

CZ2003

Computer Graphics and Visualization

Lab 4 Report: Implicit Solids

SSR2

U1821610C

Liew Zhi Li (Sherna)

Contents

[1 – Hierarchical Overview 2](#_Toc24942127)

[2 – FShape Overview 7](#_Toc24942128)

[3 – Comparison of LODs of Animal Object 11](#_Toc24942129)

[3.1 – Body 11](#_Toc24942130)

[3.2 – Eyes 12](#_Toc24942131)

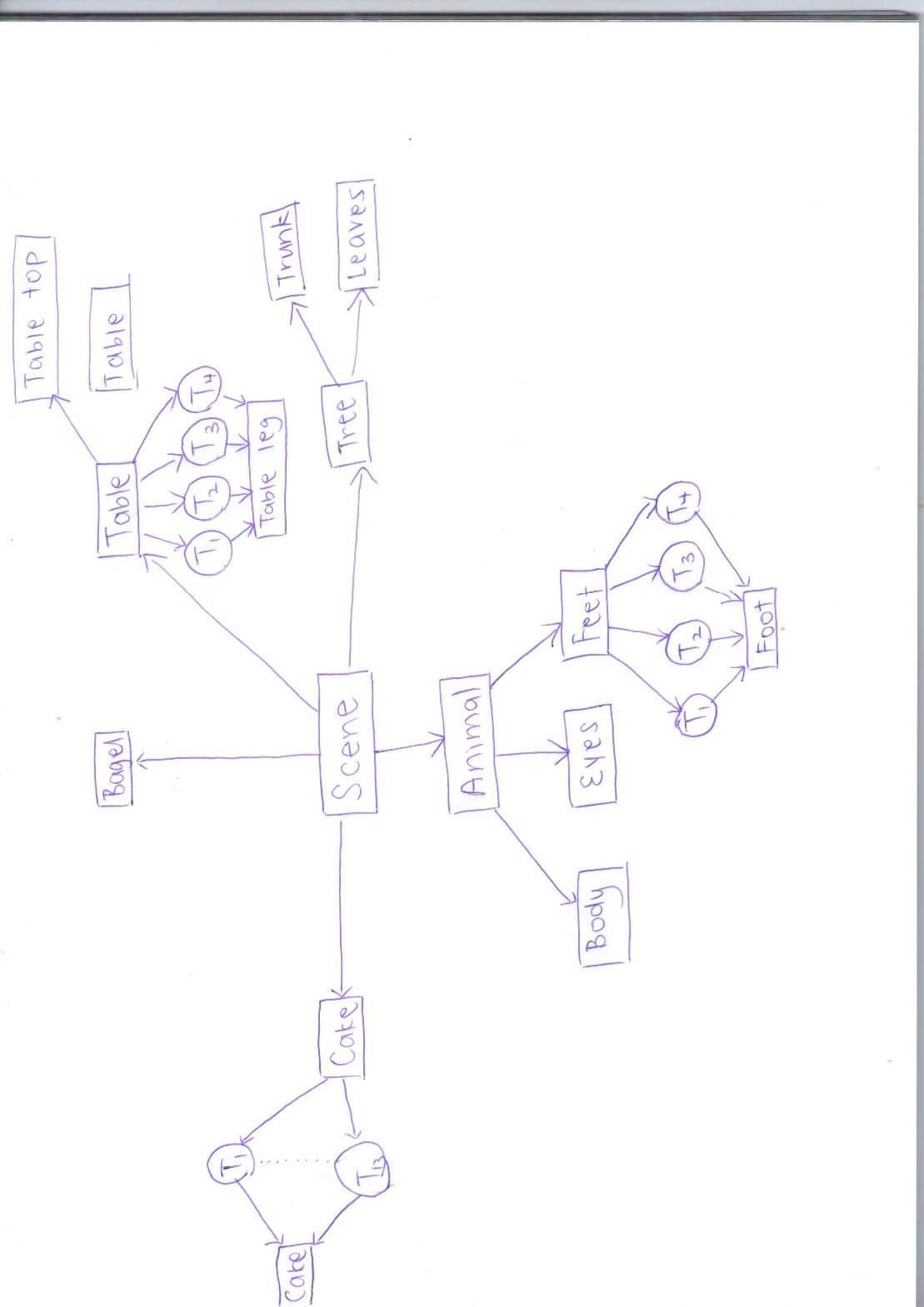
[3.3 – Foot 12](#_Toc24942132)

[4 – Code Overview 13](#_Toc24942133)

[5 – Extras 14](#_Toc24942134)

# 1 – Hierarchical Overview

The hierarchy structure allows us to model our complex FShape objects in the scene more efficiently and speed up the rendering as well. The approach I have used is logically-partitioned organization. By using a hierarchy, we can reuse an instance of an object in a scene, e.g. a foot model. If the object in the scene is an animal that has 4 feet, we can reuse by linking an instance of foot model and applying transformations to it.



The advantage of using such a structure is so that we can construct complex objects in a modular way. Some objects are that are repeated inside a higher-level object module have slight variations.

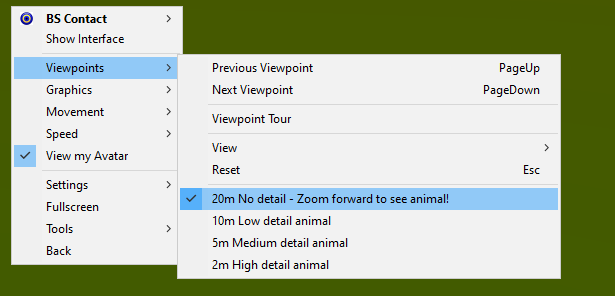
We can apply transformation to instances of an object itself. For example, a scene may contain many instances of tree objects of different sizes at different locations.

In addition, if there are changes to the geometry of an object module, we can easily update and propagate this change in the object itself. A change in the geometry of one building block module is automatically propagated to higher-level objects that uses it.

Another technique of efficient rendering is the use of level of detail (LOD).

The following are snapshots from the file “scene.wrl”. The different levels of detail of the animal has been implemented so as to maintain interactivity. We lower the level of details for objects that are far away and raise the level of details for objects nearer to the viewer.

As a viewer, you can right click and select “View my Avatar” to simulate walking in the scene. You can also right click and select the different viewpoints available.



By default, the movement has been set to “WALK” at speed 2.0 in the NavigationInfo node in the code. If you wish to toggle to different movements, you are most welcome to do so.

|  |
| --- |
| **Snapshot (No detail – Viewer is 20m away from animal)** |
| The first viewpoint is “No detail” and the viewer is located 20m away from the animal. This simulates the scenario when viewer is too far away from the animal, then we do not show the animal at all. |
| **Snapshot (Low detail – Viewer is 10m away from animal)** |
| As the viewer, move your avatar 10m forward (or alternatively select from the viewpoint menu), and you will see the low-level detail of the animal. The animal just has some rough basic shapes rendered at low resolution and a lot of its features are missing because it is not important to show the details when the viewer is still far away. Trees are rendered at low level of detail, they are simply 2 different coloured planes. |
| **Snapshot (Medium detail – Viewer is 5m away from animal)** |
| As the viewer, move your avatar 5m forward (or alternatively select from the viewpoint menu), and you will see the medium-level detail of the animal. The animal has extra shapes (nose, mouth, whiskers, spiral-shaped tail, 1 toe on its feet) rendered at a higher resolution and a lot of its features are now rendered because we can show more details when the user is nearer to the animal. Trees are rendered at medium level of detail with basic shapes using a combination of a single cone and cylinder. |
| **Snapshot (High detail – Viewer is 2m away from animal)** |
| As the viewer, move your avatar 3m forward (or alternatively select from the viewpoint menu), and you will see the high-level detail of the animal. The animal has all of its features rendered at the highest resolution and we show all of its details when the user is up close the animal. Trees are rendered at high level of detail with higher resolution and using a combination two cones instead of one cone.  Notice with directional light of colour 0.5 0.5 0.5, a grey colour is shining on the objects.    You can right click > Graphics and turn off headlight and the scene will be dark.    To the right, there is an aqua coloured vase sitting on top of the table. It is slightly transparent to simulate that it is made of glass. The table uses the image “img.jpg” and image texture mapping is performed on the surface of the table. This simulates the table is made of light-coloured wood. There is a rotating bagel inside the vase.    To the left, there is a big cake made of mooncakes sitting on top of the table. I don’t want the animal to go hungry, thus I made a cake full of moon cakes for the animal to eat. 🤣  The whole cake consists of three tiers and mini cakes placed all around itself. Displacement mapping is performed on the cake object itself to give uneven line bumps on the surface.  I hope you enjoyed and had fun exploring! 😊  If you have any feedback or suggestions, you can [email me](mailto:liew0177@e.ntu.edu.sg)! |

# 2 – FShape Overview

The complex FShape I have built is a beautiful mythical animal. It is built using union, intersection and difference of plane half-spaces, ellipsoids, spheres, cones, cylinder and other fun shapes like double torus for the ears, spiral for the tail and blobby shapes for the feet. The complex FShape for the high level of detail (LOD) renders in 2 seconds.

The following are snapshots from “animal-all-three-details.wrl”.

|  |
| --- |
| **Snapshot (Smooth) – Animal – Front** |
|  |
| **Snapshot (Flat) – Animal – Front** |
|  |
| **Snapshot (Wireframe) – Animal – Front** |
|  |
| **Snapshot (Smooth) – Animal – Back** |
|  |
| **Snapshot (Flat) – Animal – Back** |
|  |
| **Snapshot (Wireframe) – Animal – Back** |
|  |

|  |
| --- |
| **Snapshot (Smooth) – Animal – Bottom** |
|  |
| **Snapshot (Flat) – Animal – Bottom** |
|  |
| **Snapshot (Wireframe) – Animal – Bottom** |
|  |

|  |
| --- |
| **Snapshot (Smooth) – Tree** |
|  |
| **Snapshot (Flat) – Tree** |
|  |
| **Snapshot (Wireframe) – Tree** |
|  |

# 3 – Comparison of LODs of Animal Object

## 3.1 – Body

|  |  |  |
| --- | --- | --- |
| **High LOD**  **(animal-high-detail-body.wrl)** | **Medium LOD**  **(animal-medium-detail-body.wrl)** | **Low LOD**  **(animal-low-detail-body.wrl)** |
|  |  |  |
|  |  |  |
| Renders at highest resolution [70 70 70]. | Renders at medium resolution [50 50 50]. | Renders at lowest resolution [30 30 30]. |
| Varying color applied to body:  diffuseColor  "r=1;  g=abs(sin(0.3\*u\*pi)); b=abs(cos(1\*u\*pi));" | One fixed color applied to body:  diffuseColor  "r=1;  g=0.012;  b=0.98;" | |
| Have ears | No ears | |
| Have nose | | No nose |
| Have mouth | | No mouth |
| Have whiskers | | No whisker |
| Have 2 overlapped spheres for butt | Have 1 sphere for butt | No sphere for butt. Instead, torso is elongated. |
| Have cylindrical hole in butt :) | No cylindrical hole in butt | |
| Spiral tail has 3 oscillations | Spiral tail has 2 oscillations | Spiral tail replaced by cylinder tail for simplicity |

## 3.2 – Eyes

|  |  |  |
| --- | --- | --- |
| **High LOD**  **(animal-high-detail-eyes.wrl)** | **Medium LOD**  **(animal-medium-detail-eyes.wrl)** | **Low LOD**  **(animal-low-detail-eyes.wrl)** |
|  |  |  |
| Renders at highest resolution [30 30 30]. | Renders at medium resolution [10 10 10]. | Renders at lowest resolution [8 8 8]. |

## 3.3 – Foot

|  |  |  |
| --- | --- | --- |
| **High LOD**  **(animal-high-detail-foot.wrl)** | **Medium LOD**  **(animal-medium-detail-foot.wrl)** | **Low LOD**  **(animal-low-detail-foot.wrl)** |
|  |  |  |
| Renders at highest resolution [20 20 20]. | Renders at medium resolution [12 12 12]. | Renders at lowest resolution [5 5 5]. |
| Varying color applied to foot:  diffuseColor  "r=0.80;  g=abs(sin(0.1\*u\*pi));  b=1;" | One fixed color applied to foot:  diffuseColor  "r=0.80;  g=0.60;  b=0.80;" | |
| Foot has 3 toes of varying length | Foot has just 1 big toe | Foot has no toes.  Just a circular blobby shape for simplicity |

# 4 – Code Overview

We only need to know these implicit inequalities formulas:

Planes (e.g.):

Solid sphere:

Solid Ellipsoid:

Circle (Cylinder oriented along z-axis):

Circle (Cylinder oriented along x-axis):

Cone:

I have taken extra care to ensure tight fit of bounding box on the object, as well as to verify that the colour does not exceed the range [0 1].

|  |  |
| --- | --- |
| **Snapshot (Smooth)** | **Notes** |
|  | This is a snapshot of “animal-high-detail-body.wrl”.  A tight bounding box has been defined for a tight fit for the complex FShape.  bboxCenter 0 0 -0.15  bboxSize 1.85 1.9 1.7  resolution [70 70 70]  diffuseColor  "r=1; g=abs(sin(0.3\*u\*pi)); b=abs(cos(1\*u\*pi));" |
|  | This is a snapshot of “animal-high-detail-foot.wrl”.  A tight bounding box has been defined for a tight fit for the blobby shape.  bboxCenter 0 0 0.8  bboxSize 4 1.74 5.3  resolution[20 20 20]  diffuseColor "r=0.80; g=abs(sin(0.1\*u\*pi)); b=1;"  Since cos is from -1 to 1, we need to use absolute function abs() so that it will be from 0 to 1. |
|  | This is a snapshot of “animal-high-detail-eyes.wrl”.  A tight bounding box has been defined for a tight fit for the eyes.  bboxCenter 0 0 0.8  bboxSize 4 1.74 5.3  resolution [30 30 30]  diffuseColor "r=0; g=1; b=abs(cos(u\*pi));" |

# 5 – Extras

I have explored other functions to create other fun and interesting shapes.

|  |  |
| --- | --- |
| **Snapshot (Smooth)** | **Snapshot (Wireframe)** |
|  |  |
| **Notes** | |
| This is a snapshot of “object-vase.wrl”.  A tight bounding box has been defined for a tight fit for the vase object.  Function definition:  shape1 = 1-pow(x,2)-cos(y)\*sin(y)-pow(z,2);  plane = y+2;  vase = min(shape1, plane);  bboxCenter 0 0 0  bboxSize 2.5 4 2.5  resolution [20 20 20]  diffuseColor "r=0; g=0.84; b=1;"  shininess "0.16"  transparency "0.4" | |

|  |  |
| --- | --- |
| **Snapshot (Smooth)** | **Snapshot (Wireframe)** |
|  |  |
| **Notes** | |
| This is a snapshot of “object-table-full.wrl”. It is made up of the table top defined in “object-table-top.wrl” and four instances of the table leg defined in “object-table-leg.wrl”. Transformation has been applied to the four instances of table leg to move them to underneath of the table at the four corners. Image mapping uses the image file “img.jpg”. A tight bounding box has been defined a tight fit for the two different objects.  Table top:  p1=5-x;  p2=0.5-y;  p3=3-z;  p4=x+5;  p5=y+0.5;  p6=z+3;  m1 = min(p1, p2);  m2 = min(m1, p3);  m3 = min(m2, p4);  m4 = min(m3, p5);  m5 = min(m4, p6);  bboxCenter 0 0 0  bboxSize 10 1 6.5  resolution [20 20 20]  Table leg:  shape = 1-pow(x,2)-abs(cos(y)\*sin(y))-pow(z,2);  plane = y+5;  table\_leg = min(shape, plane);  bboxCenter 0 0 0  bboxSize 2.5 10 2.5  resolution [40 40 40] | |

|  |  |
| --- | --- |
| **Snapshot (Smooth)** | **Snapshot (Wireframe)** |
|  |  |
| **Notes** | |
| This is a snapshot of “object-bagel.wrl”. It is actually a torus that rotates about an axis from origin to (1,0,1). Function defined here is:  x= x1\*0.5\*(cos(t\*2\*pi)+1) - y1\*0.5\*sqrt(2)\*sin(t\*2\*pi) + z1\*0.5\*(1-cos(t\*2\*pi));  y= x1\*0.5\*sqrt(2)\*sin(t\*2\*pi) + y1\*cos(t\*2\*pi) - z1\*0.5\*sqrt(2)\*sin(t\*2\*pi);  z= x1\*0.5\*(1-cos(t\*2\*pi)) + y1\*0.5\*sqrt(2)\*sin(t\*2\*pi) + z1\*0.5\*(1+cos(t\*2\*pi));  bagel=(sqrt(x^2+z^2)-0.4)^2+y^2-0.2^2;  Specular colour of 1 1 1 has been applied and shininess of 0.5.  resolution [20 20 20]  bboxCenter 0 0 0  bboxSize 1.2 1.2 1.2 | |

|  |  |
| --- | --- |
| **Snapshot (Smooth)** | **Snapshot (Wireframe)** |
|  |  |
| **Notes** | |
| This is a snapshot of “object-cake-full.wrl”. It is a big cake made of many varying sizes of moon cakes. It consists of several instances of cake objects of different colours:  “object-cake-tier-green.wrl”  “object-cake-tier-pink.wrl”  “object-cake-tier-purple.wrl”  “object-cake-tier-yellow.wrl”  Transformation, rotation and scaling has been applied to the instances of cake object to arrange them neatly as shown above. | |
| **Snapshot (Smooth)** | **Snapshot (Wireframe)** |
|  |  |
| **Notes** | |
| This is a snapshot of “object-cake-tier-green.wrl”. Function defined here is:  cylinder = 1^2-x^2-(z^2);  p1=-y+0.3;  p2=y+0.3;  m1=min(cylinder,p1);  cake\_tier=min(m1,p2);  diffuseColor "r=0.8; g=1; b=0.8;"  Displacement mapping applied is:  texture3D FTexture3D {  definition "x=0.05\*sin(20\*pi\*u)\*sin(20\*pi\*u);  y=0.05\*sin(20\*pi\*u)\*sin(20\*pi\*u);  z=0;"  type "displacement"  } | |