FACULTY OF COMPUTING
SEMESTER 2/20232024

SECV2213-02 (FUNDAMENTAL OF COMPUTER GRAPHICS)

Assignment 3

Group 3

1. Denies Wong Ke Ying	A22EC0047
2. Lim Xiao Xuan	A22EC0071
3. Pang Zhi Xuan	A22SC0466

INSTRUCTOR: Ts. Dr Goh Eg Su

Table of Content

Introduction

The complex field of computer graphics provides approaches to create 3D models and animations that are interactive. A basic concept which is hierarchical modeling enables the parent-child relationship of large objects to be built from simpler components. This makes modeling easier and allows for flexible animations. In this assignment, we use an animation project which involves penguins as a case study to explore hierarchical modeling and 3D transformations in OpenGL. With the use of OpenGL, an effective graphics API, modeling and transformations can be managed effectively, resulting in a hierarchical structure where each component of the penguin is related and manageable separately.

By building a penguin model, the project creates a root node for the body and child nodes for the head, wings, and feet. In order to animate the penguin with movements like flapping wings, swinging feet, and turning the head, each component goes through 3D transformations including translation, rotation, and scaling. We save and restore transformation states using OpenGL's matrix stack operations which are glPushMatrix and glPopMatrix, ensuring that changes to one component do not impact others. This study shows the efficiency of hierarchical modeling and 3D transformations in OpenGL by presenting the hierarchical structure of the penguin model, the specific transformations used, and the animation logic.

Overview

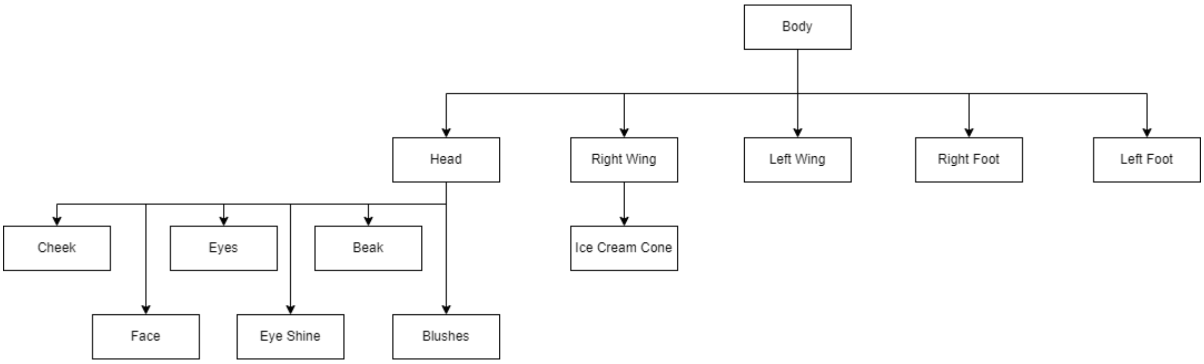
This assignment uses 3D transformations and hierarchical modeling with OpenGL to build an animated penguin model. The root node is the penguin's body, which is made up of pieces arranged hierarchically. Dynamic movements like wing flapping and head rotation are made possible by applying OpenGL's transformation features, such as translation and rotation. Realistic animation is provided by applying modifications to individual parts independently of other components through the application of OpenGL's matrix stack functions. This report offers a brief overview of the project's execution, highlighting the way that 3D transformations and hierarchical modeling were used to bring the penguin model to reality.

Design and Hierarchical Structure & Explanation of the 3D transformations

An essential component of the penguin model's development and animation is its hierarchical design. The penguin's body, which serves as the root node, is at the top of the hierarchy. The head, wings, and feet of the penguin are among the other elements that are connected as child nodes (1st level parents) to this root node. The cheeks, face, eyes, eye shine, beak and blushes as child nodes for head. For the right wing, it has an ice cream cone as its child node. A flexible method for modeling and animation is made possible by this hierarchical organization, which enables the independent adjustment of any component while maintaining the overall relationship.

Transformations extend from parent nodes to their corresponding children nodes in the hierarchical structure. For instance, the direction of the penguin's head and wings is impacted by rotations performed to its body. The body is the root node and serves as the pivot point for the entire penguin. Rotation around the y-axis is controlled by the angle variable. The head is a child of the body and rotates around its own pivot point, controlled by the headAngle variable, within a range of -90 to 90 degrees. All child nodes of the head move with it, maintaining their relative positions. The cheeks, face, eyes, eye shine, beak, and blushes move along with the head, maintaining their relative positions. The right wing is a child of the body and its movement is controlled by the rightWingAngle variable, with rightWingUp determining the direction. The attached ice cream cone moves with the right wing, maintaining its relative position. . The right wing's movement is automated to simulate natural flapping. Both feet are children of the body, with their movements controlled by leftFootSwingAngle and rightFootSwingAngle, respectively, and directional controls managed by leftFootSwingForward and rightFootSwingForward. The feet movement is automated to create a natural walking motion.

As a result of the hierarchical relationship, coordinated movements and animations are produced when modifications cascade down the model's structure. This hierarchical approach improves the flexibility and reality of the animation while additionally making the modeling process simpler. With this structure, we are able to effectively control the intricacy of the penguin model and produce smooth movements that faithfully resemble those of an actual penguin. All things considered, the penguin model's hierarchical structure provides a solid foundation for the smooth implementation of dynamic animations and 3D transformations.



Hierarchical Diagram

Implementation Output

Link of Video:

https://drive.google.com/file/d/1sFIE1K9WUC_jZM-rPzgJNXNihBDV9X9U/view?usp=drive_link

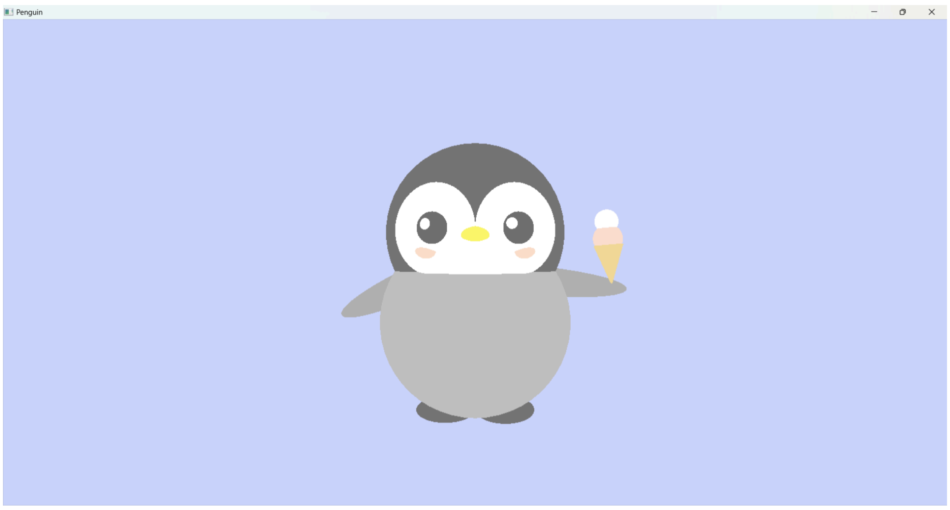


Figure 1: The output of the code

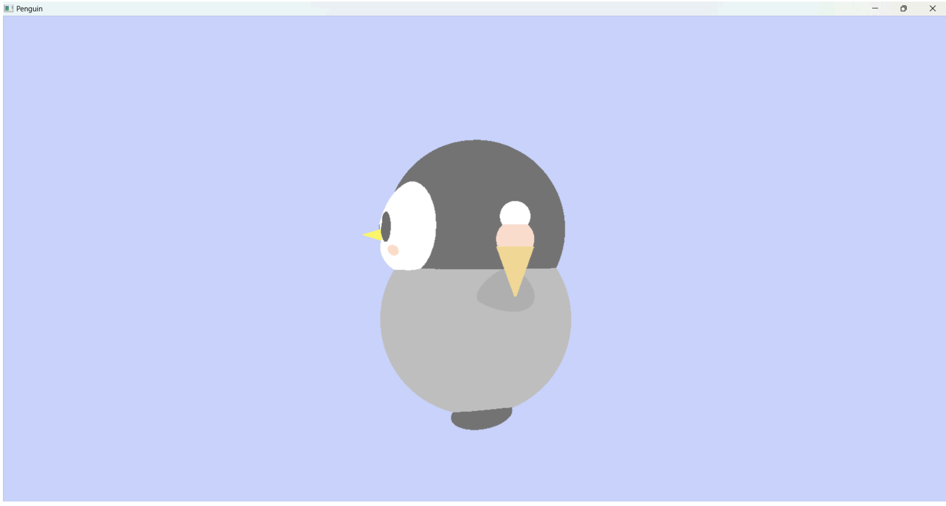


Figure 2: Rotate to the left when the 'a' key or 'A' key is pressed

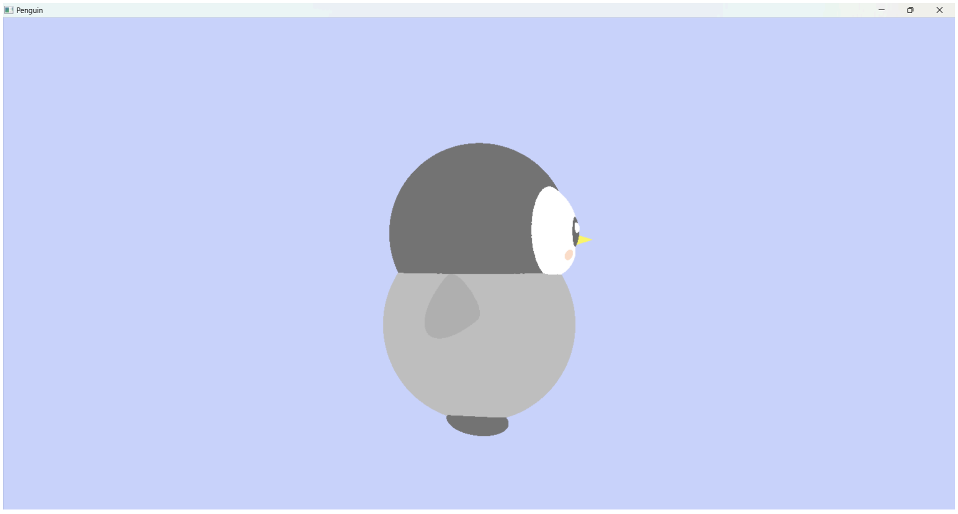


Figure 3: Rotate to the right when the 'd' key or 'D' key is pressed

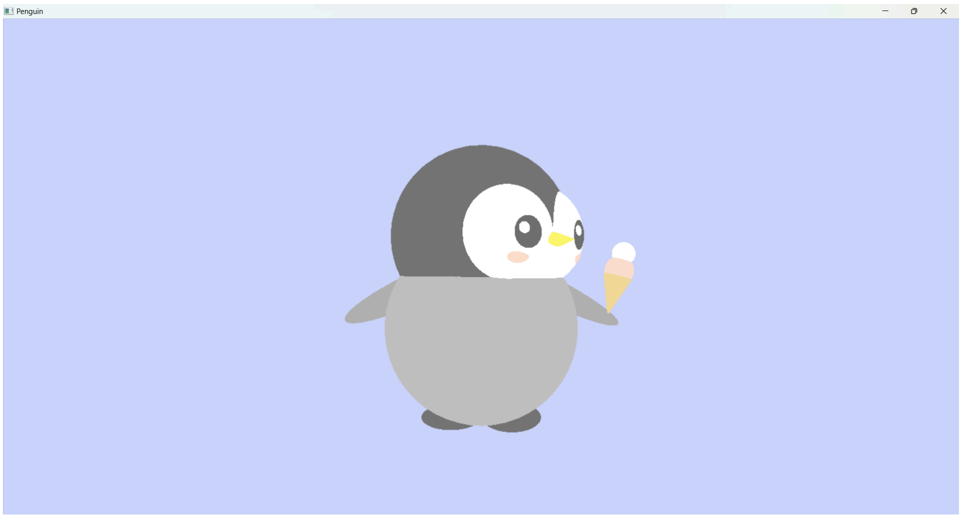


Figure 4: Rotate the head to the right when the 'h' key or 'H' key is pressed

Figure 5: Rotate the head to the left when the 'j' key or 'J' key is pressed

Conclusion

In this assignment, we developed an interactive and dynamic penguin animation to explore the concepts of hierarchical modeling and 3D transformations in OpenGL. The penguin model could be disassembled and reassembled with each component undamaged since it was arranged into a well defined hierarchy. With the use of this system, complex movements like wing flapping, head rotation, and foot swinging could be generated, providing the penguin a realistic appearance.

Retaining the hierarchical integrity of the model required the use of OpenGL's matrix stack operations, especially glPushMatrix and glPopMatrix. With the help of these functions, we were able to locally modify each component without having an impact on the whole structure. We were able to show how hierarchical modeling works well for organizing and animating complex systems with this assignment. The ideas and methods discussed in this research can be used for a variety of additional computer graphics modeling and animation jobs, demonstrating the flexibility and effectiveness of OpenGL in producing complex and dynamic 3D animations.