
}
else if(C(c17,c18,c21,c22)&& Su(Nx_,Nx,Ny_,Ny,Nz_,Nz,Nu_,Nu,Nv_,Nv,Nw_,Nw,Nr_,Nr,Ns_,Ns,No_,No)){
jmax=en_tp();                     /* tp */
}
else if(C(c19,c20,c21,c22)&& Su(Nx_,Nx,Ny_,Ny,Nz_,Nz,Nu_,Nu,Nv_,Nv,Nw_,Nw,Nr_,Nr,Ns_,Ns,Nt_,Nt)){
jmax=en_op();                     /* op */
}
#endif

else {printf(" ?algo/2\n");refill=-3;}
}/**if(algo/2)**/

return algo;
}/** set_ens_1 **/



int set_ens_0(int i,int algo)
{
char debug=0/*1*/,gotoflag=0;
int i_,j_,i_d2,BASE_4;

i_d2=i/2;

/*99*/
if(i_d2==0){
BASE_4=0;
i_=BASE_4+1;
j_=BASE_4+1;
}
else if(i_d2==1){
BASE_4=0;
i_=BASE_4+2;
j_=BASE_4+3;
}
else if(i_d2==2){
BASE_4=4;
i_=BASE_4+0;
j_=BASE_4+0;
}
else if(i_d2==3){
BASE_4=4;
i_=BASE_4+2;
j_=BASE_4+3;
}




if(i_d2==0 || (CPMAX==8 && i_d2==3)){
if(algo/2==0){
/* xy  */
/* here B9 */                                                   /* add Np_,Np */
if(C(c1,c3,c2,c4)&& Su(Nz_,Nz,Nu_,Nu,Nv_,Nv,Nw_,Nw,Nr_,Nr,Ns_,Ns,Nt_,Nt,No_,No)){
jmax=en_xy();                       /* xy */
fprintf_(debug,i,-1,algo," xy-0-xy");
}
else if(C(c3,c4,c5,c6)&& Su(Nx_,Nx,Nu_,Nu,Nv_,Nv,Nw_,Nw,Nr_,Nr,Ns_,Ns,Nt_,Nt,No_,No)){
jmax=en_yz();                       /* yz */
if(i==0) ui=algo;
if(!PMflag) {if(i==i_) PM_algo}
else algo=PM_algo_en(1,algo);
fprintf_(debug,i,-1,algo," xy-1-yz");
}
else if(C(c1,c2,c5,c6)&& Su(Ny_,Ny,Nu_,Nu,Nv_,Nv,Nw_,Nw,Nr_,Nr,Ns_,Ns,Nt_,Nt,No_,No)){
jmax=en_zx();                       /* zx */
if(i==0) ui=algo;
if(!PMflag) {if(i==j_) PM_algo}
else algo=PM_algo_en(2,algo);
fprintf_(debug,i,-1,algo," xy-2-zx");
}

else gotoflag=1;
}/**if(algo/2)**/
else if(algo/2==1){
/* yz  */
     if(C(c3,c4,c5,c6)&& Su(Nx_,Nx,Nu_,Nu,Nv_,Nv,Nw_,Nw,Nr_,Nr,Ns_,Ns,Nt_,Nt,No_,No)){
jmax=en_yz();                       /* yz */
if(i==0) ui=algo;
if(!PMflag) {if(i==i_) PM_algo}
else algo=PM_algo_en(0,algo);
}
else if(C(c1,c2,c5,c6)&& Su(Ny_,Ny,Nu_,Nu,Nv_,Nv,Nw_,Nw,Nr_,Nr,Ns_,Ns,Nt_,Nt,No_,No)){
jmax=en_zx();                       /* zx */
if(i==0) ui=algo;
if(!PMflag) {if(i==j_) PM_algo}
else algo=PM_algo_en(-1,algo);
}
else if(C(c1,c3,c2,c4)&& Su(Nz_,Nz,Nu_,Nu,Nv_,Nv,Nw_,Nw,Nr_,Nr,Ns_,Ns,Nt_,Nt,No_,No)){
jmax=en_xy();                       /* xy */
}

else gotoflag=1;
}/**else if(algo/2)**/
else if(algo/2==2){
/* zx  */
     if(C(c1,c2,c5,c6)&& Su(Ny_,Ny,Nu_,Nu,Nv_,Nv,Nw_,Nw,Nr_,Nr,Ns_,Ns,Nt_,Nt,No_,No)){
jmax=en_zx();                       /* zx */
if(i==0) ui=algo;
if(!PMflag) {if(i==j_) PM_algo}
else algo=PM_algo_en(-1,algo);
}
else if(C(c3,c4,c5,c6)&& Su(Nx_,Nx,Nu_,Nu,Nv_,Nv,Nw_,Nw,Nr_,Nr,Ns_,Ns,Nt_,Nt,No_,No)){
jmax=en_yz();                       /* yz */
if(i==0) ui=algo;
if(!PMflag) {if(i==i_) PM_algo}
else algo=PM_algo_en(0,algo);
}
else if(C(c1,c3,c2,c4)&& Su(Nz_,Nz,Nu_,Nu,Nv_,Nv,Nw_,Nw,Nr_,Nr,Ns_,Ns,Nt_,Nt,No_,No)){
jmax=en_xy();                       /* xy */
}

else gotoflag=1;
}/**else if(algo/2)**/
}
else if(i_d2==1 || (CPMAX==8 && i_d2==2)){
if(algo/2==0){
/* xy  **/
if(C(c1,c3,c2,c4)&& Su(Nz_,Nz,Nu_,Nu,Nv_,Nv,Nw_,Nw,Nr_,Nr,Ns_,Ns,Nt_,Nt,No_,No)){
jmax=en_xy();                       /* xy */
fprintf_(debug,i,-1,algo," xy-0-xy");
}
else if(C(c1,c2,c5,c6)&& Su(Ny_,Ny,Nu_,Nu,Nv_,Nv,Nw_,Nw,Nr_,Nr,Ns_,Ns,Nt_,Nt,No_,No)){
jmax=en_zx();                       /* zx */
if(!PMflag) {if(i==i_) PM_algo}
else algo=PM_algo_en(1,algo);
fprintf_(debug,i,-1,algo," xy-1-zx");
}
else if(C(c3,c4,c5,c6)&& Su(Nx_,Nx,Nu_,Nu,Nv_,Nv,Nw_,Nw,Nr_,Nr,Ns_,Ns,Nt_,Nt,No_,No)){
jmax=en_yz();                       /* yz */
if(!PMflag) {if(i==j_) PM_algo}
else algo=PM_algo_en(2,algo);
fprintf_(debug,i,-1,algo," xy-2-yz");
}

else gotoflag=1;
}/**if(algo/2)**/
else if(algo/2==1){
/* yz  **/
     if(C(c1,c2,c5,c6)&& Su(Ny_,Ny,Nu_,Nu,Nv_,Nv,Nw_,Nw,Nr_,Nr,Ns_,Ns,Nt_,Nt,No_,No)){
jmax=en_zx();                       /* zx */
if(!PMflag) {if(i==i_) PM_algo}
else algo=PM_algo_en(0,algo);
}
else if(C(c3,c4,c5,c6)&& Su(Nx_,Nx,Nu_,Nu,Nv_,Nv,Nw_,Nw,Nr_,Nr,Ns_,Ns,Nt_,Nt,No_,No)){
jmax=en_yz();                       /* yz */
if(!PMflag) {if(i==j_) PM_algo}
else algo=PM_algo_en(-1,algo);
}
else if(C(c1,c3,c2,c4)&& Su(Nz_,Nz,Nu_,Nu,Nv_,Nv,Nw_,Nw,Nr_,Nr,Ns_,Ns,Nt_,Nt,No_,No)){
jmax=en_xy();                       /* xy */
}

else gotoflag=1;
}/**else if(algo/2)**/
else if(algo/2==2){
/* zx  **/
     if(C(c3,c4,c5,c6)&& Su(Nx_,Nx,Nu_,Nu,Nv_,Nv,Nw_,Nw,Nr_,Nr,Ns_,Ns,Nt_,Nt,No_,No)){
jmax=en_yz();                       /* yz */
if(!PMflag) {if(i==j_) PM_algo}
else algo=PM_algo_en(-1,algo);
}
```

```
else if(C(c1,c2,c5,c6)&& Su(Ny_,Ny,Nu_,Nu,Nv_,Nv,Nw_,Nw,Nr_,Nr,Ns_,Ns,Nt_,Nt,No_,No)){
jmax=en_zx();                         /* zx */
if(!PMflag) {if(i==i_) PM_algo}
else algo=PM_algo_en(0,algo);
}
else if(C(c1,c3,c2,c4)&& Su(Nz_,Nz,Nu_,Nu,Nv_,Nv,Nw_,Nw,Nr_,Nr,Ns_,Ns,Nt_,Nt,No_,No)){
jmax=en_xy();                         /* xy */
}

else gotoflag=1;
}/**else if(algo/2)**/
}

if(!gotoflag) goto end;



if(i_d2==0 || (CPMAX==8 && i_d2==3)){
/* u sector */
if(C(c1,c2,c7,c8)&& Su(Ny_,Ny,Nz_,Nz,Nv_,Nv,Nw_,Nw,Nr_,Nr,Ns_,Ns,Nt_,Nt,No_,No)){
jmax=en_xu();                         /* xu */
if(i==0) ui=algo;
if(!PMflag) {if(i==i_) PM_algo}
else algo=PM_algo_en(3,algo);
fprintf_(debug,i,-1,algo," xy-3-xu");
}
else if(C(c3,c4,c7,c8)&& Su(Nx_,Nx,Nz_,Nz,Nv_,Nv,Nw_,Nw,Nr_,Nr,Ns_,Ns,Nt_,Nt,No_,No)){
jmax=en_yu();                         /* yu */
if(i==0) ui=algo;
if(!PMflag) {if(i==j_) PM_algo}
else algo=PM_algo_en(4,algo);
fprintf_(debug,i,-1,algo," xy-4-yu");
}
else if(C(c5,c6,c7,c8)&& Su(Nx_,Nx,Ny_,Ny,Nv_,Nv,Nw_,Nw,Nr_,Nr,Ns_,Ns,Nt_,Nt,No_,No)){
jmax=en_zu();                         /* zu */
fprintf_(debug,i,-1,algo," xy-5-zu");
}

else if(C(c1,c2,c9,c10)&& Su(Ny_,Ny,Nz_,Nz,Nu_,Nu,Nw_,Nw,Nr_,Nr,Ns_,Ns,Nt_,Nt,No_,No)){
jmax=en_xv();                         /* xv */
if(i==0) ui=algo;
if(!PMflag) {if(i==i_) PM_algo}
else algo=PM_algo_en(6,algo);
fprintf_(debug,i,-1,algo," xy-6-xv");
}
else if(C(c3,c4,c9,c10)&& Su(Nx_,Nx,Nz_,Nz,Nu_,Nu,Nw_,Nw,Nr_,Nr,Ns_,Ns,Nt_,Nt,No_,No)){
jmax=en_yv();                         /* yv */
if(i==0) ui=algo;
if(!PMflag) {if(i==j_) PM_algo}
else algo=PM_algo_en(7,algo);
fprintf_(debug,i,-1,algo," xy-7-yv");
}
else if(C(c5,c6,c9,c10)&& Su(Nx_,Nx,Ny_,Ny,Nu_,Nu,Nw_,Nw,Nr_,Nr,Ns_,Ns,Nt_,Nt,No_,No)){
jmax=en_zv();                         /* zv */
fprintf_(debug,i,-1,algo," xy-8-zv");
}
```

```
else if(i>=4 && (algo=pm_algo(algo))<0) ;
else if(C(c7,c8,c9,c10)&& Su(Nx_,Nx,Ny_,Ny,Nz_,Nz,Nw_,Nw,Nr_,Nr,Ns_,Ns,Nt_,Nt,No_,No)){
jmax=en_uv();                         /* uv */
fprintf_(debug,i,-1,algo," xy-9-uv");
}

else if(i>=4 && (algo=pm_algo(algo))<0) ;
else if(C(c1,c2,c11,c12)&& Su(Ny_,Ny,Nz_,Nz,Nu_,Nu,Nv_,Nv,Nr_,Nr,Ns_,Ns,Nt_,Nt,No_,No)){
jmax=en_xw();                         /* xw */
if(i==0) ui=algo;
if(!PMflag) {if(i==i_) PM_algo}
else algo=PM_algo_en(10,algo);
fprintf_(debug,i,-1,algo," xy-10-xw");
}
else if(C(c3,c4,c11,c12)&& Su(Nx_,Nx,Nz_,Nz,Nu_,Nu,Nv_,Nv,Nr_,Nr,Ns_,Ns,Nt_,Nt,No_,No)){
jmax=en_yw();                         /* yw */
if(i==0) ui=algo;
if(!PMflag) {if(i==j_) PM_algo}
else algo=PM_algo_en(11,algo);
fprintf_(debug,i,-1,algo," xy-11-yw");
}
else if(C(c5,c6,c11,c12)&& Su(Nx_,Nx,Ny_,Ny,Nu_,Nu,Nv_,Nv,Nr_,Nr,Ns_,Ns,Nt_,Nt,No_,No)){
jmax=en_zw();                         /* zw */
fprintf_(debug,i,-1,algo," xy-12-zw");
}
else if(i>=4 && (algo=pm_algo(algo))<0) ;
else if(C(c7,c8,c11,c12)&& Su(Nx_,Nx,Ny_,Ny,Nz_,Nz,Nv_,Nv,Nr_,Nr,Ns_,Ns,Nt_,Nt,No_,No)){
jmax=en_uw();                         /* uw */
fprintf_(debug,i,-1,algo," xy-13-uw");
}
else if(C(c9,c10,c11,c12)&& Su(Nx_,Nx,Ny_,Ny,Nz_,Nz,Nu_,Nu,Nr_,Nr,Ns_,Ns,Nt_,Nt,No_,No)){
jmax=en_vw();                         /* vw */
fprintf_(debug,i,-1,algo," xy-14-vw");
}

else if(i>=4 && (algo=pm_algo(algo))<0) ;
else if(C(c1,c2,c13,c14)&& Su(Ny_,Ny,Nz_,Nz,Nu_,Nu,Nv_,Nv,Nw_,Nw,Ns_,Ns,Nt_,Nt,No_,No)){
jmax=en_xr();                         /* xr */
if(i==0) ui=algo;
if(!PMflag) {if(i==i_) PM_algo}
else algo=PM_algo_en(15,algo);
fprintf_(debug,i,-1,algo," xy-15-xr");
}
else if(C(c3,c4,c13,c14)&& Su(Nx_,Nx,Nz_,Nz,Nu_,Nu,Nv_,Nv,Nw_,Nw,Ns_,Ns,Nt_,Nt,No_,No)){
jmax=en_yr();                         /* yr */
if(i==0) ui=algo;
if(!PMflag) {if(i==j_) PM_algo}
else algo=PM_algo_en(16,algo);
fprintf_(debug,i,-1,algo," xy-16-yr");
}
else if(C(c5,c6,c13,c14)&& Su(Nx_,Nx,Ny_,Ny,Nu_,Nu,Nv_,Nv,Nw_,Nw,Ns_,Ns,Nt_,Nt,No_,No)){
jmax=en_zr();                         /* zr */
fprintf_(debug,i,-1,algo," xy-17-zr");
}
else if(i>=4 && (algo=pm_algo(algo))<0) ;
else if(C(c7,c8,c13,c14)&& Su(Nx_,Nx,Ny_,Ny,Nz_,Nz,Nv_,Nv,Nw_,Nw,Ns_,Ns,Nt_,Nt,No_,No)){
jmax=en_ur();                         /* ur */
fprintf_(debug,i,-1,algo," xy-18-ur");
}
else if(C(c9,c10,c13,c14)&& Su(Nx_,Nx,Ny_,Ny,Nz_,Nz,Nu_,Nu,Nw_,Nw,Ns_,Ns,Nt_,Nt,No_,No)){
jmax=en_vr();                         /* vr */
fprintf_(debug,i,-1,algo," xy-19-vr");
}
else if(C(c11,c12,c13,c14)&& Su(Nx_,Nx,Ny_,Ny,Nz_,Nz,Nu_,Nu,Nv_,Nv,Ns_,Ns,Nt_,Nt,No_,No)){
jmax=en_wr();                         /* wr */
fprintf_(debug,i,-1,algo," xy-20-wr");
}

else if(i>=4 && (algo=pm_algo(algo))<0) ;
else if(C(c1,c2,c15,c16)&& Su(Ny_,Ny,Nz_,Nz,Nu_,Nu,Nv_,Nv,Nw_,Nw,Nr_,Nr,Nt_,Nt,No_,No)){
jmax=en_xs();                         /* xs */
if(i==0) ui=algo;
if(!PMflag) {if(i==i_) PM_algo}
else algo=PM_algo_en(21,algo);
fprintf_(debug,i,-1,algo," xy-21-xs");
}
else if(C(c3,c4,c15,c16)&& Su(Nx_,Nx,Nz_,Nz,Nu_,Nu,Nv_,Nv,Nw_,Nw,Nr_,Nr,Nt_,Nt,No_,No)){
jmax=en_ys();                         /* ys */
if(i==0) ui=algo;
if(!PMflag) {if(i==j_) PM_algo}
else algo=PM_algo_en(22,algo);
fprintf_(debug,i,-1,algo," xy-22-ys");
}
else if(C(c5,c6,c15,c16)&& Su(Nx_,Nx,Ny_,Ny,Nu_,Nu,Nv_,Nv,Nw_,Nw,Nr_,Nr,Nt_,Nt,No_,No)){
jmax=en_zs();                         /* zs */
fprintf_(debug,i,-1,algo," xy-23-zs");
}
else if(i>=4 && (algo=pm_algo(algo))<0) ;
else if(C(c7,c8,c15,c16)&& Su(Nx_,Nx,Ny_,Ny,Nz_,Nz,Nv_,Nv,Nw_,Nw,Nr_,Nr,Nt_,Nt,No_,No)){
jmax=en_us();                         /* us */
fprintf_(debug,i,-1,algo," xy-24-us");
}
else if(C(c9,c10,c15,c16)&& Su(Nx_,Nx,Ny_,Ny,Nz_,Nz,Nu_,Nu,Nw_,Nw,Nr_,Nr,Nt_,Nt,No_,No)){
jmax=en_vs();                         /* vs */
fprintf_(debug,i,-1,algo," xy-25-vs");
}
else if(C(c11,c12,c15,c16)&& Su(Nx_,Nx,Ny_,Ny,Nz_,Nz,Nu_,Nu,Nv_,Nv,Nr_,Nr,Nt_,Nt,No_,No)){
jmax=en_ws();                         /* ws */
fprintf_(debug,i,-1,algo," xy-26-ws");
}
else if(C(c13,c14,c15,c16)&& Su(Nx_,Nx,Ny_,Ny,Nz_,Nz,Nu_,Nu,Nv_,Nv,Nw_,Nw,Nt_,Nt,No_,No)){
jmax=en_rs();                         /* rs */
fprintf_(debug,i,-1,algo," xy-27-ws");
}

else if(i>=4 && (algo=pm_algo(algo))<0) ;
else if(C(c1,c2,c17,c18)&& Su(Ny_,Ny,Nz_,Nz,Nu_,Nu,Nv_,Nv,Nw_,Nw,Nr_,Nr,Ns_,Ns,No_,No)){
jmax=en_xt();                         /* xt */
if(i==0) ui=algo;
if(!PMflag) {if(i==i_) PM_algo}
```

```
else algo=PM_algo_en(28,algo);
fprintf_(debug,i,-1,algo," xy-28-xt");
}
else if(C(c3,c4,c17,c18)&& Su(Nx_,Nx,Nz_,Nz,Nu_,Nu,Nv_,Nv,Nw_,Nw,Nr_,Nr,Ns_,Ns,No_,No)){
jmax=en_yt();                        /* yt */
if(i==0) ui=algo;
if(!PMflag) {if(i==j_) PM_algo}
else algo=PM_algo_en(29,algo);
fprintf_(debug,i,-1,algo," xy-29-yt");
}
else if(C(c5,c6,c17,c18)&& Su(Nx_,Nx,Ny_,Ny,Nu_,Nu,Nv_,Nv,Nw_,Nw,Nr_,Nr,Ns_,Ns,No_,No)){
jmax=en_zt();                        /* zt */
fprintf_(debug,i,-1,algo," xy-30-zt");
}
else if(i>=4 && (algo=pm_algo(algo))<0) ;
else if(C(c7,c8,c17,c18)&& Su(Nx_,Nx,Ny_,Ny,Nz_,Nz,Nv_,Nv,Nw_,Nw,Nr_,Nr,Ns_,Ns,No_,No)){
jmax=en_ut();                        /* ut */
fprintf_(debug,i,-1,algo," xy-31-ut");
}
else if(C(c9,c10,c17,c18)&& Su(Nx_,Nx,Ny_,Ny,Nz_,Nz,Nu_,Nu,Nw_,Nw,Nr_,Nr,Ns_,Ns,No_,No)){
jmax=en_vt();                        /* vt */
fprintf_(debug,i,-1,algo," xy-32-vt");
}
else if(C(c11,c12,c17,c18)&& Su(Nx_,Nx,Ny_,Ny,Nz_,Nz,Nu_,Nu,Nv_,Nv,Nr_,Nr,Ns_,Ns,No_,No)){
jmax=en_wt();                        /* wt */
fprintf_(debug,i,-1,algo," xy-33-wt");
}
else if(C(c13,c14,c17,c18)&& Su(Nx_,Nx,Ny_,Ny,Nz_,Nz,Nu_,Nu,Nv_,Nv,Nw_,Nw,Ns_,Ns,No_,No)){
jmax=en_rt();                        /* rt */
fprintf_(debug,i,-1,algo," xy-34-rt");
}
else if(C(c15,c16,c17,c18)&& Su(Nx_,Nx,Ny_,Ny,Nz_,Nz,Nu_,Nu,Nv_,Nv,Nw_,Nw,Nr_,Nr,No_,No)){
jmax=en_st();                        /* st */
fprintf_(debug,i,-1,algo," xy-35-st");
}

/* o sector */
else if(i>=4 && (algo=pm_algo(algo))<0) ;
else if(C(c1,c2,c19,c20)&& Su(Ny_,Ny,Nz_,Nz,Nu_,Nu,Nv_,Nv,Nw_,Nw,Nr_,Nr,Ns_,Ns,Nt_,Nt)){
jmax=en_xo();                        /* xo */
if(i==0) ui=algo;
if(!PMflag) {if(i==i_) PM_algo}
else algo=PM_algo_en(36,algo);
fprintf_(debug,i,-1,algo," xy-36-xo");
}
else if(C(c3,c4,c19,c20)&& Su(Nx_,Nx,Nz_,Nz,Nu_,Nu,Nv_,Nv,Nw_,Nw,Nr_,Nr,Ns_,Ns,Nt_,Nt)){
jmax=en_yo();                        /* yo */
if(i==0) ui=algo;
if(!PMflag) {if(i==j_) PM_algo}
else algo=PM_algo_en(37,algo);
fprintf_(debug,i,-1,algo," xy-37-yo");
}
else if(C(c5,c6,c19,c20)&& Su(Nx_,Nx,Ny_,Ny,Nu_,Nu,Nv_,Nv,Nw_,Nw,Nr_,Nr,Ns_,Ns,Nt_,Nt)){
jmax=en_zo();                        /* zo */
fprintf_(debug,i,-1,algo," xy-38-zo");
```

```
}
else if(i>=4 && (algo=pm_algo(algo))<0) ;
else if(C(c7,c8,c19,c20)&& Su(Nx_,Nx,Ny_,Ny,Nz_,Nz,Nv_,Nv,Nw_,Nw,Nr_,Nr,Ns_,Ns,Nt_,Nt)){
jmax=en_uo();                        /* uo */
fprintf_(debug,i,-1,algo," xy-39-uo");
}
else if(C(c9,c10,c19,c20)&& Su(Nx_,Nx,Ny_,Ny,Nz_,Nz,Nu_,Nu,Nw_,Nw,Nr_,Nr,Ns_,Ns,Nt_,Nt)){
jmax=en_vo();                        /* vo */
fprintf_(debug,i,-1,algo," xy-40-vo");
}
else if(C(c11,c12,c19,c20)&& Su(Nx_,Nx,Ny_,Ny,Nz_,Nz,Nu_,Nu,Nv_,Nv,Nr_,Nr,Ns_,Ns,Nt_,Nt)){
jmax=en_wo();                        /* wo */
fprintf_(debug,i,-1,algo," xy-41-wo");
}
else if(C(c13,c14,c19,c20)&& Su(Nx_,Nx,Ny_,Ny,Nz_,Nz,Nu_,Nu,Nv_,Nv,Nw_,Nw,Ns_,Ns,Nt_,Nt)){
jmax=en_ro();                        /* ro */
fprintf_(debug,i,-1,algo," xy-42-ro");
}
else if(C(c15,c16,c19,c20)&& Su(Nx_,Nx,Ny_,Ny,Nz_,Nz,Nu_,Nu,Nv_,Nv,Nw_,Nw,Nr_,Nr,Nt_,Nt)){
jmax=en_so();                        /* so */
fprintf_(debug,i,-1,algo," xy-43-so");
}
else if(C(c17,c18,c19,c20)&& Su(Nx_,Nx,Ny_,Ny,Nz_,Nz,Nu_,Nu,Nv_,Nv,Nw_,Nw,Nr_,Nr,Ns_,Ns)){
jmax=en_to();                        /* to */
fprintf_(debug,i,-1,algo," xy-44-to");
}



/* here B6 *//* change 0 into 1 */
#if 0
/* p sector */
else if(i>=4 && (algo=pm_algo(algo))<0) ;
else if(C(c1,c2,c21,c22)&& Su(Ny_,Ny,Nz_,Nz,Nu_,Nu,Nv_,Nv,Nw_,Nw,Nr_,Nr,Ns_,Ns,Nt_,Nt,No_,No)){
jmax=en_xp();                        /* xp */
if(i==0) ui=algo;
if(!PMflag) {if(i==i_) PM_algo}
else algo=PM_algo_en(45,algo);
fprintf_(debug,i,-1,algo," xy-45-xp");
}
else if(C(c3,c4,c21,c22)&& Su(Nx_,Nx,Nz_,Nz,Nu_,Nu,Nv_,Nv,Nw_,Nw,Nr_,Nr,Ns_,Ns,Nt_,Nt,No_,No)){
jmax=en_yp();                        /* yp */
if(i==0) ui=algo;
if(!PMflag) {if(i==j_) PM_algo}
else algo=PM_algo_en(46,algo);
fprintf_(debug,i,-1,algo," xy-46-yp");
}
else if(C(c5,c6,c21,c22)&& Su(Nx_,Nx,Ny_,Ny,Nu_,Nu,Nv_,Nv,Nw_,Nw,Nr_,Nr,Ns_,Ns,Nt_,Nt,No_,No)){
jmax=en_zp();                        /* zp */
fprintf_(debug,i,-1,algo," xy-47-zp");
}
else if(i>=4 && (algo=pm_algo(algo))<0) ;
else if(C(c7,c8,c21,c22)&& Su(Nx_,Nx,Ny_,Ny,Nz_,Nz,Nv_,Nv,Nw_,Nw,Nr_,Nr,Ns_,Ns,Nt_,Nt,No_,No)){
jmax=en_up();                        /* up */
fprintf_(debug,i,-1,algo," xy-48-up");
```

```
}
else if(C(c9,c10,c21,c22)&& Su(Nx_,Nx,Ny_,Ny,Nz_,Nz,Nu_,Nu,Nw_,Nw,Nr_,Nr,Ns_,Ns,Nt_,Nt,No_,No)){
jmax=en_vp();                          /* vp */
fprintf_(debug,i,-1,algo," xy-49-vp");
}
else if(C(c11,c12,c21,c22)&& Su(Nx_,Nx,Ny_,Ny,Nz_,Nz,Nu_,Nu,Nv_,Nv,Nr_,Nr,Ns_,Ns,Nt_,Nt,No_,No)){
jmax=en_wp();                          /* wp */
fprintf_(debug,i,-1,algo," xy-50-wp");
}
else if(C(c13,c14,c21,c22)&& Su(Nx_,Nx,Ny_,Ny,Nz_,Nz,Nu_,Nu,Nv_,Nv,Nw_,Nw,Ns_,Ns,Nt_,Nt,No_,No)){
jmax=en_rp();                          /* rp */
fprintf_(debug,i,-1,algo," xy-51-rp");
}
else if(C(c15,c16,c21,c22)&& Su(Nx_,Nx,Ny_,Ny,Nz_,Nz,Nu_,Nu,Nv_,Nv,Nw_,Nw,Nr_,Nr,Nt_,Nt,No_,No)){
jmax=en_sp();                          /* sp */
fprintf_(debug,i,-1,algo," xy-51-sp");
}
else if(C(c17,c18,c21,c22)&& Su(Nx_,Nx,Ny_,Ny,Nz_,Nz,Nu_,Nu,Nv_,Nv,Nw_,Nw,Nr_,Nr,Ns_,Ns,No_,No)){
jmax=en_tp();                          /* tp */
fprintf_(debug,i,-1,algo," xy-52-tp");
}
else if(C(c19,c20,c21,c22)&& Su(Nx_,Nx,Ny_,Ny,Nz_,Nz,Nu_,Nu,Nv_,Nv,Nw_,Nw,Nr_,Nr,Ns_,Ns,Nt_,Nt)){
jmax=en_op();                          /* op */
fprintf_(debug,i,-1,algo," xy-53-op");
}
#endif




else {printf(" ?algo/2\n");refill=-3;}
}
else if(i_d2==1 || (CPMAX==8 && i_d2==2)){
/* u sector */
if(C(c3,c4,c7,c8)&& Su(Nx_,Nx,Nz_,Nz,Nv_,Nv,Nw_,Nw,Nr_,Nr,Ns_,Ns,Nt_,Nt,No_,No)){
jmax=en_yu();                          /* yu */
if(!PMflag) {if(i==i_) PM_algo}
else algo=PM_algo_en(3,algo);
fprintf_(debug,i,-1,algo," xy-3-yu");
}
else if(C(c1,c2,c7,c8)&& Su(Ny_,Ny,Nz_,Nz,Nv_,Nv,Nw_,Nw,Nr_,Nr,Ns_,Ns,Nt_,Nt,No_,No)){
jmax=en_xu();                          /* xu */
if(!PMflag) {if(i==j_) PM_algo}
else algo=PM_algo_en(4,algo);
fprintf_(debug,i,-1,algo," xy-4-xu");
}
else if(C(c5,c6,c7,c8)&& Su(Nx_,Nx,Ny_,Ny,Nv_,Nv,Nw_,Nw,Nr_,Nr,Ns_,Ns,Nt_,Nt,No_,No)){
jmax=en_zu();                          /* zu */
fprintf_(debug,i,-1,algo," xy-5-zu");
}

else if(C(c3,c4,c9,c10)&& Su(Nx_,Nx,Nz_,Nz,Nu_,Nu,Nw_,Nw,Nr_,Nr,Ns_,Ns,Nt_,Nt,No_,No)){
jmax=en_yv();                          /* yv */
if(!PMflag) {if(i==i_) PM_algo}
else algo=PM_algo_en(6,algo);
fprintf_(debug,i,-1,algo," xy-6-yv");
```

```
}
else if(C(c1,c2,c9,c10)&& Su(Ny_,Ny,Nz_,Nz,Nu_,Nu,Nw_,Nw,Nr_,Nr,Ns_,Ns,Nt_,Nt,No_,No)){
jmax=en_xv();                       /* xv */
if(!PMflag) {if(i==j_) PM_algo}
else algo=PM_algo_en(7,algo);
fprintf_(debug,i,-1,algo," xy-7-xv");
}
else if(C(c5,c6,c9,c10)&& Su(Nx_,Nx,Ny_,Ny,Nu_,Nu,Nw_,Nw,Nr_,Nr,Ns_,Ns,Nt_,Nt,No_,No)){
jmax=en_zv();                       /* zv */
fprintf_(debug,i,-1,algo," xy-8-zv");
}
else if(i>=4 && (algo=pm_algo(algo))<0) ;
else if(C(c7,c8,c9,c10)&& Su(Nx_,Nx,Ny_,Ny,Nz_,Nz,Nw_,Nw,Nr_,Nr,Ns_,Ns,Nt_,Nt,No_,No)){
jmax=en_uv();                       /* uv */
fprintf_(debug,i,-1,algo," xy-9-uv");
}

else if(i>=4 && (algo=pm_algo(algo))<0) ;
else if(C(c3,c4,c11,c12)&& Su(Nx_,Nx,Nz_,Nz,Nu_,Nu,Nv_,Nv,Nr_,Nr,Ns_,Ns,Nt_,Nt,No_,No)){
jmax=en_yw();                       /* yw */
if(!PMflag) {if(i==i_) PM_algo}
else algo=PM_algo_en(10,algo);
fprintf_(debug,i,-1,algo," xy-10-yw");
}
else if(C(c1,c2,c11,c12)&& Su(Ny_,Ny,Nz_,Nz,Nu_,Nu,Nv_,Nv,Nr_,Nr,Ns_,Ns,Nt_,Nt,No_,No)){
jmax=en_xw();                       /* xw */
if(!PMflag) {if(i==j_) PM_algo}
else algo=PM_algo_en(11,algo);
fprintf_(debug,i,-1,algo," xy-11-xw");
}
else if(C(c5,c6,c11,c12)&& Su(Nx_,Nx,Ny_,Ny,Nu_,Nu,Nv_,Nv,Nr_,Nr,Ns_,Ns,Nt_,Nt,No_,No)){
jmax=en_zw();                       /* zw */
fprintf_(debug,i,-1,algo," xy-12-zw");
}
else if(i>=4 && (algo=pm_algo(algo))<0) ;
else if(C(c7,c8,c11,c12)&& Su(Nx_,Nx,Ny_,Ny,Nz_,Nz,Nv_,Nv,Nr_,Nr,Ns_,Ns,Nt_,Nt,No_,No)){
jmax=en_uw();                       /* uw */
fprintf_(debug,i,-1,algo," xy-13-uw");
}
else if(C(c9,c10,c11,c12)&& Su(Nx_,Nx,Ny_,Ny,Nz_,Nz,Nu_,Nu,Nr_,Nr,Ns_,Ns,Nt_,Nt,No_,No)){
jmax=en_vw();                       /* vw */
fprintf_(debug,i,-1,algo," xy-14-vw");
}

else if(i>=4 && (algo=pm_algo(algo))<0) ;
else if(C(c3,c4,c13,c14)&& Su(Nx_,Nx,Nz_,Nz,Nu_,Nu,Nv_,Nv,Nw_,Nw,Ns_,Ns,Nt_,Nt,No_,No)){
jmax=en_yr();                       /* yr */
if(!PMflag) {if(i==i_) PM_algo}
else algo=PM_algo_en(15,algo);
fprintf_(debug,i,-1,algo," xy-15-yr");
}
else if(C(c1,c2,c13,c14)&& Su(Ny_,Ny,Nz_,Nz,Nu_,Nu,Nv_,Nv,Nw_,Nw,Ns_,Ns,Nt_,Nt,No_,No)){
jmax=en_xr();                       /* xr */
if(!PMflag) {if(i==j_) PM_algo}
else algo=PM_algo_en(16,algo);
```

```
fprintf_(debug,i,-1,algo," xy-16-xr");
}
else if(C(c5,c6,c13,c14)&& Su(Nx_,Nx,Ny_,Ny,Nu_,Nu,Nv_,Nv,Nw_,Nw,Ns_,Ns,Nt_,Nt,No_,No)){
jmax=en_zr();                       /* zr */
fprintf_(debug,i,-1,algo," xy-17-zr");
}
else if(i>=4 && (algo=pm_algo(algo))<0) ;
else if(C(c7,c8,c13,c14)&& Su(Nx_,Nx,Ny_,Ny,Nz_,Nz,Nv_,Nv,Nw_,Nw,Ns_,Ns,Nt_,Nt,No_,No)){
jmax=en_ur();                       /* ur */
fprintf_(debug,i,-1,algo," xy-18-ur");
}
else if(C(c9,c10,c13,c14)&& Su(Nx_,Nx,Ny_,Ny,Nz_,Nz,Nu_,Nu,Nw_,Nw,Ns_,Ns,Nt_,Nt,No_,No)){
jmax=en_vr();                       /* vr */
fprintf_(debug,i,-1,algo," xy-19-vr");
}
else if(C(c11,c12,c13,c14)&& Su(Nx_,Nx,Ny_,Ny,Nz_,Nz,Nu_,Nu,Nv_,Nv,Ns_,Ns,Nt_,Nt,No_,No)){
jmax=en_wr();                       /* wr */
fprintf_(debug,i,-1,algo," xy-20-wr");
}

else if(i>=4 && (algo=pm_algo(algo))<0) ;
else if(C(c3,c4,c15,c16)&& Su(Nx_,Nx,Nz_,Nz,Nu_,Nu,Nv_,Nv,Nw_,Nw,Nr_,Nr,Nt_,Nt,No_,No)){
jmax=en_ys();                       /* ys */
if(!PMflag) {if(i==i_) PM_algo}
else algo=PM_algo_en(21,algo);
fprintf_(debug,i,-1,algo," xy-21-ys");
}
else if(C(c1,c2,c15,c16)&& Su(Ny_,Ny,Nz_,Nz,Nu_,Nu,Nv_,Nv,Nw_,Nw,Nr_,Nr,Nt_,Nt,No_,No)){
jmax=en_xs();                       /* xs */
if(!PMflag) {if(i==j_) PM_algo}
else algo=PM_algo_en(22,algo);
fprintf_(debug,i,-1,algo," xy-22-xs");
}
else if(C(c5,c6,c15,c16)&& Su(Nx_,Nx,Ny_,Ny,Nu_,Nu,Nv_,Nv,Nw_,Nw,Nr_,Nr,Nt_,Nt,No_,No)){
jmax=en_zs();                       /* zs */
fprintf_(debug,i,-1,algo," xy-23-zs");
}
else if(i>=4 && (algo=pm_algo(algo))<0) ;
else if(C(c7,c8,c15,c16)&& Su(Nx_,Nx,Ny_,Ny,Nz_,Nz,Nv_,Nv,Nw_,Nw,Nr_,Nr,Nt_,Nt,No_,No)){
jmax=en_us();                       /* us */
fprintf_(debug,i,-1,algo," xy-24-us");
}
else if(C(c9,c10,c15,c16)&& Su(Nx_,Nx,Ny_,Ny,Nz_,Nz,Nu_,Nu,Nw_,Nw,Nr_,Nr,Nt_,Nt,No_,No)){
jmax=en_vs();                       /* vs */
fprintf_(debug,i,-1,algo," xy-25-vs");
}
else if(C(c11,c12,c15,c16)&& Su(Nx_,Nx,Ny_,Ny,Nz_,Nz,Nu_,Nu,Nv_,Nv,Nr_,Nr,Nt_,Nt,No_,No)){
jmax=en_ws();                       /* ws */
fprintf_(debug,i,-1,algo," xy-26-ws");
}
else if(C(c13,c14,c15,c16)&& Su(Nx_,Nx,Ny_,Ny,Nz_,Nz,Nu_,Nu,Nv_,Nv,Nw_,Nw,Nt_,Nt,No_,No)){
jmax=en_rs();                       /* rs */
fprintf_(debug,i,-1,algo," xy-27-ws");
}
```

```
else if(i>=4 && (algo=pm_algo(algo))<0) ;
else if(C(c3,c4,c17,c18)&& Su(Nx_,Nx,Nz_,Nz,Nu_,Nu,Nv_,Nv,Nw_,Nw,Nr_,Nr,Ns_,Ns,No_,No)){
jmax=en_yt();                           /* yt */
if(!PMflag) {if(i==i_) PM_algo}
else algo=PM_algo_en(28,algo);
fprintf_(debug,i,-1,algo," xy-28-yt");
}
else if(C(c1,c2,c17,c18)&& Su(Ny_,Ny,Nz_,Nz,Nu_,Nu,Nv_,Nv,Nw_,Nw,Nr_,Nr,Ns_,Ns,No_,No)){
jmax=en_xt();                           /* xt */
if(!PMflag) {if(i==j_) PM_algo}
else algo=PM_algo_en(29,algo);
fprintf_(debug,i,-1,algo," xy-29-xt");
}
else if(C(c5,c6,c17,c18)&& Su(Nx_,Nx,Ny_,Ny,Nu_,Nu,Nv_,Nv,Nw_,Nw,Nr_,Nr,Ns_,Ns,No_,No)){
jmax=en_zt();                           /* zt */
fprintf_(debug,i,-1,algo," xy-30-zt");
}
else if(i>=4 && (algo=pm_algo(algo))<0) ;
else if(C(c7,c8,c17,c18)&& Su(Nx_,Nx,Ny_,Ny,Nz_,Nz,Nv_,Nv,Nw_,Nw,Nr_,Nr,Ns_,Ns,No_,No)){
jmax=en_ut();                           /* ut */
fprintf_(debug,i,-1,algo," xy-31-ut");
}
else if(C(c9,c10,c17,c18)&& Su(Nx_,Nx,Ny_,Ny,Nz_,Nz,Nu_,Nu,Nw_,Nw,Nr_,Nr,Ns_,Ns,No_,No)){
jmax=en_vt();                           /* vt */
fprintf_(debug,i,-1,algo," xy-32-vt");
}
else if(C(c11,c12,c17,c18)&& Su(Nx_,Nx,Ny_,Ny,Nz_,Nz,Nu_,Nu,Nv_,Nv,Nr_,Nr,Ns_,Ns,No_,No)){
jmax=en_wt();                           /* wt */
fprintf_(debug,i,-1,algo," xy-33-wt");
}
else if(C(c13,c14,c17,c18)&& Su(Nx_,Nx,Ny_,Ny,Nz_,Nz,Nu_,Nu,Nv_,Nv,Nw_,Nw,Ns_,Ns,No_,No)){
jmax=en_rt();                           /* rt */
fprintf_(debug,i,-1,algo," xy-34-rt");
}
else if(C(c15,c16,c17,c18)&& Su(Nx_,Nx,Ny_,Ny,Nz_,Nz,Nu_,Nu,Nv_,Nv,Nw_,Nw,Nr_,Nr,No_,No)){
jmax=en_st();                           /* st */
fprintf_(debug,i,-1,algo," xy-35-st");
}

/* o sector */
else if(i>=4 && (algo=pm_algo(algo))<0) ;
else if(C(c3,c4,c19,c20)&& Su(Nx_,Nx,Nz_,Nz,Nu_,Nu,Nv_,Nv,Nw_,Nw,Nr_,Nr,Ns_,Ns,Nt_,Nt)){
jmax=en_yo();                           /* yo */
if(!PMflag) {if(i==i_) PM_algo}
else algo=PM_algo_en(36,algo);
fprintf_(debug,i,-1,algo," xy-36-yo");
}
else if(C(c1,c2,c19,c20)&& Su(Ny_,Ny,Nz_,Nz,Nu_,Nu,Nv_,Nv,Nw_,Nw,Nr_,Nr,Ns_,Ns,Nt_,Nt)){
jmax=en_xo();                           /* xo */
if(!PMflag) {if(i==j_) PM_algo}
else algo=PM_algo_en(37,algo);
fprintf_(debug,i,-1,algo," xy-37-xo");
}
else if(C(c5,c6,c19,c20)&& Su(Nx_,Nx,Ny_,Ny,Nu_,Nu,Nv_,Nv,Nw_,Nw,Nr_,Nr,Ns_,Ns,Nt_,Nt)){
jmax=en_zo();                           /* zo */
fprintf_(debug,i,-1,algo," xy-38-zo");
}
else if(i>=4 && (algo=pm_algo(algo))<0) ;
else if(C(c7,c8,c19,c20)&& Su(Nx_,Nx,Ny_,Ny,Nz_,Nz,Nv_,Nv,Nw_,Nw,Nr_,Nr,Ns_,Ns,Nt_,Nt)){
jmax=en_uo();                         /* uo */
fprintf_(debug,i,-1,algo," xy-39-uo");
}
else if(C(c9,c10,c19,c20)&& Su(Nx_,Nx,Ny_,Ny,Nz_,Nz,Nu_,Nu,Nw_,Nw,Nr_,Nr,Ns_,Ns,Nt_,Nt)){
jmax=en_vo();                         /* vo */
fprintf_(debug,i,-1,algo," xy-40-vo");
}
else if(C(c11,c12,c19,c20)&& Su(Nx_,Nx,Ny_,Ny,Nz_,Nz,Nu_,Nu,Nv_,Nv,Nr_,Nr,Ns_,Ns,Nt_,Nt)){
jmax=en_wo();                         /* wo */
fprintf_(debug,i,-1,algo," xy-41-wo");
}
else if(C(c13,c14,c19,c20)&& Su(Nx_,Nx,Ny_,Ny,Nz_,Nz,Nu_,Nu,Nv_,Nv,Nw_,Nw,Ns_,Ns,Nt_,Nt)){
jmax=en_ro();                         /* ro */
fprintf_(debug,i,-1,algo," xy-42-ro");
}
else if(C(c15,c16,c19,c20)&& Su(Nx_,Nx,Ny_,Ny,Nz_,Nz,Nu_,Nu,Nv_,Nv,Nw_,Nw,Nr_,Nr,Nt_,Nt)){
jmax=en_so();                         /* so */
fprintf_(debug,i,-1,algo," xy-43-so");
}
else if(C(c17,c18,c19,c20)&& Su(Nx_,Nx,Ny_,Ny,Nz_,Nz,Nu_,Nu,Nv_,Nv,Nw_,Nw,Nr_,Nr,Ns_,Ns)){
jmax=en_to();                         /* to */
fprintf_(debug,i,-1,algo," xy-44-to");
}



/* here B6 *//* change 0 into 1 */
#if 0
/* p sector */
else if(i>=4 && (algo=pm_algo(algo))<0) ;
else if(C(c3,c4,c21,c22)&& Su(Nx_,Nx,Nz_,Nz,Nu_,Nu,Nv_,Nv,Nw_,Nw,Nr_,Nr,Ns_,Ns,Nt_,Nt,No_,No)){
jmax=en_yp();                         /* yp */
if(!PMflag) {if(i==i_) PM_algo}
else algo=PM_algo_en(45,algo);
fprintf_(debug,i,-1,algo," xy-45-yp");
}
else if(C(c1,c2,c21,c22)&& Su(Ny_,Ny,Nz_,Nz,Nu_,Nu,Nv_,Nv,Nw_,Nw,Nr_,Nr,Ns_,Ns,Nt_,Nt,No_,No)){
jmax=en_xp();                         /* xp */
if(!PMflag) {if(i==j_) PM_algo}
else algo=PM_algo_en(46,algo);
fprintf_(debug,i,-1,algo," xy-46-xp");
}
else if(C(c5,c6,c21,c22)&& Su(Nx_,Nx,Ny_,Ny,Nu_,Nu,Nv_,Nv,Nw_,Nw,Nr_,Nr,Ns_,Ns,Nt_,Nt,No_,No)){
jmax=en_zp();                         /* zp */
fprintf_(debug,i,-1,algo," xy-47-zp");
}
else if(i>=4 && (algo=pm_algo(algo))<0) ;
else if(C(c7,c8,c21,c22)&& Su(Nx_,Nx,Ny_,Ny,Nz_,Nz,Nv_,Nv,Nw_,Nw,Nr_,Nr,Ns_,Ns,Nt_,Nt,No_,No)){
jmax=en_up();                         /* up */
fprintf_(debug,i,-1,algo," xy-48-up");
}
```

```
else if(C(c9,c10,c21,c22)&& Su(Nx_,Nx,Ny_,Ny,Nz_,Nz,Nu_,Nu,Nw_,Nw,Nr_,Nr,Ns_,Ns,Nt_,Nt,No_,No)){
jmax=en_vp();                          /* vp */
fprintf_(debug,i,-1,algo," xy-49-vp");
}
else if(C(c11,c12,c21,c22)&& Su(Nx_,Nx,Ny_,Ny,Nz_,Nz,Nu_,Nu,Nv_,Nv,Nr_,Nr,Ns_,Ns,Nt_,Nt,No_,No)){
jmax=en_wp();                          /* wp */
fprintf_(debug,i,-1,algo," xy-50-wp");
}
else if(C(c13,c14,c21,c22)&& Su(Nx_,Nx,Ny_,Ny,Nz_,Nz,Nu_,Nu,Nv_,Nv,Nw_,Nw,Ns_,Ns,Nt_,Nt,No_,No)){
jmax=en_rp();                          /* rp */
fprintf_(debug,i,-1,algo," xy-51-rp");
}
else if(C(c15,c16,c21,c22)&& Su(Nx_,Nx,Ny_,Ny,Nz_,Nz,Nu_,Nu,Nv_,Nv,Nw_,Nw,Nr_,Nr,Nt_,Nt,No_,No)){
jmax=en_sp();                          /* sp */
fprintf_(debug,i,-1,algo," xy-51-sp");
}
else if(C(c17,c18,c21,c22)&& Su(Nx_,Nx,Ny_,Ny,Nz_,Nz,Nu_,Nu,Nv_,Nv,Nw_,Nw,Nr_,Nr,Ns_,Ns,No_,No)){
jmax=en_tp();                          /* tp */
fprintf_(debug,i,-1,algo," xy-52-tp");
}
else if(C(c19,c20,c21,c22)&& Su(Nx_,Nx,Ny_,Ny,Nz_,Nz,Nu_,Nu,Nv_,Nv,Nw_,Nw,Nr_,Nr,Ns_,Ns,Nt_,Nt)){
jmax=en_op();                          /* op */
fprintf_(debug,i,-1,algo," xy-53-op");
}
#endif




else {printf(" ?algo/2\n");refill=-3;}
}

end:

return algo;
}/** set_ens_0 **/



void stt(int j)
{
/* here A7 *//* add p[j]=P; */
x[j]=X;y[j]=Y;z[j]=Z;u[j]=U;v[j]=V;w[j]=W;r[j]=R;s[j]=S;t[j]=T;
o[j]=O;
/* here A7 *//* add p_[j]=P_; */
x_[j]=X_;y_[j]=Y_;z_[j]=Z_;u_[j]=U_;v_[j]=V_;w_[j]=W_;r_[j]=R_;s_[j]=S_;t_[j]=T_;
o_[j]=O_;
}/** stt **/



int cag_r(long oldtime)
{
char str[32],buf[10];
int cp,i,j,n,once;
int flag_[CPMAX],flag_pp[CPMAX];
int ssize,pos,cflag,alg,algo;
int nxp,nxm,nyp,nym,nzp,nzm,nup,num,nxp_c,nxm_c,nyp_c,nym_c,nzp_c,nzm_c,nup_c,num_c;
int nvp,nvm,nwp,nwm,nrp,nrm,nsp,nsm,nvp_c,nvm_c,nwp_c,nwm_c,nrp_c,nrm_c,nsp_c,nsm_c;
/* here A3 *//* add npp,npm, npp_c,npm_c */
int ntp,ntm,nop,nom,ntp_c,ntm_c,nop_c,nom_c;
long nowtime,dlt,dlt_sum,val;
dbl fal;

if(refill==0) return 1;

cnt++;
ssize=sizeof(sss);
cp=CPMAX;
dlt_sum=0;
once=0;
tmp3=-1;

for(i=0;i<CPMAX;i++){
pcount[i]=0;
qcount[i]=0;
flag_[i]=1;
}

for(i=0;i<CPMAX;i++){
strcpy(str,"rtn[");
#if !WX
strcat(str,itoa(i,buf,10));
#else
strcat(str,gcvt(i,8,buf));
#endif
strcat(str,"].bin");

if((fp[i]=FOPEN(str,"w+b"))==NULL){
strcpy(str,"Do \"ulimit -n ");
strcat(str,gcvt(CPMAX+1000,8,buf));
strcat(str,"\".");
printf(" %s\n",str);
refill=0;
return 1;
}
}

ca=/*16*/fcolor;

set_colors();
if(CPMAX==1){
nax[0]=1;nay[0]=1;naz[0]=0;nau[0]=0;nav[0]=0;naw[0]=0;nar[0]=0;nas[0]=0;nat[0]=0;
/* here A7 *//* add nap[0]=0; */
nao[0]=0;
}
else          set_seedpoints();

i=0;
while(1){
if(flag_[i]){                         /* CP_? */
ig=i;
```

```
nx[i]=nax[i];ny[i]=nay[i];nz[i]=naz[i];nu[i]=nau[i];nv[i]=nav[i];nw[i]=naw[i];
/* here A7 */                   /* add np[i]=nap[i]; */
nr[i]=nar[i];ns[i]=nas[i];nt[i]=nat[i];no[i]=nao[i];
/* here A9 */                                           /* add np[i] */
putpixel(nx[i],ny[i],nz[i],nu[i],nv[i],nw[i],nr[i],ns[i],nt[i],no[i],acolor[i]);
pcount[i]++;
}/**if(flag_[i])**/

i++;
if(i==CPMAX) break;
}/**while(1)**/

i=0;
while(1){
if(flag_[i]){                       /* CP_? */
nx_[i]=nax[i];ny_[i]=nay[i];nz_[i]=naz[i];nu_[i]=nau[i];nv_[i]=nav[i];nw_[i]=naw[i];
/* here A7 */                      /* add np_[i]=nap[i]; */
nr_[i]=nar[i];ns_[i]=nas[i];nt_[i]=nat[i];no_[i]=nao[i];
ig=i;

if(PBC || CPMAX==1) nay[i]++;
else{
/* usable:z,u,v,w,r,s,t,o -> */
     if(_1st__==0 && _2nd__==1 && DMS>3){
if(i<=3) naz[i]++;else naz[i]--;
}
else if(_1st__==0 && _2nd__==2 && DMS>4){  /* DMS:4 -> x,y */
nau[i]++;
}
/*else if(_1st__==1 && _2nd__==0 && DMS>3){
}*/
else if(_1st__==1 && _2nd__==1 && DMS>3){
if(i<=3) naz[i]++;else naz[i]--;
}
else if(_1st__==1 && _2nd__==2 && DMS>3){
/*nau[i]++;*/
nav[i]++;
}
/* <- usable:z,u,v,w,r,s,t,o */
else if(CPMAX==2){
if(i%2==0) nax[i]++;
else if(i%2==1) nax[i]--;
}
else if(CPMAX==4){
if(i%4==0) nax[i]++;
else if(i%4==1) nax[i]--;

else if(i%4==2) nay[i]--;
else if(i%4==3) nay[i]++;
}/**else if(CPMAX)**/
else if(CPMAX==8){
/*if(i<CPMAX/2){
if(i%4==0) nax[i]++;
else if(i%4==1) nax[i]--;
else if(i%4==2) nay[i]--;
else if(i%4==3) nay[i]++;
}
else{
if(i%4==0) nay[i]++;
else if(i%4==1) nay[i]--;

else if(i%4==2) nax[i]++;
else if(i%4==3) nax[i]--;
}*/
if(i<CPMAX/2){
if(i%4==0) nay[i]++;
else if(i%4==1) nay[i]--;

else if(i%4==2) nax[i]++;
else if(i%4==3) nax[i]--;
}
else{
if(i%4==0) nax[i]++;
else if(i%4==1) nax[i]--;

else if(i%4==2) nay[i]--;
else if(i%4==3) nay[i]++;
}
}/**else if(CPMAX)**/
}/**else(PBC,CPMAX)**/

nx[i]=nax[i];ny[i]=nay[i];nz[i]=naz[i];nu[i]=nau[i];nv[i]=nav[i];nw[i]=naw[i];
/* here A7 */                   /* add np[i]=nap[i]; */
nr[i]=nar[i];ns[i]=nas[i];nt[i]=nat[i];no[i]=nao[i];

jump(i);                /* modification of S,S' */

/* here A9 */                                           /* add np[i] */
putpixel(nx[i],ny[i],nz[i],nu[i],nv[i],nw[i],nr[i],ns[i],nt[i],no[i],acolor[i]);
pcount[i]++;
}/**if(flag_[i])**/

i++;
if(i==CPMAX) break;
}/**while(1)**/

/*if(GRPH==1 && cnt==1) use_subroop();*/
#if CHK_H && CPMAX>1
check_hole(1,-1);
#endif

/***************************** while(cp) -> *****************************/

/*999*/
while(cp){
kbhit_();
if(refill<=0) break;

#if PBC
```

```
alg=0+random_(2);
#else
alg=/*4*//*2*/0+random_(/*2*//*4*//*6*/6);
PMflag=/*random_(2)*/0;
#endif

i=0;
while(1){
if(flag_[i]){                          /* CP_? */
ig=i;

#if !PBC
if(CPMAX==8 && i==CPMAX/2) {if(alg%2==0) alg++;else alg--;}
algo=alg;
#endif

/* here A7 *//* add Np=np[i]; */
Nx=nx[i];Ny=ny[i];Nz=nz[i];Nu=nu[i];Nv=nv[i];Nw=nw[i];Nr=nr[i];
Ns=ns[i];Nt=nt[i];No=no[i];
/* here A7 */  /* add Np_=np_[i]; */
Nx_=nx_[i];Ny_=ny_[i];Nz_=nz_[i];Nu_=nu_[i];Nv_=nv_[i];Nw_=nw_[i];Nr_=nr_[i];
Ns_=ns_[i];Nt_=nt_[i];No_=no_[i];
/* here A7 */    /* add npp=np[i]+1; */
nxp=nx[i]+1;nyp=ny[i]+1;nzp=nz[i]+1;nup=nu[i]+1;nvp=nv[i]+1;nwp=nw[i]+1;nrp=nr[i]+1;
nsp=ns[i]+1;ntp=nt[i]+1;nop=no[i]+1;
/* here A7 */    /* add npm=np[i]-1; */
nxm=nx[i]-1;nym=ny[i]-1;nzm=nz[i]-1;num=nu[i]-1;nvm=nv[i]-1;nwm=nw[i]-1;nrm=nr[i]-1;
nsm=ns[i]-1;ntm=nt[i]-1;nom=no[i]-1;
/* here A7 *//* add npp_c=connect_nppm(npp);npm_c=connect_nppm(npm); */
nxp_c=connect_nxpm(nxp);nxm_c=connect_nxpm(nxm);
nyp_c=connect_nypm(nyp);nym_c=connect_nypm(nym);
nzp_c=connect_nzpm(nzp);nzm_c=connect_nzpm(nzm);
nup_c=connect_nupm(nup);num_c=connect_nupm(num);
nvp_c=connect_nvpm(nvp);nvm_c=connect_nvpm(nvm);
nwp_c=connect_nwpm(nwp);nwm_c=connect_nwpm(nwm);
nrp_c=connect_nrpm(nrp);nrm_c=connect_nrpm(nrm);
nsp_c=connect_nspm(nsp);nsm_c=connect_nspm(nsm);
ntp_c=connect_ntpm(ntp);ntm_c=connect_ntpm(ntm);
nop_c=connect_nopm(nop);nom_c=connect_nopm(nom);

/* here A9 */                                                 /* add np[i] */
c1=getpixel(nxp_c,ny[i],nz[i],nu[i],nv[i],nw[i],nr[i],ns[i],nt[i],no[i]);
j=1;stt(j);
c3=getpixel(nx[i],nyp_c,nz[i],nu[i],nv[i],nw[i],nr[i],ns[i],nt[i],no[i]);
j=3;stt(j);
c2=getpixel(nxm_c,ny[i],nz[i],nu[i],nv[i],nw[i],nr[i],ns[i],nt[i],no[i]);
j=2;stt(j);
c4=getpixel(nx[i],nym_c,nz[i],nu[i],nv[i],nw[i],nr[i],ns[i],nt[i],no[i]);
j=4;stt(j);

c5=getpixel(nx[i],ny[i],nzp_c,nu[i],nv[i],nw[i],nr[i],ns[i],nt[i],no[i]);
j=5;stt(j);
c6=getpixel(nx[i],ny[i],nzm_c,nu[i],nv[i],nw[i],nr[i],ns[i],nt[i],no[i]);
j=6;stt(j);
```

```
c7=getpixel(nx[i],ny[i],nz[i],nup_c,nv[i],nw[i],nr[i],ns[i],nt[i],no[i]);
j=7;stt(j);
c8=getpixel(nx[i],ny[i],nz[i],num_c,nv[i],nw[i],nr[i],ns[i],nt[i],no[i]);
j=8;stt(j);

c9=getpixel(nx[i],ny[i],nz[i],nu[i],nvp_c,nw[i],nr[i],ns[i],nt[i],no[i]);
j=9;stt(j);
c10=getpixel(nx[i],ny[i],nz[i],nu[i],nvm_c,nw[i],nr[i],ns[i],nt[i],no[i]);
j=10;stt(j);

c11=getpixel(nx[i],ny[i],nz[i],nu[i],nv[i],nwp_c,nr[i],ns[i],nt[i],no[i]);
j=11;stt(j);
c12=getpixel(nx[i],ny[i],nz[i],nu[i],nv[i],nwm_c,nr[i],ns[i],nt[i],no[i]);
j=12;stt(j);

c13=getpixel(nx[i],ny[i],nz[i],nu[i],nv[i],nw[i],nrp_c,ns[i],nt[i],no[i]);
j=13;stt(j);
c14=getpixel(nx[i],ny[i],nz[i],nu[i],nv[i],nw[i],nrm_c,ns[i],nt[i],no[i]);
j=14;stt(j);

c15=getpixel(nx[i],ny[i],nz[i],nu[i],nv[i],nw[i],nr[i],nsp_c,nt[i],no[i]);
j=15;stt(j);
c16=getpixel(nx[i],ny[i],nz[i],nu[i],nv[i],nw[i],nr[i],nsm_c,nt[i],no[i]);
j=16;stt(j);

c17=getpixel(nx[i],ny[i],nz[i],nu[i],nv[i],nw[i],nr[i],ns[i],ntp_c,no[i]);
j=17;stt(j);
c18=getpixel(nx[i],ny[i],nz[i],nu[i],nv[i],nw[i],nr[i],ns[i],ntm_c,no[i]);
j=18;stt(j);

c19=getpixel(nx[i],ny[i],nz[i],nu[i],nv[i],nw[i],nr[i],ns[i],nt[i],nop_c);
j=19;stt(j);
c20=getpixel(nx[i],ny[i],nz[i],nu[i],nv[i],nw[i],nr[i],ns[i],nt[i],nom_c);
j=20;stt(j);

/* here A7 *//* add*/
/*c21=getpixel(nx[i],ny[i],nz[i],nu[i],nv[i],nw[i],nr[i],ns[i],nt[i],no[i],npp_c);
j=21;stt(j);
c22=getpixel(nx[i],ny[i],nz[i],nu[i],nv[i],nw[i],nr[i],ns[i],nt[i],no[i],npm_c);
j=22;stt(j);
*/




/* here A10 */  /* add ||(c21==ca)||(c22==ca) */
if((c1==ca)||(c3==ca)||(c2==ca)||(c4==ca)||(c5==ca)||(c6==ca)||
   (c7==ca)||(c8==ca)||(c9==ca)||(c10==ca)||(c11==ca)||(c12==ca)||
   (c13==ca)||(c14==ca)||(c15==ca)||(c16==ca)||(c17==ca)||(c18==ca)||
   (c19==ca)||(c20==ca)) cflag=1;
else cflag=0;

if(cflag){
/* here A7 */                /* add ss.pp=np[i]; */
ss.xx=nx[i];ss.yy=ny[i];ss.zz=nz[i];ss.uu=nu[i];ss.vv=nv[i];ss.ww=nw[i];
ss.rr=nr[i];ss.ss=ns[i];ss.tt=nt[i];ss.oo=no[i];
```

```
/* here A7 */                        /* add ss.pp_=np_[i]; */
ss.xx_=nx_[i];ss.yy_=ny_[i];ss.zz_=nz_[i];ss.uu_=nu_[i];ss.vv_=nv_[i];ss.ww_=nw_[i];
ss.rr_=nr_[i];ss.ss_=ns_[i];ss.tt_=nt_[i];ss.oo_=no_[i];
if((n=fwrite(&ss,1,ssize,fp[i]))<ssize){
printf(" fwrite? i:%d pcnt:%lld n:%n\n",i,pcount[i],n);
refill=0;
}

#if PBC
algo=set_ens_1(alg);
#else
algo=set_ens_0(i,algo);
/*printf(" algo:%d\n",algo);*/
#endif




for(j=0;j<=jmax;j++){
/* here A10 */                                      /* add  && enP[j]==Np_ */
if(enX[j]==Nx_ && enY[j]==Ny_ && enZ[j]==Nz_ && enU[j]==Nu_ && enV[j]==Nv_ &&
   enW[j]==Nw_ && enR[j]==Nr_ && enS[j]==Ns_ && enT[j]==Nt_ && enO[j]==No_) {pos=j;break;}
}
/*if(j>jmax) printf(" ?\n");*/
if(algo%2==0) CW(pos,jmax);
else          CCW(pos,jmax);
/* here A7 *//* add np[i]=P; */
nx[i]=X;ny[i]=Y;nz[i]=Z;nu[i]=U;nv[i]=V;nw[i]=W;nr[i]=R;ns[i]=S;
nt[i]=T;no[i]=O;
/* here A7 *//* add np_[i]=Np; */
nx_[i]=Nx;ny_[i]=Ny;nz_[i]=Nz;nu_[i]=Nu;nv_[i]=Nv;nw_[i]=Nw;nr_[i]=Nr;ns_[i]=Ns;
nt_[i]=Nt;no_[i]=No;

jump(i);                /* modification of S,S' */

/* here A9 */                                           /* add np[i] */
putpixel(nx[i],ny[i],nz[i],nu[i],nv[i],nw[i],nr[i],ns[i],nt[i],no[i],acolor[i]);
pcount[i]++;
#if CHK_H && CPMAX>1
check_hole(0,i);
#endif

flag_pp[i]=1;
}/**if(cflag)**/
else{
if(ftell(fp[i])==0LL) {flag_[i]=0;cp--;if(cp==0) break;}
fseek(fp[i],-ssize,SEEK_CUR);
if((n=fread(&ss,1,ssize,fp[i]))<ssize){
printf(" fread? i:%d pcnt:%lld n:%n\n",i,pcount[i],n);
refill=0;
}
/* here A7 */                /* add np[i]=ss.pp; */
nx[i]=ss.xx;ny[i]=ss.yy;nz[i]=ss.zz;nu[i]=ss.uu;nv[i]=ss.vv;nw[i]=ss.ww;
nr[i]=ss.rr;ns[i]=ss.ss;nt[i]=ss.tt;no[i]=ss.oo;
/* here A7 */                        /* add np_[i]=ss.pp_; */
nx_[i]=ss.xx_;ny_[i]=ss.yy_;nz_[i]=ss.zz_;nu_[i]=ss.uu_;nv_[i]=ss.vv_;nw_[i]=ss.ww_;
```

```
nr_[i]=ss.rr_;ns_[i]=ss.ss_;nt_[i]=ss.tt_;no_[i]=ss.oo_;
fseek(fp[i],-ssize,SEEK_CUR);

flag_pp[i]=0;
}/**else(cflag)**/
}/**if(flag_[i])**/

i++;
if(i==CPMAX) break;
}/**while(1)**/

#if 1
if(once!=-1){
time(&nowtime);
n=10*60;                              /* interval=10 [m] */

if((dlt=nowtime-oldtime)>=n){
oldtime=nowtime;
dlt_sum+=dlt;
/*if(once!=-1) */once++;

fal=(1.*dlt_sum*(sum*2-CPMAX*2)/((pcount[0]+qcount[0])*CPMAX))/60;  /* p+q => *2 */
val=dlt_sum/60;

if(once==1){                         /* 1st printf */
printf(" %ld:%.2f [dm]\n",val,fal);
tmp3=fal;
once=-1;
}
else{
printf(" %ld:%.2f [m]\n",val,fal);
}
}
}/**if(once)**/
#endif

#if !PBC && CPMAX>1
n=flag_pp[0];
i=1;
while(1){
if(flag_pp[i]!=n){
refill=-1;
printf(" refill:%d\n",refill);
break;
}

i++;if(i==CPMAX) break;
}
#endif
}/**while(cp)**/

for(i=0;i<CPMAX;i++) fclose(fp[i]);

return 0;
}/** cag_r **/
```