3D Scene Graph

Introduction

A scene graph, an important data structure and consisting of nodes, is commomly used in building the 3D models and it based on the oriented-object program concept which regard every node as an object. In the report, it would be around the scene graph to show how the assignment could be finished. The effect of each node’s transformation and the APIs for the scene graph would be mentioned. Also, the advance approach of using scene graph would be research part of the report.

Scene Graphs for Geometry and Transformation

In the assignment, the scene graphs had been broadly used. As the image 1 showing, it could be dived into six parts which are the desk, the lamp, the pen container, the books, the showing space and the spotlight source. When creating the model in the MyEventListener.java, it only needs to build these six independent objects. (Code showing in image 2)



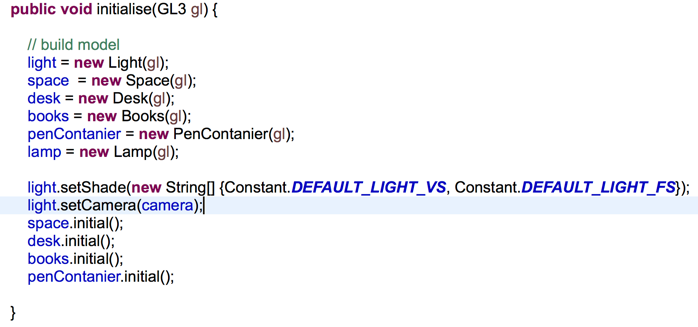


Image 1. The Scene Image 2. The Building Code

Except the spotlight, building the model of each part needs to build a scene graph at beginning.

Firstly, the space consists of 10 planes with 4 cubes after transforming. The scene graph is complex and would be showing in below image 3. (“T” for Translate, “S” for Scale and “R” for Rotate) If only render the space model, it would be look like below image 4.

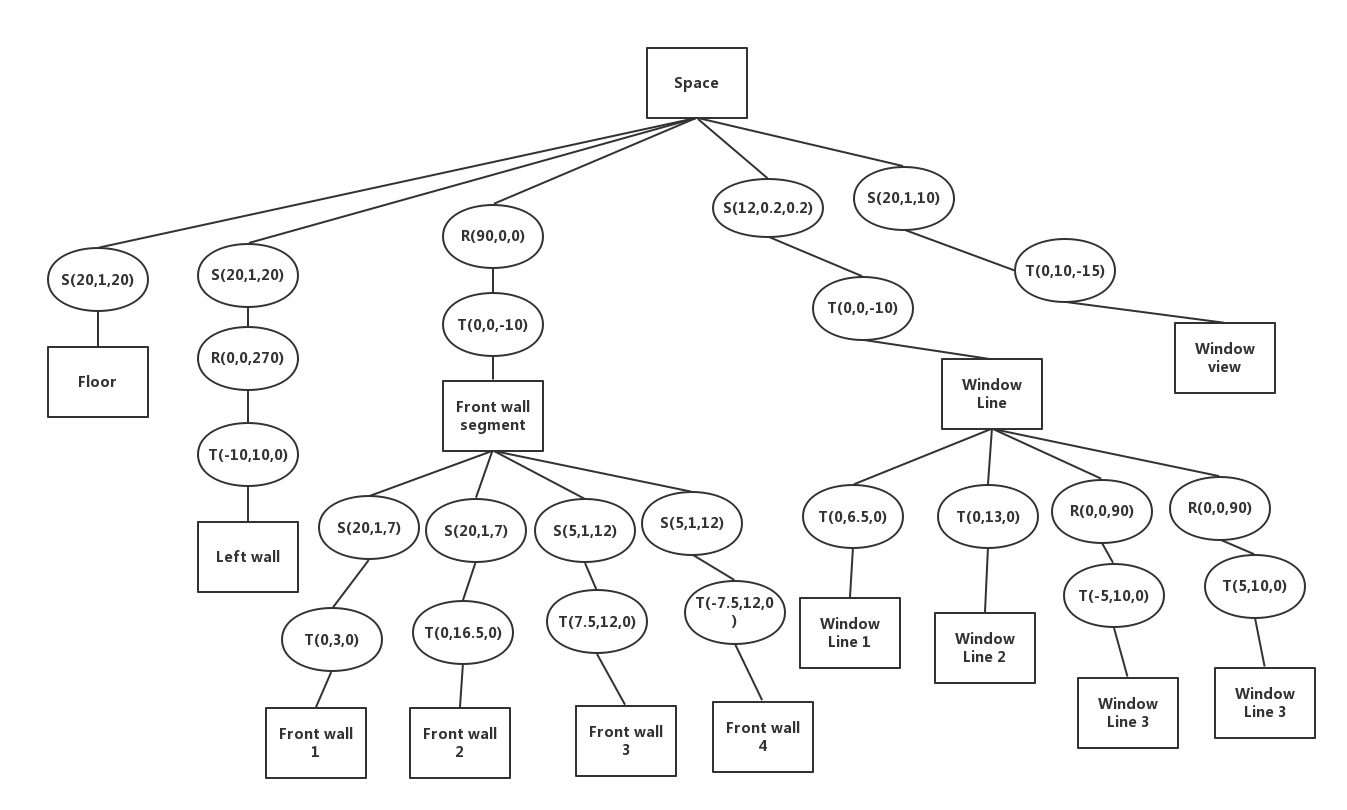


Image 3. Space Scene Graph



Image 4. The Space

Secondly, the desk consists of a platform and four legs and all of the component are transforming from the basis object which is the cube. The scene graph would be shown in below image 5. If only render the space model, it would be look like below image 6.

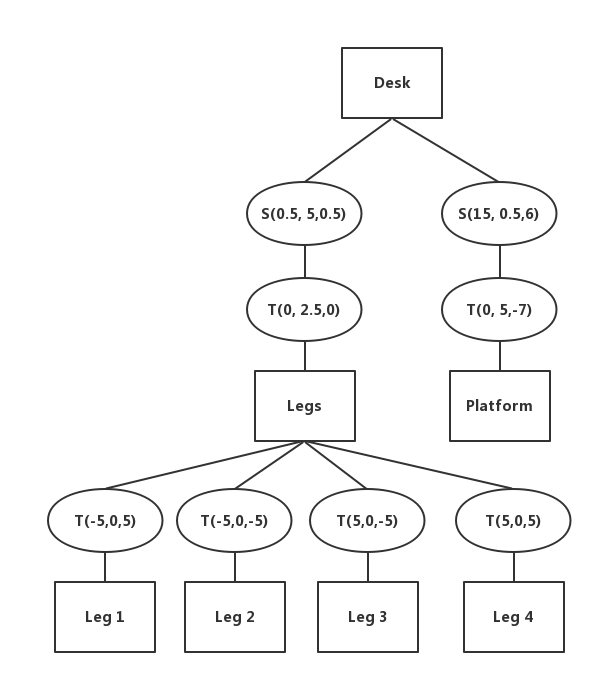
 

Image 5. The Desk Scene Graph Image 6. The Desk

Thirdly, there some stuff had been placed on the desktop. The pen container and the pen are whole object. Same to the books, all books are regarded as one object. The scene graph of these two objects could be represented in one image (Image 7), because it is not complex. The render scene of these stuff could be showing in Image 8.

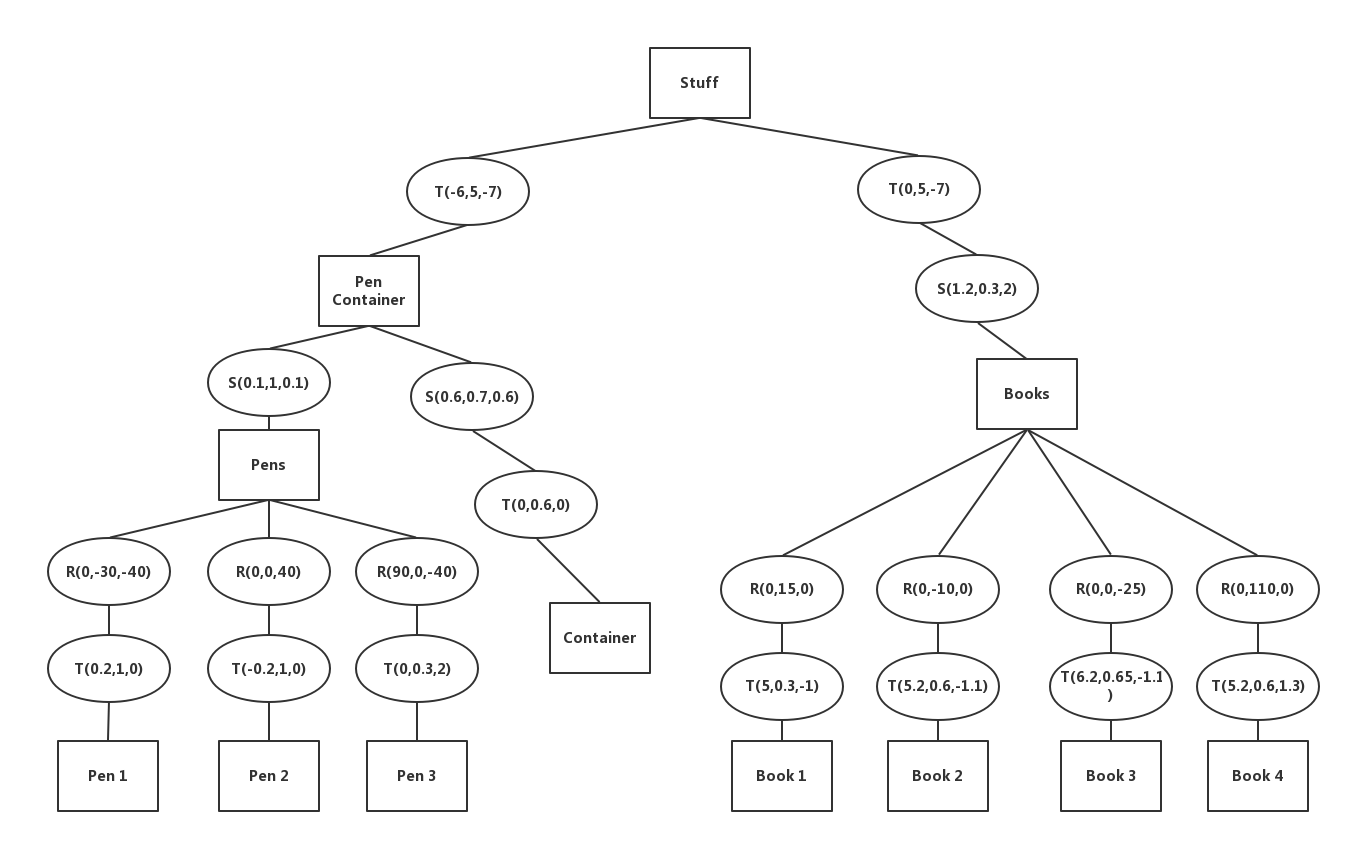


Image 7. The Stuff Scene Graph



Image 8. The Stuff on The Desktop

Finally, the lamp is the most complex model in this project, because the animation would be applied on the lamp and the lamp should be flexible enough so that could do transform easily. The lamp contains a few basis models which are the cylinder, the cone, the sphere and the hemisphere. The scene graph of lamp is in the below image 9. In the image 9, the lamp scene graph, would not specify some parameters in transformation because these parameters are to control the pose when the lamp move and they would be represented in x, y or z. Also, only the lamp rendering scene would show in the image 10.

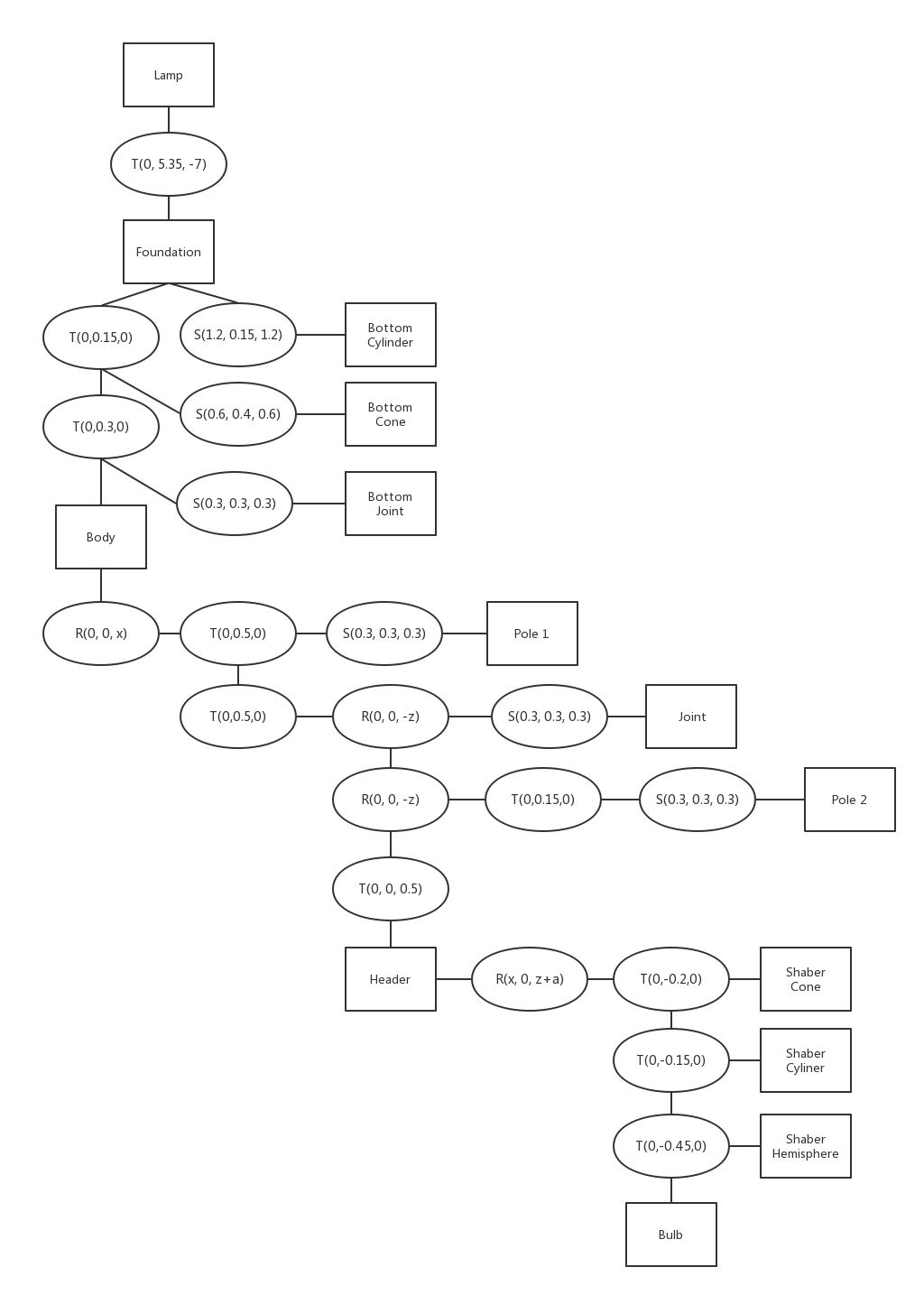


Image 9. The lamp Scene Graph



Image 10. The Lamp

According to the figures shown in above, it could be noticed that is all the children nodes’ transformation is based on the last node’s transformation so that if one node has been changed, then the nodes in following would be changed as well. So, that could connect the component nodes as a whole object and move simultaneously.

Scene Graph and APIs

In the implementation, there is a basis class named “BaseNode” which also is a superclass and inhered by component nodes such as pole, bulb, joint, etc. But actually these component nodes are geometry nodes, because these nodes had been created based on the basis geometry such as the cylinder, sphere, cube etc. The class “NodesContainer”, also inhere the class “BaseNode”, is mainly used to collect the component nodes and constitute the higher lever components nodes. Taking the lamp as an example, there are three “NodesContainer”s which are separately “Foundation”, “BodyPole” and “Header”. The lamp consists of these three containers and the containers represent the different parts of structure of the lamp. In each container, it contains the basic component nodes such as “BottomCone”, “JointNode”, “PoleNode”, etc. There is a figure which could show the relationship among “BaseNode”, “NodesContainer” and parts of other geometry nodes. Seeing in following image 11.

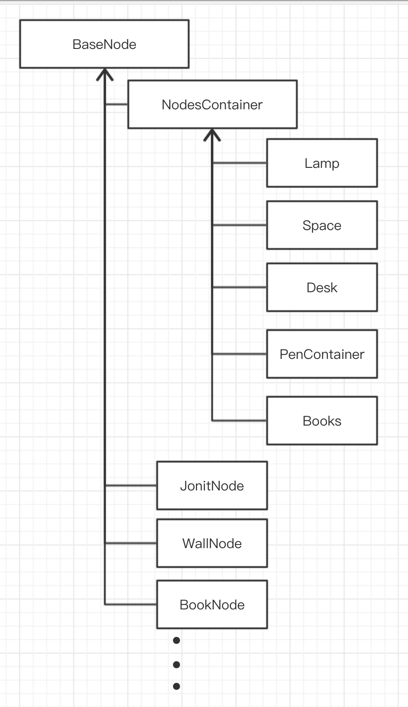


Image 11. Inhered Diagram

Expending the class “BaseNode”, some basis attributes and functions contained inside and the following image 12 would show the inside construction.



Image 12 Class BaseNode

The important functions in “BaseNode” are “addChild()”, “update()” and “render()”. In the program, every node could be the parent node of a new node which means that could add the any new node as the children nodes. Then every node has its own state when call “render()” function, all the transformation matrix would be calculated and pass to the children nodes. The transformation matrix updating is from calling the function “update()”.

Advance use of Scene Graphs

Conclusion