ΠΡΩΤΗ ΠΡΟΓΡΑΜΜΑΤΙΣΤΙΚΗ ΑΣΚΗΣΗ 1-Γ

Γραφικά Υπολογιστών και Συστήματα Αλληλεπίδρασης

Ακαδημαϊκό Έτος 2021-2022

Σίντος Δημήτριος Α.Μ:4012

Ματαράγκας Δημήτριος Α.Μ:3272

Το πρόγραμμα μας είναι βασισμένο πάνω στα παράδειγμα που μας δόθηκαν στα εργαστήρια του μαθήματος και στην άσκηση 1-Β. Το περιβάλλον που το υλοποιήσαμε και το τρέξαμε είναι σε Visual Studio για Windows 10.

Ιδιαιτερότητες αλληλεπίδρασης με τον χρήστη:

- Η κάμερα κινείται με τα πλήκτρα, όπως περιγράφονται στην εκφώνηση. Επιτρέπουμε και το παρατεταμένο πάτημα των πλήκτρων για μια πιο ομαλή αλληλεπίδραση με τον χρήστη.
- Όταν ο χρήστης πατάει το *space bar* εκτοξεύει έναν μετεωρίτη. Άμα δεν πετύχει τον πλανήτη μπορεί να ξανά εκτοξεύσει κι άλλο.

Αρχιτεκτονική του προγράμματος:

Τα αρχεία που πραγματοποιούν την λειτουργία του προγράμματος είναι:

- Main.cpp
- SimpleFragmetShader.fragmentshader
- SimpleVertexShader.vertexshader

~Να σημειώσουμε ότι τα περισσότερα conventions της γλώσσας C++ δεν τηρούνται, μιας και αυτή είναι η πρώτη μας επαφή με την γλώσσα.

Το πρόγραμμα μας ακολουθεί τεχνική functional programming και όχι αντικειμενοστραφή προγραμματισμού. Πράγμα που δεν το κάνει εύκολα επεκτάσιμο και δυσανάγνωστο προς τον αναγνώστη.

Εξηγήσεις για ιδιαίτερα τμήματα του προγράμματος:

Camera: αλλάξαμε την κάμερα από αυτή που είχαμε στην 1-Β, ώστε να κάνει σωστά την περιστροφή γύρο από τους άξονες και την κίνηση προς το κέντρο του κύκλου. Δημιουργήσαμε μια συνάρτηση *getViewMatrix()* η οποία μας επιστρέφει τον πίνακα της κάμερας. Ο συγκεκριμένος πίνακας είναι κοινός για τον υπολογισμό των πινάκων MVP (ήλιος), MVP1 (πλανήτης), MVP2 (μετεωρίτης).

Textures: Για την φόρτωση των εικόνων χρησιμοποιήσαμε την δημόσια βιβλιοθήκη stb_image καλώντας την stbi_load() για εύκολη προσπέλαση εικόνων από το δίσκο.

Objects: Για την φόρτωση των αρχείων .obj χρησιμοποιήσαμε την συνάρτησης loadObj() η οποία στηρίχθηκε στην σελίδα http://www.opengl-tutorial.org/beginners-tutorials/tutorial-7-model-loading/.

Πλανήτης: ώστε να έχει κέντρο το σημείο Β(25,0,0) χρησιμοποιήσαμε την συνάρτηση *translate()* της βιβλιοθήκης glm. Για την περιστροφική κίνηση γύρο από τον ήλιο χρησιμοποιώντας την *translate()* με διάνυσμα μετατόπισης το vec3(u,n,0). Υπολογίσαμε το u με το συνημίτονο μιας γωνίας (την οποία αυξάναμε όσο υπήρχε ο πλανήτης) επί την απόσταση του κέντρου του πλανήτη από το κέντρο του κύκλου. Αντίστοιχα για το n με το ημίτονο της ίδιας γωνίας.

Μετεωρίτης: για να κινείται από την θέση του παρατηρητή έπρεπε να βρούμε τις συντεταγμένες της κάμερας. Για αυτό τον σκοπό χρησιμοποιήσαμε την *ExtractCameraPos* η οποία στηρίχθηκε στην σελίδα https://community.khronos.org/t/extracting-camera-position-from-a-modelview-matrix/68031?fbclid=IwAR00EtVOcQEgVAsxv5UON9vJNcGT3ZOiklHDnYuuCy8f5Q5u64islEX-XFA. Μειώνοντας αυτές τις συντεταγμένες ομοιόμορφα υπολογίζαμε το διάνυσμα μετακίνησης και χρησιμοποιώντας την *translate* ο μετεωρίτης μετακινείται στην ευθεία που ενώνει το σημείο παρατήρησης με το κέντρο της σφαίρας. Με την χρήση της συνάρτησης *distance* της *glm* υπολογίζεται η απόσταση του κέντρου του μετεωρίτη με το κέντρο του ήλιου όταν η απόσταση γίνει μικρότερη από 17 ο μετεωρίτης χάνεται. Αντίστοιχα για την σύγκρουση με τον πλανήτη όταν η απόσταση του κέντρου του πλανήτη με το κέντρο του μετεωρίτη γίνει μικρότερη από 7 με την χρήση ενός flag (crash) σταματάμε την απεικόνιση του πλανήτη και του μετεωρίτη.

BONUS:

β. Για την κίνηση του πλανήτη γύρο από τον εαυτό του χρησιμοποιήσαμε την *rotate* της *glm*.

ε. Για την αυξομείωση της ταχύτητας περιστροφής του πλανήτη προσθέσαμε τα πλήκτρα <u> και <p> στην συνάρτηση key_callback(έχει γίνει αναφορά στις ασκήσεις 1-Α, 1-Β) και αυξάνοντας μια global μεταβλητή (planetSpeed) αυξάνεται και η γωνία περιστροφής.