THE PERFORMANCE ESTIMATION OF SCENE MANAGER IN REAL-TIME SCENE GENEGATOR

Ewgenij A. Bashkov, kandidat of technical sciences
Raisa V. Malcheva, kandidat of technical sciences
Computers department,
Donetsk State Technical University,
Donetsk, Artem str., 58, 340000, Ukraine

Multichannel pipeline visual systems are used for generating realistic images at real-time rates. In such systems the scene manager has a particular aim to form a local data base for each channel. In the present paper the algorithm for multichannel scene manager is described. The performance evaluation of the special-purpose scene manager is also discussed.

1. INTRODUCTION

One of the main branches of the computer graphic systems applications is the image generators for trainer development and flight simulators. This system must generate high quality images in real time. Usually [1][2] it is constructed as specialized pipeline of the four basic subsystems:

- the ***scene manager*** - to form a local data base for each of the synthesized frame in accordance with the position of pilot, objects (features, targets, etc) and data from scene's data base;
- ***geometric subsystem*** - to process all necessary geometric transformations, clipping and calculating intensity parameters;
- ***rastrization subsystem*** - to transform the image to a raster form with shading visible surfaces at the same time;
- ***videoprocessor*** with display - to generate all necessary synchro and video signals and to display a resulting frame.

2. THE STRUCTURE OF THE IMAGE SYNTHESIS SYSTEM

In order to increase the productivity and to widen the functional capability the multichannel systems is employed in image generators. These systems consist of common scene manager and some rendering channels. Each channel as well as a system, in general, presents a three-stage pipeline: geometric subsystem, rastrization subsystem and videoprocessor/display. The image is put out by means of projection video devices. The images having generated by separate channels, are mixed to a single whole on the screen. In such multichannel systems the scene manager has a particular functional aim to form a few priority lists. Below we present our version of constructing the scene manager for a multichannel system as a special-purpose device.

3. THE COMMON STAGES OF PROCESSING OF SPECIALIZED SCENE MANAGER

The following stages were joint to built up the efficient algorithm:

- the definition of potentially visible objects;

- the depth sorting of potentially visible objects with marking the hidden objects;
- sorting out the subobjects of the hidden objects.

In accordance with the extended algorithm the scene manager has to include:

- a preliminary processing unit;
- a preliminary sorting unit;
- a subobject sorter.

The preliminary processing unit sets a "visual" flag for potentially visible objects. The preliminary sorting unit does the depth sorting and forms the priority lists and the list of the hidden objects numbers using its 3D spherical extents. The subobject sorter forms the priority subobjects list for the groups of the occluded objects.

4. SCENE MANAGER ALGORITHM

The first scene manager action begins with loading the vectors $\mathbf{Po_i}$ of the positions and orientations of the pilot and objects. On this base of that the 3 X3 rotating matrix B is calculated. Then the centers of the objects are transformed into the pilot coordinate system for preliminary processing and setting "visual" flags. To simplify the process the object's extent is analyzed, as bounded sphere with radius Ro. It is obvious, the object is located beyond the scope of visible region, if it is located behind the screen or outside the bounders of the simple viewing pyramid. The depth sorting of potential visible objects [3] is reduced, in fact, to the analysis of the distances from the pilot to the objects' centers.

The resulting scene manager algorithm for the multychannel synthesizing system by means of the priority methods is presented below.

```
loading the position  Poi & orientation
   vectors of objects Oi  and pilot;
 calculating the matrix  B;
   for  i := 1, No
   { transforming  Poi  into the pilot coordinate system;
     forming a "visual" flag (V[i]:=1) to object  Oi ;
   }
   for  i := 1, No
   {depth sorting of potentially visible objects(V[i]=1);
     forming the priority list SP[1..Nov]
                          for the objects with V[i]=1;
   }
   for  i := 1, Nov
     {
     for  j := j+1, Nov
         {if  (distance between the object SP[j] and the object SP[i]
          less than the sum of the extent radiuses?)
       then
       {  marking the objects SP[i],SP[j] as occluding each other;}
        }
      }
   for  i := 1, Nov
    {if  (the object are not marked in SP[i] as occluded?)
      then
     { forming the priority list QS[i]  for the subobjects
                                    from i-th object;}
     else
     { depth sorting of the subobjects for the group of
```

```
            the occluded objects;
        forming the priority list QS[i] for the subobjects
            from the group of the occluded objects};
     }
   if ( Nc<  the number of the groups? )
     then {mixing the priorities lists of the subobjects
         into the  groups of single neighbouring objects;}
   //the priority lists of subobjects QS[i], i=1,..,Nc are formed
   for  k := 1, Nc
   {  i := k;
     associating k-th channel with the priority of  i-th group;
   }
 while  (some QS[k] is not empty ?)
 { for  k := 1, Nc
     {   if  (is the k-th channel free?)
        {if  (is the list QS[k] not empty?)  then
          {loading k-th channel with the next subobject from QS[k];
           deleting the selected subobject from QS[k];}
        }
     }
   }
```

The sorting of priority lists of subobjects for the occluded objects, in fact, is reduced to the analysis of the distances between the viewing point and the subobjects ' centers and their processing in accordance with the next algorithm.

```
 for  ni := 1, Ni
   {calculating the distances DI[ni] from the pilot to
     the subobjects' centers of the object i;
   }
   for 1 nj := 1, Nj
   {
     calculating the distances DJ[nj] from the pilot to
     the subobjects' centers of the object j ;
   }
   ni=0; nj=0;
   for  i_s := 1, (Ni + Nj)
     {
     if ( DI[ni] >= DJ[nj] )
       then
         { QS[i_s] = SP[i][ni]; ni++ }
       else
         { QS[i_s] = SP[j][nj]; nj++ }
     }
```

As a result, this algorithm allows to form Nc disjoint lists of objects or subobjects for each of the synthesized frames in accordance with the pilots and the objects' positions.

5 SCENE MANAGER CHARACTERISTICS

To evaluate the performance of the scene manager device it is necessary to define the structure of each object and the scene on the whole and the instruction set of processing units.

Let the scene include ***No*** objects, ***Noc*** of which are occluded. Each object consists of ***Npl*** planes of ***Np*** points. Then let's define the set of basic instructions as: the exchange with host computer (R), the comparision+condition jump (C), logical instruction (O), memory read/write (L) instruction , addition instruction (A), multiplication instruction (M)? Division (D). Finally, the summary of scene manager is presented in the table below.

Table

Summary marks of scene manager

Operation	Preliminary proc.unit	Prelim. sorting unit	Subobject sorter
R	25 + 34·***No***	25 + 36·***No***	***Noc· Npl·***(3+10·***Npl***) + ***No*** ·[***Npl·***(3+7·***Np***)+1]
C	6·***No***	2·***No*** - 1	(***Noc· Npl***-1)2·***Np***
O	4 + 6·***No***	-	-
L	241+ 224·***No***	90·***No*** - 20	***Noc·***(202 ***·Npl·Np*** +63·***Npl***-144·***Np***)+ ***No***·[***Npl·***(3+7·***Np***)+1]
A	4 + 8·***No***	7·***No*** - 7	27·***Np·***(***Noc· Npl***-1)
M	16 + 9·***No***	4·***No*** - 4	48·***Np·***(***Noc· Npl***-1)
D	-	-	9·***Np·***(***Noc· Npl***-1)

CONCLUSION

The analysis shows, that the scene manager with the clock frequency 100 MHz is able to process in real time (frame rate 25 Hz) nearly 1000 objects, 60 of which are occluded.

REFERENCES

1. Harris D. Computer Graphics and Applications.- London, N.Y., Chapmann and Hall.- 1984.- 171 p.
2. Poulton J., Eyles J., Molnar S., Fuchs H. Breaking the Frame-Buffer Bottleneck with Logic-Enhanced Memories // IEEE Computer Graphics and Applications.- 1992.- 19, N 5.- pp.65-74.
3. Yan J.K. Advances in computer-generated imagery for flight simulation // IEEE Computer Graphics and Applications.- 1985.- 5, N 8.- pp.37-51.
4. Hill F.S.Jr. Computer Graphics.- N.Y., Macilan Pub. Co.- 1990.- 754 p.
5.Зингер Б.Х., Талныкин Э.А. Предварительная пространственная сортировка - основа алгоритма удаления невидимых поверхностей для систем отображения приоритетного типа // Автометрия. - 1983. N6, С.3-9.