НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
імені ІГОРЯ СІКОРСЬКОГО»
Теплоенергетичний факультет
Кафедра автоматизації проектування енергетичних процесів і систем

До захисту допущено:
Завідувач кафедри
_____ Олександр Коваль
«___»_____2021 р.

Дипломна робота

на здобуття ступеня бакалавра

спеціальності 122 «Комп'ютерні науки»

за освітньо-професійною програмою «Комп'ютерний моніторинг та геометричне моделювання процесів та систем»

на тему: «Розробка кольорової стереобази даних для візуалізації різнокольорових анагліфічних каркасних зображень в задачах 3-Д друку»

Виконав:
студент IV курсу, групи ТР-72 _____
Киселевич Владислав Юрійович

Керівник:
д.т.н. професор кафедри АПЕПС,
Груць Юрій Миколайович _____

Рецензент: _____

Засвідчую, що у цій дипломній роботі немає запозичень з праць інших авторів без відповідних посилань.
Студент _____

Київ – 2021 року

Національний технічний університет України
“Київський політехнічний інститут імені Ігоря Сікорського”

Факультет теплоенергетичний

Кафедра автоматизації проектування енергетичних процесів і систем

Рівень вищої освіти перший рівень

спеціальності 122 «Комп’ютерні науки»

освітня програма «Комп’ютерний моніторинг та геометричне моделювання процесів та систем»

ЗАТВЕРДЖУЮ
Завідувач кафедри
_____ Олександр Коваль
(підпис)
”___”_____2021р.

ЗАВДАННЯ
на дипломну роботу студенту
Киселевичу Владиславу Юрійовичу
(прізвище, ім’я, по батькові)

1. Тема роботи: Розробка кольорової стереобази даних для візуалізації різнокольорових анагліфічних каркасних зображень в задачах 3-Д друку
керівник роботи Груць Юрій Миколайович, д.т.н.,професор кафедри АПЕПС
(прізвище, ім’я, по батькові науковий ступінь, вчене звання)

затверджена наказом вищого навчального закладу від травня 2021р. № **1267-с**

2. Строк подання студентом роботи _____
3. Вихідні дані до роботи
мова програмування C, C++, C#, .NET
середовище розробки VisualStudio
4.Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити)
Розширити існуючий програмний продукт на кафедрі розробивши програму з використанням різноматнітних кольорових спектрів для візуалізації анагліфічних каркасних зображень. _____
_____
5. Перелік ілюстративного матеріалу
схеми архітектури програмного продукту, знімки інтерфейсу, знімки структури проекту

6. Дата видачі завдання ”10” жовтня 2021 р.

КАЛЕНДАРНИЙ ПЛАН

/п	Назва етапів виконання дипломної роботи	Термін виконання етапів роботи	Примітки
.	Затвердження теми роботи	6.04.20р.	
.	Вивчення та аналіз задачі	12.04.21р. – 16.04.21р	
.	Розробка архітектури та загальної структури системи	17.04.21р. – 22.04.21р.	
.	Розробка структур окремих підсистем	23.04.21р – 25.04.21р.	
.	Програмна реалізація системи	26.04.21р – 08.05.21р	
.	Оформлення пояснювальної записки	09.05.21р – 20.05.2021	
.	Захист програмного продукту	11.05.2021р.	
.	Передзахист	25.05.2021р.	
.	Захист	15.06.2021р.	

Студент _____ __Киселевич В.Ю._____
(підпис) (прізвище та ініціали,)

Керівник роботи _____ __Груць Ю.М._____
(підпис) (прізвище та ініціали,)

АНОТАЦІЯ

Киселевич Владислав Юрійович

Тема: Розробка кольорової стереобази даних для візуалізації різнокольорових анагліфічних каркасних зображень в задачах 3-Д друку

Спеціальність: 122 «Комп'ютерні науки»

Установа: Національний технічний університет України «Київський політехнічний інститут ім. Ігоря Сікорського», 2021 р.

Бакалаврська робота: 60с., вступ, 5 розділів, висновок, 9 джерел, 41 рисунків

Актуальність теми. На сьогоднішній день стереозображення відіграють велику роль в житті людей. Кожен день люди ходять в кіно, грають в комп'ютерні ігри незамислючись про те, як вони переглядають стерео 3-Д зображення. Одним із глобальних використань, окрім використання в хімії для детальної візуалізації молекул, растрові 3-Д стереозображення передаються для дослідження поверхонь інших планет. Для того щоб працювати з такими зображеннями використовуються вже існуючі методи сепарації стереозображень. Моєю задачею було програмування додатку, на якому будуть розроблені каскадні стереозображення для демонстарції роботи анагліфічного методу.

Метою дослідження. Метою дослідження є розробка розширення вже існуючого програмного продукту для демонстрації роботи анагліфічного методу з використанням різноманітних існуючих кольорових спектрів.

Об'єктом дослідження є Кольорові спектри у використанні анагліфічного методу

Предметом дослідження є Додаток візуалізації каркасних зображень

Результати роботи: Розроблено програмний продукт з каркасними зображеннями, інтерфейсом для вибірки кольорових спектрів

Ключові слова: АНАГЛІФ; ПРОГРАМ; OPENGL; СТЕРЕО; ЗОБРАЖЕННЯ.

SUMMARY

Kyselevych Vladyslav Yurievych

Topic: iOS application of diploma design management system.

Specialty: 122 Computer Science and Information Technology.

Institution: National Technical University of Ukraine "Kyiv Polytechnic Institute. Igor Sikorsky ", 2021

Bachelor work: . s., introduction,. sections, conclusion, 9 source,. applications,. drawings,. tables.

Actuality of theme.

Today, stereo images play a big role in people's lives. Every day, people go to the movies, play computer games without thinking about how they view stereo 3-D images. One of the global uses, in addition to the use in chemistry for the detailed visualization of molecules, raster 3-D stereo images are transmitted to study the surfaces of other planets. In order to work with such images, existing methods of stereo image separation are used. My task was to program an application that would develop cascading stereo images to demonstrate the operation of the anaglyphic method.

The purpose of the study. The purpose of the study is to develop an extension of an existing software product to demonstrate the operation of the anaglyphic method using a variety of existing color spectra.

The object of research are color spectra using the anaglyphic method

The subject of research is the application for visualization of frame images

Results of work: a software product with frame images and an interface for sampling color spectra has been developed

Keywords: ANAGLYPH; PROGRAMM; OPENGL; STEREO; IMAGES.

ЗМІСТ

ВСТУП……8
1.Задача розробки засобів візуалізації анагліфічних каркасних зображень 9
1.1 Основні задачі.....9
1.2 Задачі для системи.....9
1.2.1 Загальні вимоги 9
1.2.2 Головні задачі 9
1.2.3 Додаткові задачі 10
1.3 Задачі додатку.....10
1.3.1 Загальні вимоги 11
1.3.2 Головні задачі 12
2. Аналіз аналогічних засобів візуалізації 14
2.1 Аналогічні системи.....14
2.2.1 Редактор зображень Adobe Photoshop. 14
2.2.2 NVIDIA 3D Vision – драйвер з викристанням 3-Д стерео.....17
2.2.3 Nasa Mastcam. 20
2.2.4 ChemDoodle 3D.....23
3.1 Додаток 25
3.1.1 Windows 27
3.1.2 C.....29
3.1.3 C++ 31

3.1.4 C# 33
3.1.5 Платформа .NET 36
3.1.6 Nuget 38
3.1.7 OpenGL 41
3.1.8 GLUT 42
3.1.9 GLX 43
3.1.10 GLU 43
3.1.11 Visual Studio – редактор коду 44
4.2 Налаштування технологій та їх взаємодія 45
4.Опис програмної реалізації 46
4.1 Структурна схема 46
4.2 UML діаграма системи 48
5 Методика роботи з програмою 49
5.1 Інтерфейс та меню 49
5.2 Каркасні моделі 51
5.3 Кольорові спектри 53
5.4 Ефект парралаксу 55
5.3 Впровадження зовнішніх комп'юетрних приладів 57
Висновки 58
Список використаних джерел 59

ВСТУП

На сьогоднішній день існує кілька сучасних технічних рішень для перегляду (демонстрації) об'ємного (3D-стерео) рухомого або статичного зображення, анагліфічний метод - один із них.

Анагліф (від грец. Ἀνάγλυφος «рельєфний») - метод сепарації зображень чорно-білої стереопари за допомогою колірного кодування. Різні частини стереопари, призначені для лівого і правого ока, проектуються або друкуються на одну і ту ж поверхню, але забарвлюються в різні кольори, додаткові один до одного. Червоно-сині (в моєму випадку) анагліфічні окуляри дозволяють «обдурити» мозок і створити ілюзію тривимірності зображення за рахунок колірного кодування.

На теоретичній основі дуже важко зрозуміти як працює цей метод и навіщо він потрібен. Саме тому розроблена програма полегшить для студентів сприйняття та розуміння принципу роботи анагліфічного методу та явища сепарації в цілому. За допомогою окулярів студенти, вивчаючи дану наукову область, зможуть не лише почути, але й побачити так звану «глибину» геометричних фігур , що на мою думку, дуже цікаве та захоплююче явище, яке пробуджує інтерес до такої науки як стереоскопія.

Сьогодні анагліфічний метод активно використовується для відображення моделей хімічних молекул, використовується в дослідженні поверхні планети Марс, раніше активно використовувався в 3-Д кінематографі, в розробці ігор. Анагліфний метод – найдешевший в використанні, одним із його основних переваг над іншими методами є доступність для кожного, адже зараз будь-хто, використовуючи інтернет, має змогу подивитись стереоскопічне кіно, відео або зображення, для цього знадобляться лише анагліфічні окуляри, які можна легко та дешево придбати. Така програма візуалізації є актуальною для будь-якого вищого навчального закладу.

Для реалізації програмної частини додатку було вирішено використовувати фреймворк OpenGL. Однією з важливих переаваг є ортогоналість - всі функції

OpenGL є ортогональними, тобто незалежними, що дає змогу використовувати їх в довільній комбінації, наприклад використання меппінга не обмежує можливостей застосування світлотіні. Головною перевагою OpenGL в моїй програмі це здобуття робочої картинки, навіть якщо продуктивність і не дозволяє отримати її з усіма подробицями. Тобто, якщо щось працює на одній платформі, то цей же код буде працювати і на інший. Базові можливості цього фреймворку полегшують та дозволяють розробнику інтегрувати в розробку деякі функції, які я використав, а саме:

- створення геометричних і растрових примітивів, на основі яких будуються всі об'єкти.
- видові і модельні перетворення, що дозволяють обертати об'єкти, розташовувати їх в просторі, змінювати форму, а також змінювати положення камери, з якої ведеться спостереження.
- робота з кольором. OpenGL надає можливості роботи з кольором в режимі RGBA або використовуючи індексний режим, де колір вибирається з палітри.
- подвійна буферизація, що дозволяє усунути мерехтіння при мультиплікації (зображення кожного кадру спочатку малюється в другому буфері, а потім, коли кадр повністю намальований, весь буфер відображається на екран).

Помаранчева книга OpenGL схематично демонструє програмований конвеєр OpenGL з вершинними і фрагментного обработчиками. Як ви бачите, конвеєр OpenGL є досить складним, але вам не потрібно розуміти принцип роботи кожного з його компонентів для того, щоб успішно використовувати OpenGL.

Архітектура OpenGL дозволяє реалізувати додаток в повному обсязі згідно обраної теми для реалізації програмного продукту (Рисунок 0.0.1).

Схематичне зображення конвеєра (Рисунок 0.0.2) ілюструє принцип роботи бібліотек OpenGL.

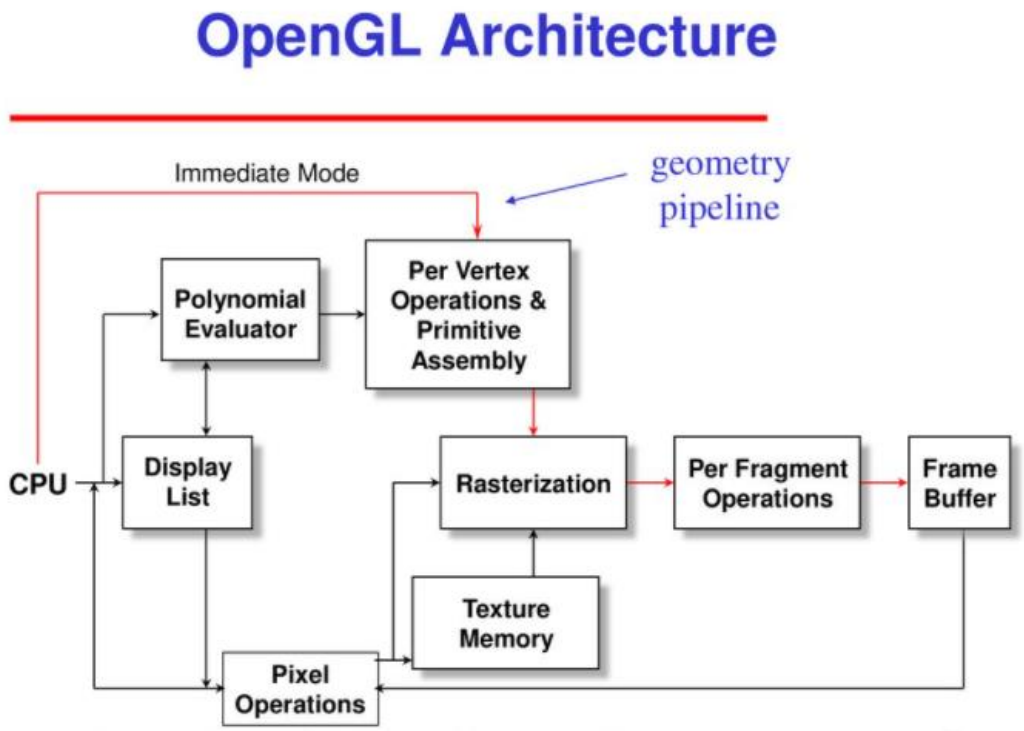

Рисунок 0.0.1 — Архітектура OpenGL

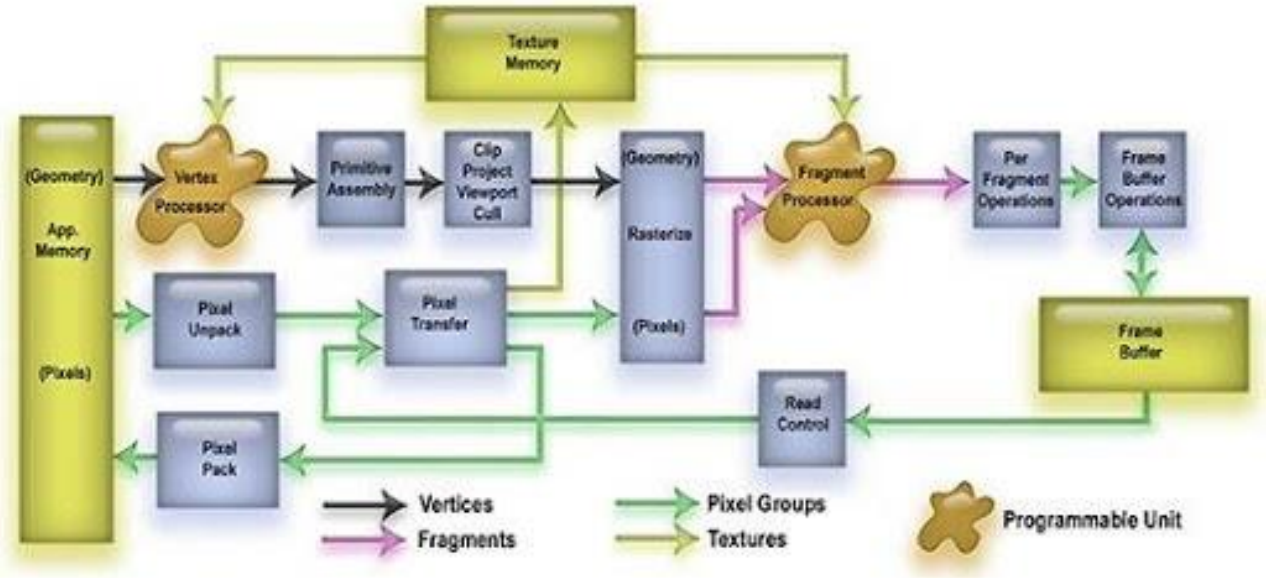

Рисунок 0.0.2 — Конвейер OpenGL

1. ЗАДАЧА РОЗРОБКИ ЗАСОБІВ ВІЗУАЛІЗАЦІЇ АНАГЛІФІЧНИХ КАРКАСНИХ ЗОБРАЖЕНЬ

У розділі проаналізовано вимоги до розробки засобів візуалізації каркасних зображень, побудови стереопари при розробці анагліфічного методу сепарації.

1.1 Основні задачі

Дослідити кольорові спектри, розробити функціонал їх впровадженя в програмі. Розробка побудови стереопари, побудува каркасних зображень, інтерфейс для динамічної зміни зображень, кольорового спектру.

1.2 Задачі для програми

Задачами для програми є візуалізація стереозображень анагліфним методом, розробка різноманітних кольорових спетрів для взаєємодії з існуючими анаглфіними окулярами, функціонал взаємодії з каркасними зображеннями.

1.2.1 Загальні вимоги

Інтерфейс повинен бути зручним та зрозумілим для будь-якого користувача.

Запрограмувати основні налаштування так , щоб дати змогу користувачу змогу змінювати під час роботи програми.

Додаток не має перевизначати або некоректно використовувати інтегровані бібліотеки для автоматизації явища.

Взаємодія зовнішніх комп’ютерних приладів з системою(мишка, клавіатура)

1.2.2 Головні задачі

Додаток розрахований на будь-якого користувача, інтерфейс повинен бути зрозумілим та зручним.

Необхідно розробити функціонал для обертання навколо осі, впровадити функціонал для зовнішнього управління анагліфічного явища в 3-Д просторі за допомогою комп'ютерними приладів.

1.2.3 Додаткові задачі

- Обертання за потребою користувача
    - швидкість обертання;
    - обертання за допомогою зовнішніх комп'ютерних приладів;
- зміна кольору стерео-пари для будь-якого виду анагліфних окулярів;
- змога вибрати фігуру із списку наданих.

1.3 Задачі для додатку

-У додатку повинен бути реалізований інтерфейс взаємодії між користувачем та додатком.

-Інтерфейс повинен працювати динамічно при зміні кольорової палітри , швидкості обертання , зсуву зображень по осям.

-Розробити та впровадити в систему використання існуючих кольорових спектрів.

-Розробити каркасні моделі та їх взаємодію з користувачем.

-Розробити функціонал взаємодії таких приладів як мишка та клавіатура для вазємодії з програмою.

1.3.1 Загальні вимоги

Додаток та користувач взаємодіють за допомогою мишки та клавіатури.

Для розробки явища анагліфної сепарації застосований метод сходження для проекції лівого та правого ока на екран.

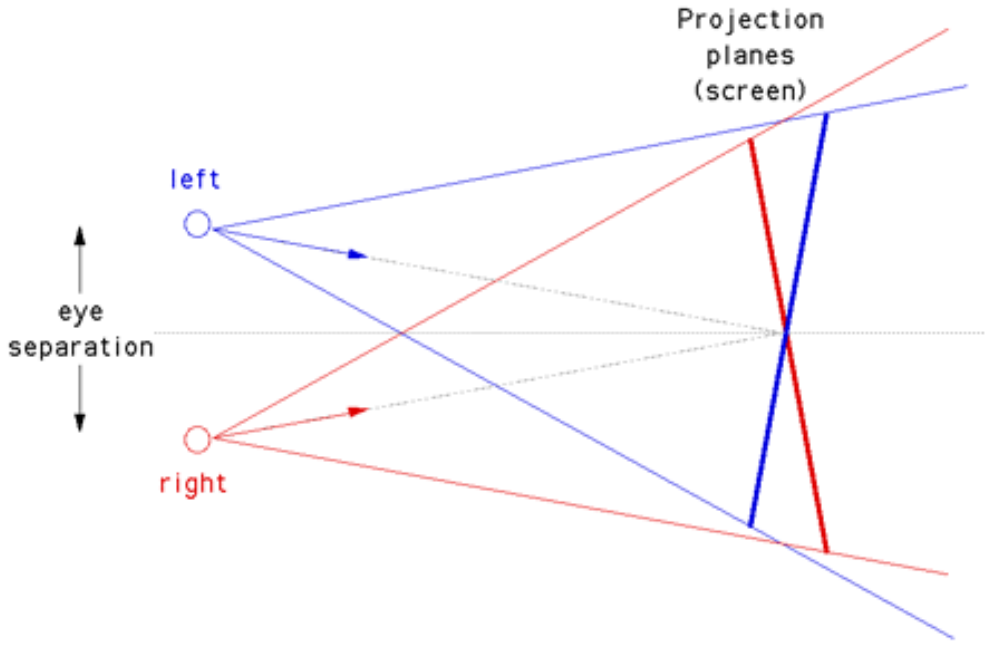

Рисунок 1.3.1 — метод «сходження»

1.3.2 Головні задачі

Об'єкти, розташовані ближче до камери, ніж відстань конвергенції здаватимуться поза екраном, а об'єкти, що знаходяться далі глибини, ніж відстань конвергенції, з'являться всередині екрана. (Рисунок 1.3.2)

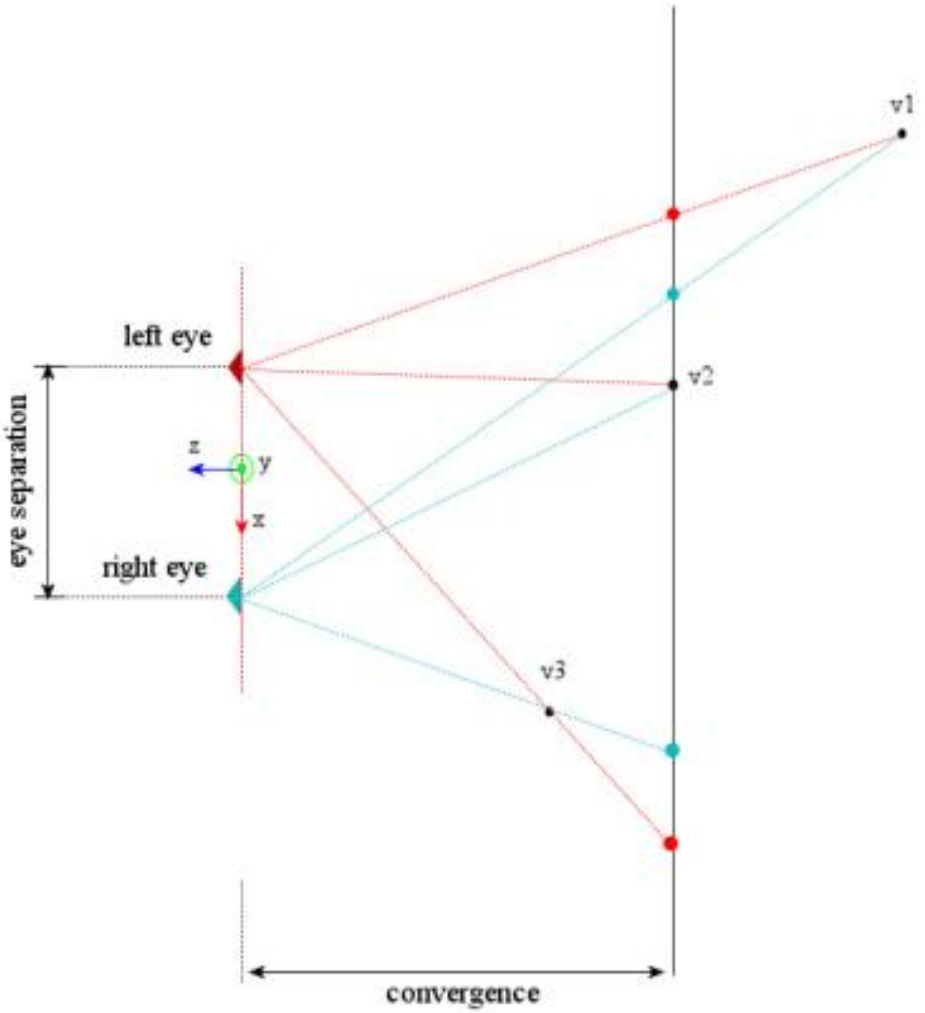

Рисунок 1.3.2 — користувач та экран

Додаток дозволяє користувачу взаємодіяти з функціоналом через динамічну зміну глобальних змінних, які використовуються, як налаштування програми.

Користувач може розвернути панель взаємодії , а саме : зміна кольору стерео-пари, налаштування відображення “ on screen ” , “ in front of screen ”, “ behind of screen ” , Зміна геометричної фігури , вибір швидкості обертання/зупинка обертання – все це користувач може змінити під час роботи програми через натискання на області 3-Д простору правої клавіші мишки.(Рисунок 1.3.3)

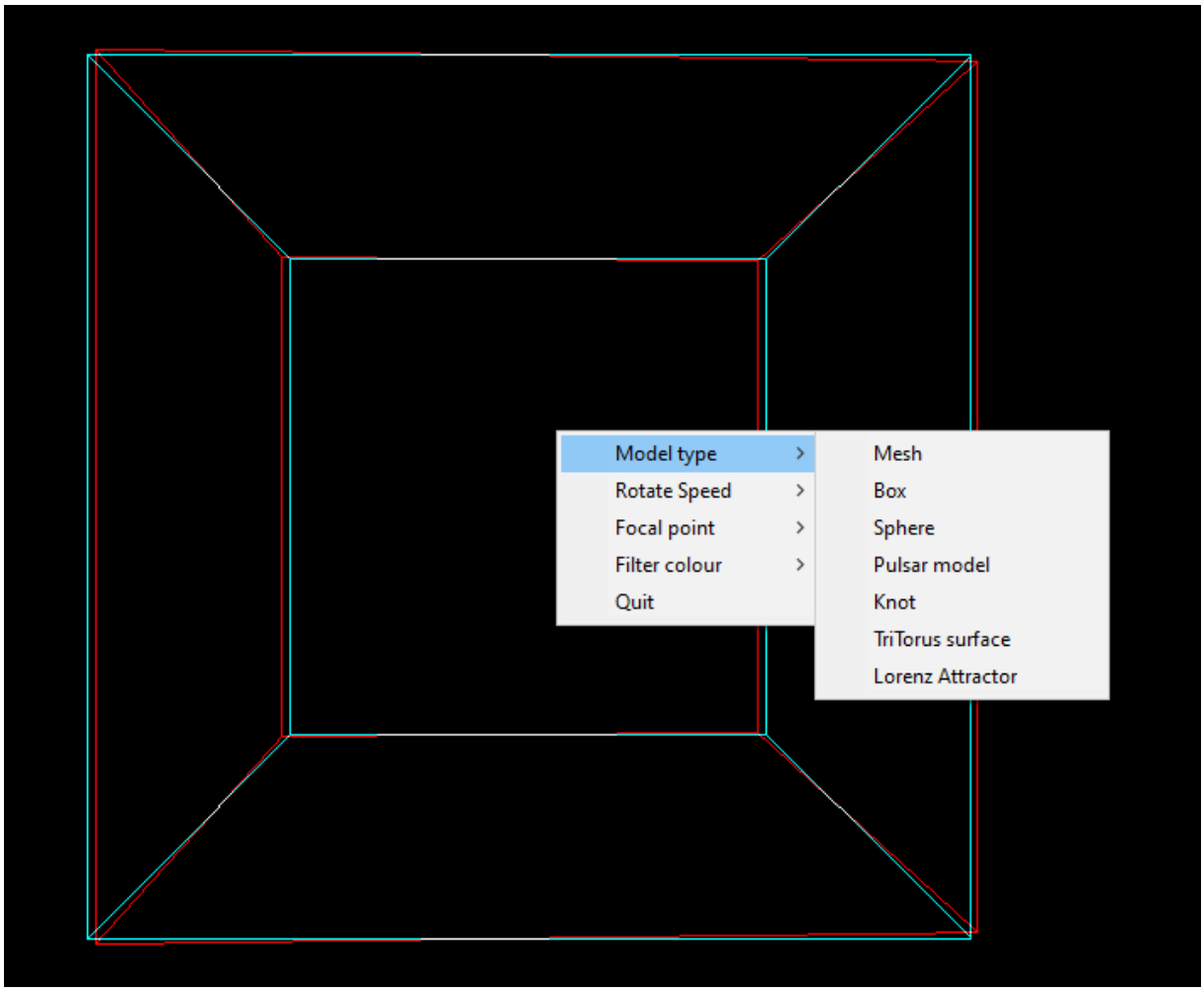

Рисунок 1.3.3 — інтерфейс взаємодії

2. АНАЛІЗ АНАЛОГІЧНИХ ЗАСОБІВ ВІЗУАЛІЗАЦІЇ

2.1 Аналогічні системи

На сьогоднішній день явище анагліфної 3-Д сепарації викрористовується в навчальних закладах для візуалізації хімічних 3-Д молекул та в розробці комп'ютерних ігор.

2.1.1 Редактор зображень "Adobe Photoshop".

Одним з найдавніших і найпростіших методів створення тривимірних зображень є техніка, відома як анагліф 3D. Фільми про монстрів та інопланетяни 50-х, 60-х і навіть 70-х використовували цей тип 3D як трюк, щоб залучити аудиторію та зробити так, щоб вони відчували себе більше, ніби були частиною фільму. З плином років, а технологія прогресувала і трансформувалась, анагліф 3D залишався новинкою і підтримував популярність серед фільмів, серіалів та соцмереж, як Youtube.

Поряд з Photoshop, Adobe також розробляє та публікує Photoshop Elements, Photoshop Lightroom, Photoshop Express, Photoshop Fix, Photoshop Sketch та Photoshop Mix. Станом на листопад 2019 року

Згідно з теорією тристимулу, людське око дуже чутливе до основних кольорів - червоного, зеленого та синього. У випадку Anaglyph 3D ця чутливість дозволяє червоному фільтру блокувати блакитний колір на одному оці, а потім блакитному фільтру блокує червоний, ефективно надаючи вам автономне стереоскопічне зображення.

У будь-якому випадку ви подивитесь на це, ось простий спосіб перетворити зображення та мистецтво Photoshop(Рисунок 2.1.1) у проекти 3D.

Рисунок 2.1.1 — Adobe Photoshop

Одягнтьі свої 3D-окуляри і подивіться на своє зображення. Ваше зображення має мати більшу глибину і виглядати дивно 3D. Щоб отримати більшу глибину, перемістіть шар «cyan» далі від «red» шару. Тепер, оскільки ми застосували цю техніку до всього зображення, ви не побачите великого розмежування між усіма різними об’єктами вашого зображення.

Щоб отримати ще більше вражаючих 3D-ефектів, виберіть і відокремте кожен окремий об’єкт, який ви хочете налаштувати самостійно, і виконайте однакову 3D-техніку для кожного. (Рисунок – 2.1.2)

Рисунок – 2.1.2 – 3-Д стерео-ефект

2.1.2 NVIDIA 3D Vision – драйвер з використання 3-Д стерео.

NVIDIA (Рисунок 2.1.3) - американська багатонаціональна технологічна компанія, зареєстрована в штаті Делавер та базується в Санта-Кларі, штат Каліфорнія. Він розробляє графічні процесори (графічні процесори) для ігрових та професійних ринків, а також системи на мікросхемах (SoC) для мобільних обчислювальних машин та автомобілебудування.

Рисунок 2.1.3 — інтро зображення NVIDIA

Коли відтворюються 3D-ігри та вмикається драйвер, драйвер перетворює зображення у 3D стерео і відображає його на екрані. Ви можете налаштувати драйвер для використання будь-якої сторінки метод перегортання або анагліф стереозображення. Завдяки сумісному обладнанню для перегляду ви зможете побачити зображення із сприйняттям глибини.

Стерео-драйвер NVIDIA 3D поставляється з низкою елементів управління для використання в іграх:

• Налаштування стереосепарації відповідно до індивідуальної адаптації до стереоперегляду.

• Гарячі клавіші для ігрового стерео управління.

• 3D лазерний приціл для додаткового реалізму в шутерах від першої особи.

• Процес стереотесту для визначення найкращого режиму екрану для використання.

• Спеціальні конфігурації ігор пристосовують стереопроцес до широкого кола ігор

Окуляри Nvidia можна надягати поверх діоптричних, а для більшої зручності в комплект входить кілька різних вставок для перенісся. На відміну від старовинних аналогів, які читачі можуть пам'ятати з часів розквіту ПК-ігор, окуляри Nvidia беспроводние.Очкі Nvidia можна надягати поверх діоптричних, а для більшої зручності в комплект входить кілька різних вставок для перенісся. На відміну від старовинних аналогів, які читачі можуть пам'ятати з часів розквіту ПК-ігор, окуляри Nvidia бездротові.(Рисунок 2.1.3)

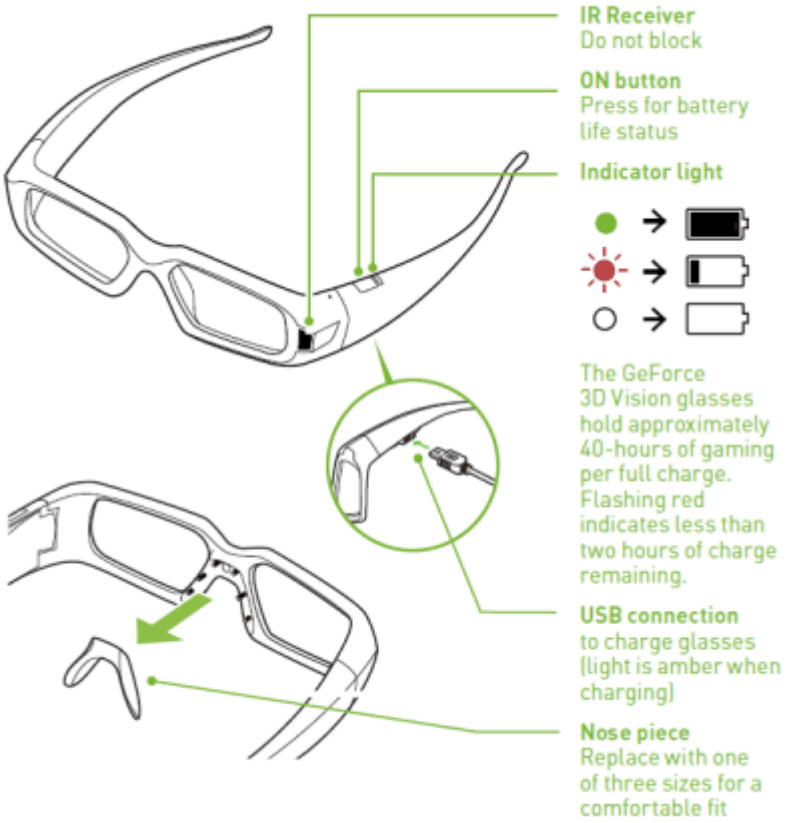

Рисунок 2.1.3 — бездротові окуляри NVIDIA

На сьогоднішній день технологія підтримується більш ніж в 150 іграх. Сюди входять як провідні тайтли минулого року: Need for Speed sfift, Fifa 10, Resident Evil 5, Batman Arkham Asylum і Disciples 3, так і сучасні хіти: Bioshock 2, Battlefield 2 bad company, Metro2033, Mafia2. (Рисунок 2.1.4) Чи не станемо розповідати про глибину сприйняття і неймовірних відчуттях занурення - щоб зрозуміти, це треба спробувати самому.

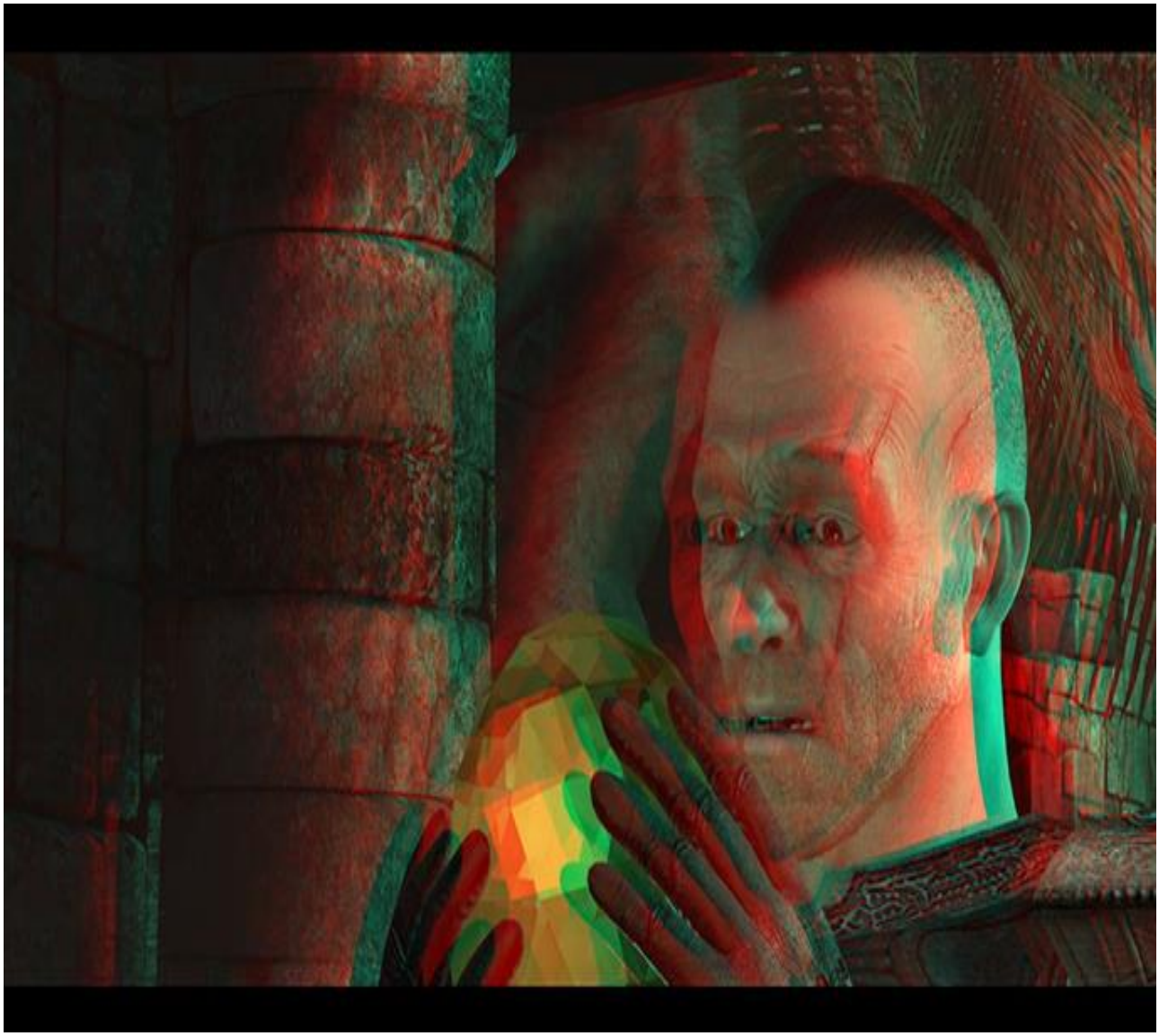

Рисунок 2.1.4 — 3-Д стерео-ефект в іграх

2.1.3 NASA Mastcam

Камера Mast (Рисунок 2.1.5), або коротше Mastcam, робить кольорові зображення та кольорові відеокадри марсіанської місцевості. Зображення можна з’єднати, щоб створити панорами пейзажу навколо марсохода. Як і камери на марсоходах Mars Exploration, які приземлилися на Червоній планеті в 2004 році, конструкція Mastcam складається з двох систем камер, встановлених на щоглі, що тягнеться вгору від палуби марсохода Mars Science Laboratory (корпусу). Mastcam можна використовувати для вивчення марсіанського ландшафту, скель та ґрунтів; для перегляду погодних явищ; і для підтримки рушійних та пробних операцій ровера.

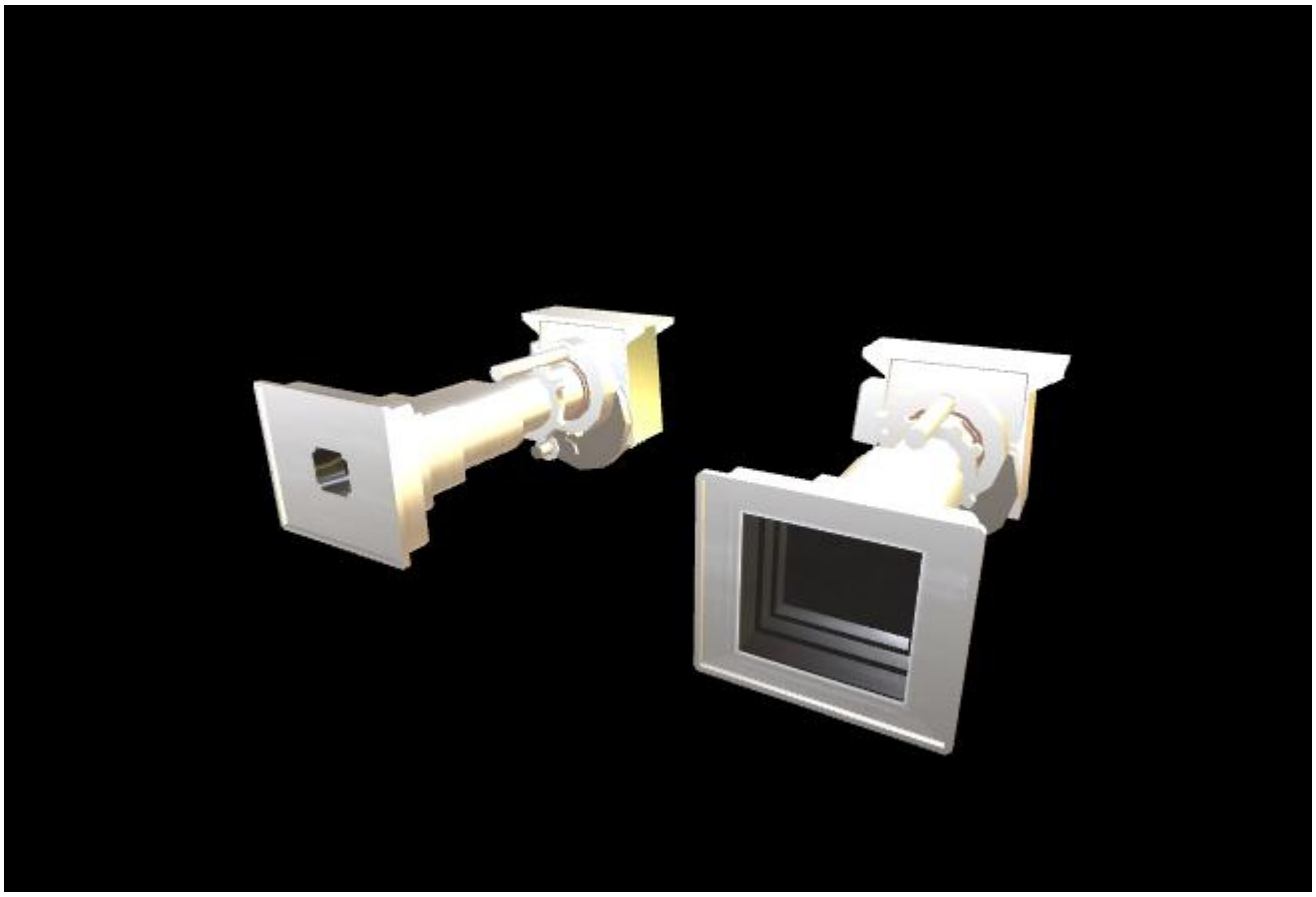

Рисунок 2.1.5 — 3-Д модель камери Mastcam

Марсохід NASA Curiosity Mars (Рисунок 2.1.6) за допомогою свого приладу Mastcam зробив 32 окремих зображення, що складають цю панораму відслонення на прізвисько "Мон Мерку". Для створення стереоскопічного ефекту, подібного до тривимірного видошукача, знадобилася друга панорама, котячись убік на 13 футів (4 метри). Ефект допомагає вченим краще зрозуміти геометрію осадових шарів гори Мерку, ніби вони стоять перед формацією.

Обидві панорами були зроблені 4 березня 2021 року, на 3049-й марсіанський день місії, з відстані близько 40 футів (40 метрів) від обриву скелі, висота якого близько 20 футів (6 метрів). Вони були збалансовані за білим кольором, так що кольори гірських порід нагадують те, як вони могли б виглядати в денних умовах освітлення на Землі.

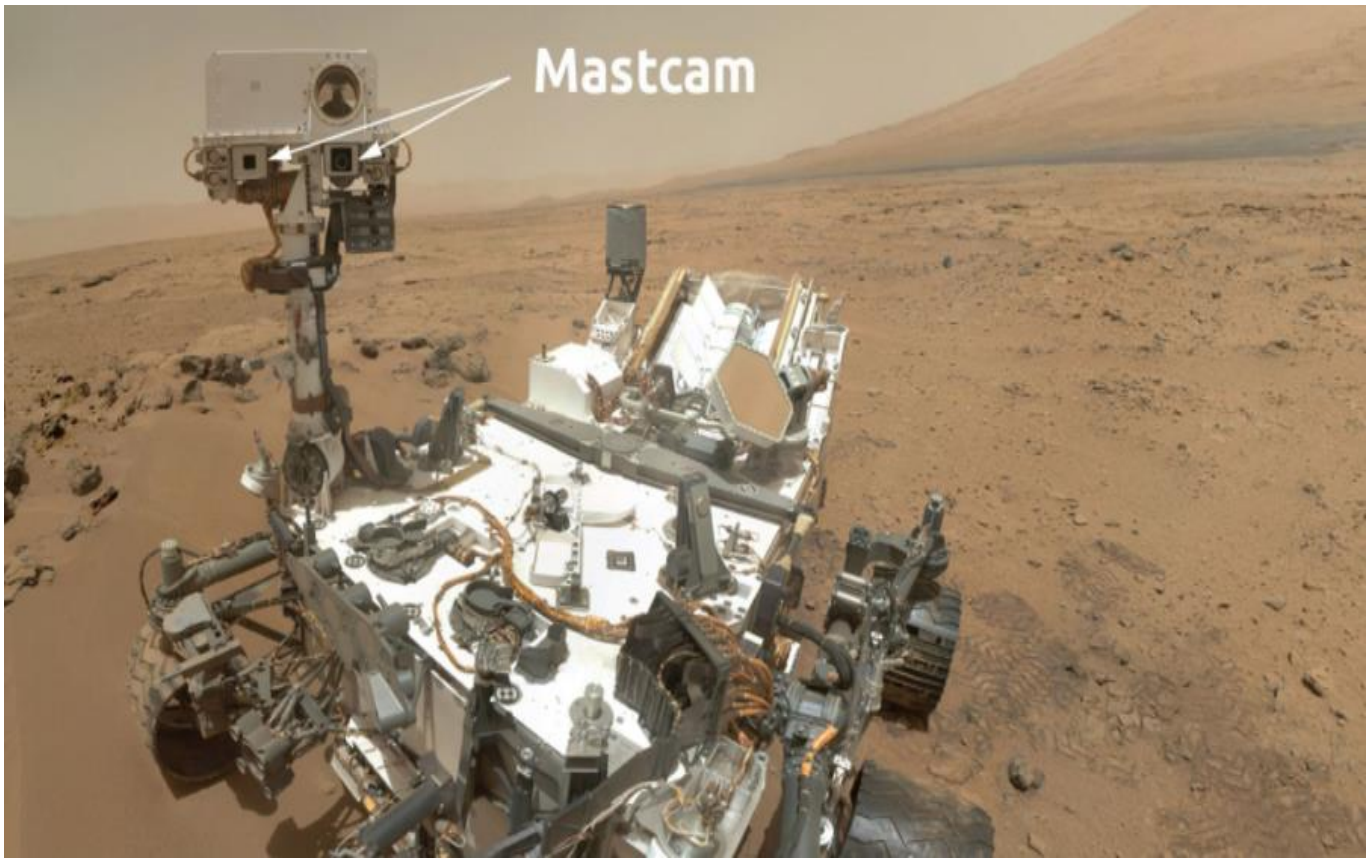

Рисунок 2.1.6 — Марсохід NASA Curiosity Mars

Сюди входить анімація, яка показує стереоскопічний ефект (Рисунок 2.1.9) разом із двома панорамами (Рисунок 2.1.7, 2.1.8), використовуваними для його створення. Додатковий анагліф пропонує тривимірну проблиск у вигляді червоно-синіх окулярів.

Рисунок 2.1.7 — Перша панорама

Рисунок 2.1.8 — Друга панорама

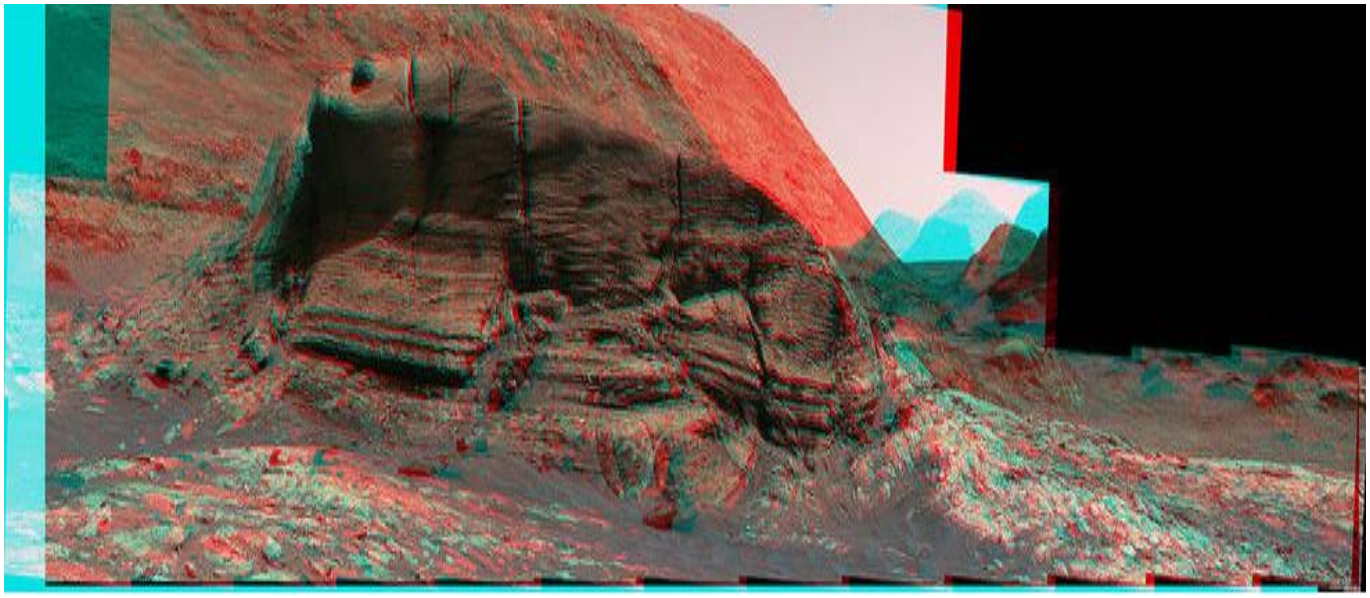

Рисунок 2.1.9 — Стереоскопічний ефект

2.1.4 ChemDoodle 3D

ChemDoodle 3D представляє розширену та настроювану підтримку анагліфів для доповненої реальності (AR) прямо на вашому комп’ютері. Тепер ви можете повністю візуалізувати 3D-молекули, які ви будуєте в 3D! Якщо у вас червоно-блакитні анагліфічні окуляри, ви побачите наведене вище зображення у форматі 3D (Рисунок 2.1.10). Ви можете натиснути на зображення, щоб побачити його у більшому розмірі.

На додаток до класичних червоно-блакитних окулярів підтримуються також зелено-пурпурові та бурштиново-сині окуляри.

Також застосований метод Дюбуа для найбільш комфортного перегляду анагліфів. Застосування методу Дюбуа - варіант, і ви можете керувати як фокусною відстанню, так і параметрами відокремлення очей, щоб налаштувати анагліф на свій смак.

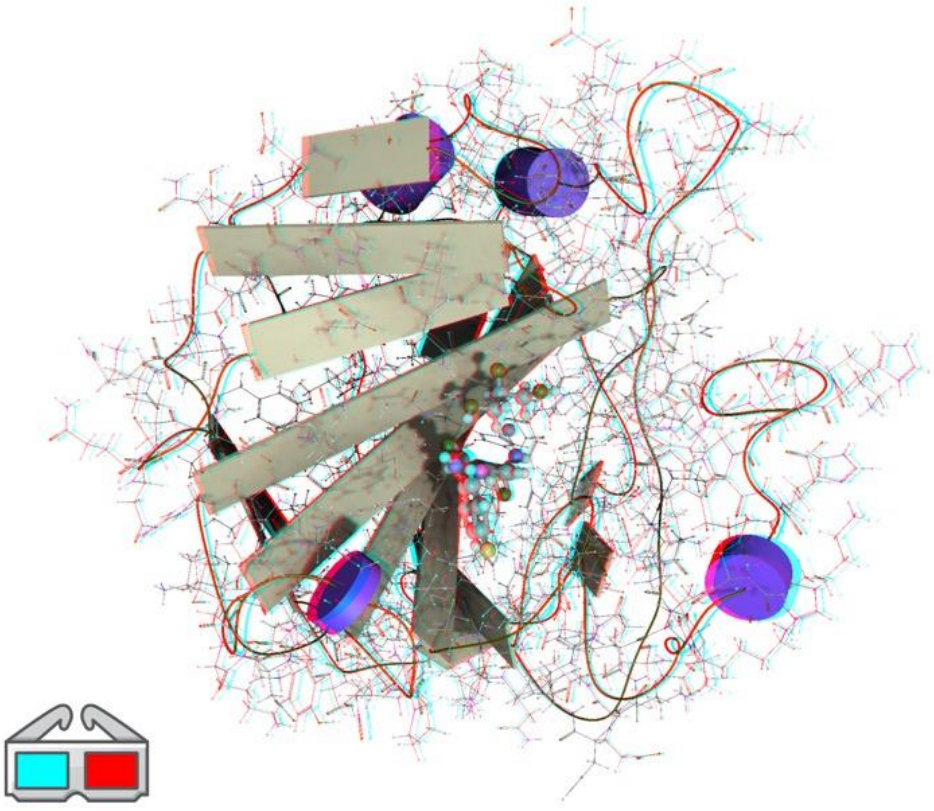

Рисунок 2.1.10 — Приклад використання анагліфів в хімії

Тривимірна модель генерується хімічним розпізнавачем ідентифікаторів, який підтримується Markus Sitzmann. Якщо 3D-модель не завантажується, можливо, розчинник хімічних ідентифікаторів не працює. Будь ласка, поверніться пізніше. Ви також можете використовувати інструменти оптимізації силового поля в ChemDoodle 3D для побудови тривимірних хімічних структур у реальному часі.

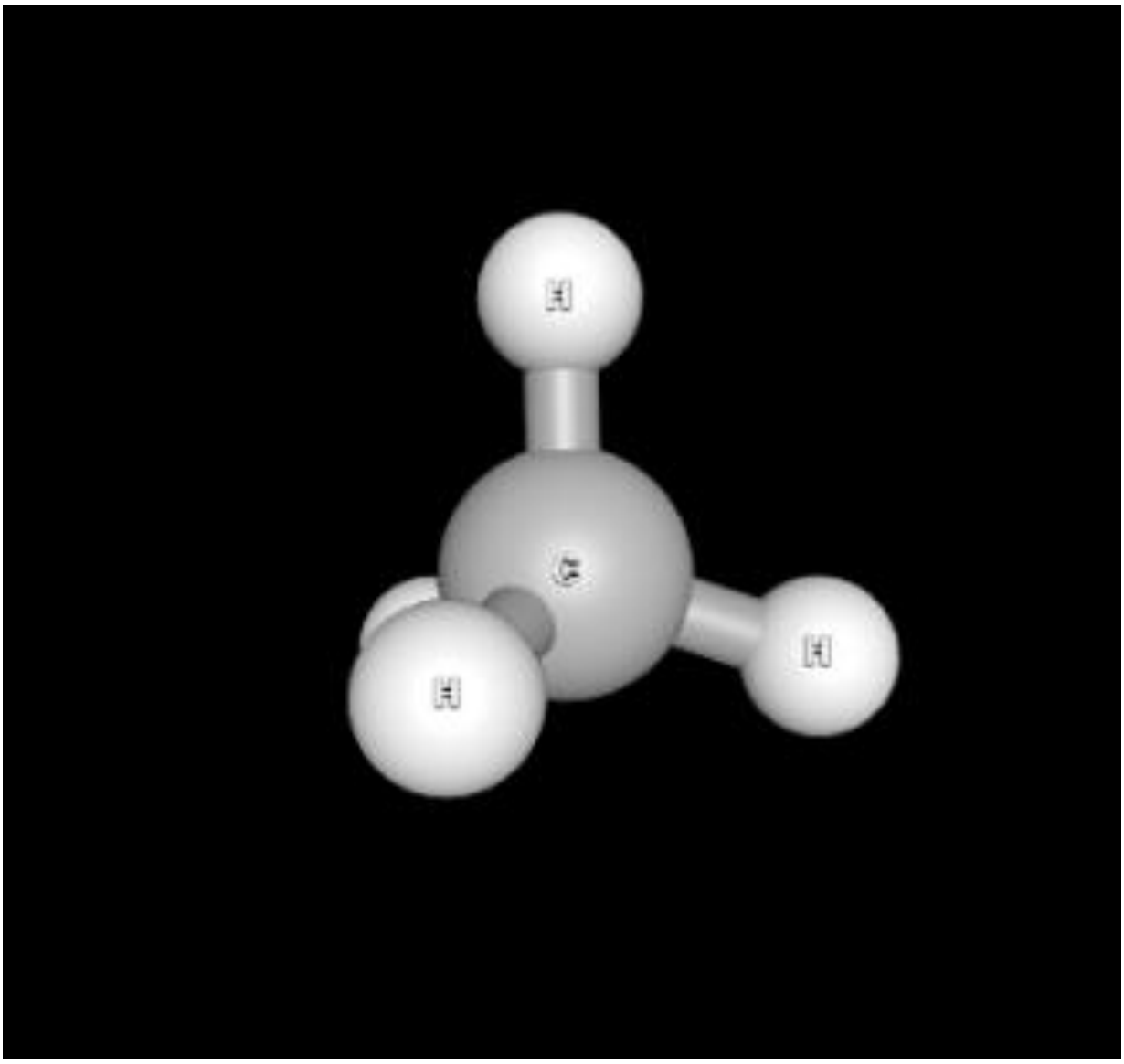

Рисунок 2.1.11 — Тривімірна модель

3 ЗАСОБИ РОЗРОБКИ

Для розробки використовувались актуальні на сгогодення мови програмування та модулі.

3.1 Додаток

Реалізація веб-застосунку проводилася у середовищі Visual Studio 2019.

Для розробки додатку було вирішено використовувати мови C.

Мова програмування C - найшвидший у світі високорівнева мова програмування. Для мови C характерні лаконічність, стандартний набір конструкцій управління потоком виконання, структур даних і великий набір операцій. Мова C є найпопулярнішою мовою для створення системного програмного забезпечення. Його також часто використовують для створення прикладних програм. Надалі синтаксис мови C став основою для багатьох інших мов.

Для динамічної роботи системи та її налаштування використовувалась бібліотека GLUT. Вона бере на себе роботу по створенню вікна, ініціалізації OpenGL, обробки повідомлень від миші і клавіатури. Більш того, використовуючи бібліотеку GLUT, такі функції, як включення апаратного згладжування. Альтернатива вільному програмному забезпеченню / відкритому коду бібліотеці OpenGL Utility Toolkit (GLUT). GLUT застосовується в широкому діапазоні практичних застосувань, оскільки він простий, широко доступний і дуже портативний. GLUT (а отже, і freeglut) дбає про всі системні клопоти, необхідні для створення вікон, ініціалізації контекстів OpenGL та обробки вхідних подій, щоб забезпечити справді портативні програми OpenGL.

reeglut випускається за ліцензією X-Consortium.

Для взаэмодії користувача з інтерфейсом використовувалась мова C# платформи .NET.

Веб-додаток буде встановлений на операційній системі Windows 10, як на самій оптимальній ОС для даного застосунку, на якій можна використовувати додатки з великим навантаженням.

3.1.1 Windows 10

Windows 10 - це серія операційних систем, розроблена корпорацією Майкрософт і випущена як частина її сімейства операційних систем Windows NT. Він є наступником Windows 8.1, випущеного майже двома роками раніше, і випущений у виробництво 15 липня 2015 року, і широко випущений для широкої громадськості 29 липня 2015 року [18]. Windows 10 був доступний для завантаження через MSDN та Technet, як безкоштовне оновлення для роздрібних копій користувачів Windows 8 та Windows 8.1 через Магазин Windows, а також для користувачів Windows 7 через Windows Update. Windows 10 постійно отримує нові збірки, які доступні без додаткових витрат для користувачів, на додаток до додаткових тестових збірок Windows 10, доступних для інсайдерів Windows.

Windows 10 отримала переважно позитивні відгуки після свого первісного випуску. Критики високо оцінили рішення Microsoft запропонувати настільний орієнтований інтерфейс відповідно до попередніх версій Windows, протиставляючи орієнтований на планшети підхід Windows 8, хоча режим сенсорного орієнтованого інтерфейсу Windows 10 критикували за утримання регресій щодо сенсорно-орієнтованого інтерфейсу свого попередника.

Windows 10 доступний як у 32, так і в 64-розрядної архітектурі. Це в основному означає обсяг пам’яті, на який здатний адресувати ваш комп’ютер. Деякі комп’ютери можуть підтримувати 64-розрядну версію, але обмежені об’ємом пам'яті, яку можна встановити. 64-розрядна версія Windows 10 Home підтримує до 128 ГБ оперативної пам'яті, тоді як Windows 10 Pro, Pro Edu, Education та Enterprise

підтримують до 2 ТБ оперативної пам'яті, Pro for Workstation може використовувати до 6 ТБ.

Windows SIM (Рисунок 3.1.1) використовує інтерфейс платформи компонентів (API CPI) для створення файлів відповідей і управління ними. Компоненти і параметри в конкретному образі Windows використовуються для створення файлу каталогу. Цей файл каталогу використовується в Windows SIM для створення файлів відповідей.

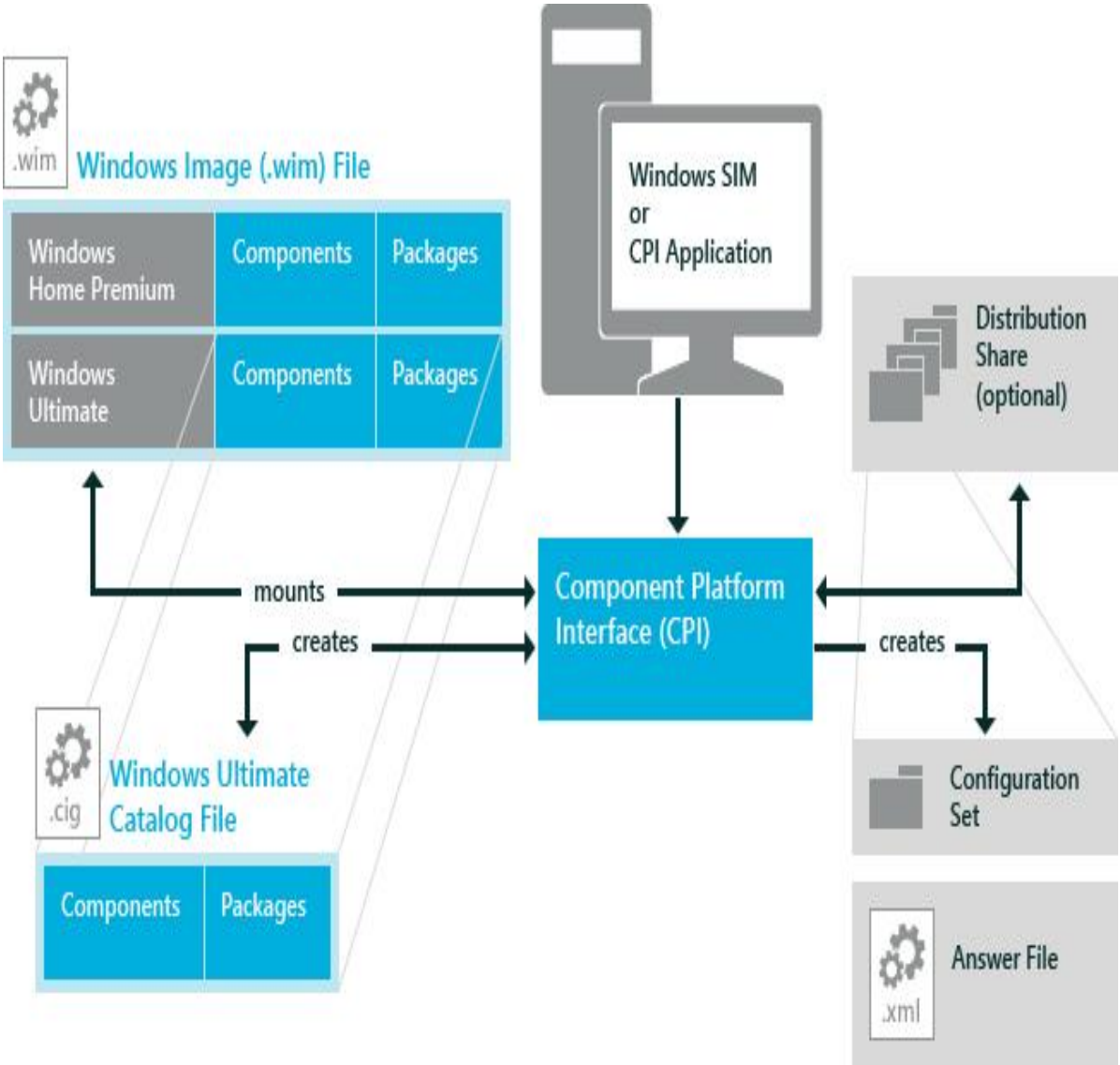

Рисунок 3.1.1 — Архітектура Windows 10

3.1.2 Мова програмування C

C - це загальноприйнята процедурна мова комп’ютерного програмування, що підтримує структуроване програмування, лексичну область змінних та рекурсію із системою статичного типу. За задумом C пропонує конструкції, які ефективно відображають типові машинні інструкції. Він знайшов тривале використання в додатках, кодованих раніше мовою асемблера. Такі додатки включають операційні системи та різне прикладне програмне забезпечення для комп'ютерних архітектур, які варіюються від суперкомп'ютерів до ПЛК та вбудованих систем.

Він був застосований для повторної реалізації ядра операційної системи Unix. Протягом 1980-х років C поступово набирав популярності. Він став однією з найбільш широко використовуваних мов програмування, із компіляторами C від різних постачальників, доступних для більшості існуючих архітектур комп'ютерів та операційних систем. C стандартизовано ANSI з 1989 р. (ANSI C) та Міжнародною організацією зі стандартизації (ISO) (Риснок 3.1.2).

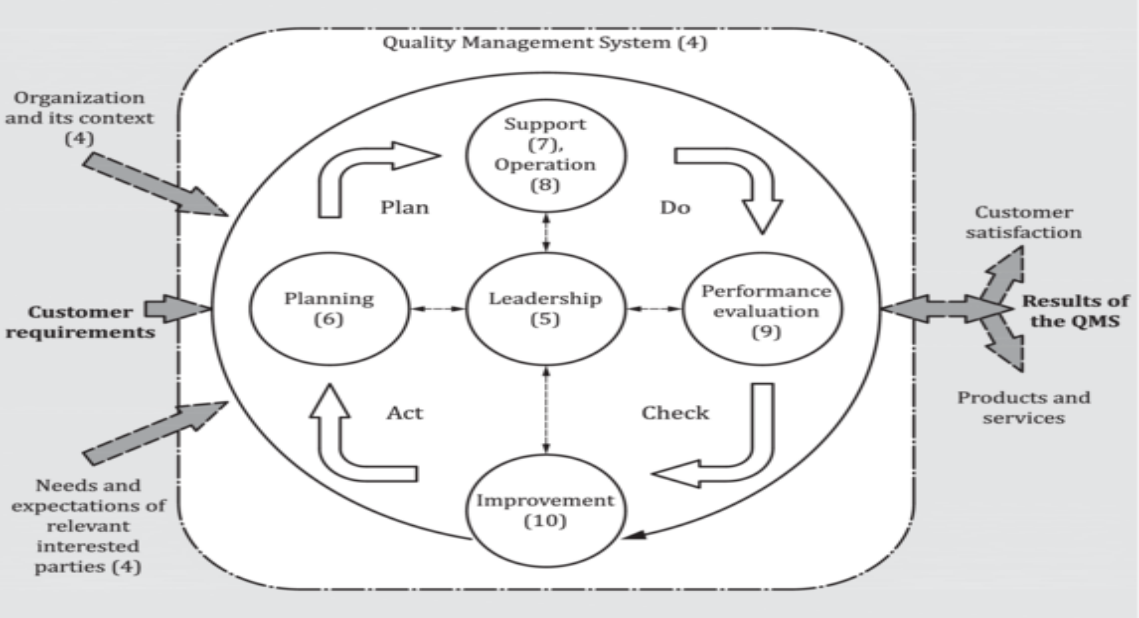

Рисунок 3.1.2 — ISO схема

C (Рисунок 3.1.2) є імперативною процесуальною мовою. Він був розроблений для компіляції, щоб забезпечити низькорівневий доступ до пам’яті та мовних конструкцій, які ефективно відповідають машинним інструкціям, все з мінімальною підтримкою виконання. Незважаючи на свої низькорівневі можливості, мова була розроблена для заохочення міжплатформеного програмування. Програма C, що відповідає стандартам, написана з урахуванням переносимості, може бути складена для широкого кола комп'ютерних платформ та операційних систем з невеликим зміною вихідного коду

Рисунок 3.1.3 — Синтаксис мови C

3.1.3 Мова програмування C++

Мова C ++ складається з двох основних компонентів: прямого відображення апаратних функцій, що забезпечуються, головним чином, підмножиною C, та абстракцій із нульовими накладними витратами на основі цих відображень. Stroustrup описує C ++ як "полегшену мову програмування абстракцій [розроблену] для побудови та використання ефективних та елегантних абстракцій"; та "пропонування як апаратного доступу, так і абстракції є основою C ++. Ефективне використання - це те, що відрізняє його від інші мови.

Статичні об'єкти тривалості зберігання створюються до введення main () та знищуються в зворотному порядку створення після виходу main (). Точний порядок створення не визначений стандартом (хоча є деякі правила, визначені нижче), щоб забезпечити реалізація певну свободу в організації їх реалізації. Більш формально, об’єкти цього типу мають тривалість життя, яка «триватиме протягом програми».

Статичні об'єкти тривалості зберігання ініціалізуються у два етапи. Спочатку виконується "статична ініціалізація", і лише після того, як виконується вся статична ініціалізація, виконується "динамічна ініціалізація". При статичній ініціалізації всі об’єкти спочатку ініціалізуються нулями; після цього всі об'єкти, які мають фазу постійної ініціалізації, ініціалізуються константним виразом (тобто змінні, ініціалізовані літералом або constexpr). Незважаючи на те, що це не вказано в стандарті, стадію статичної ініціалізації можна завершити під час компіляції та зберегти у розділі даних виконуваного файлу. Динамічна ініціалізація включає всю ініціалізацію об’єкта, виконану за допомогою конструктора або виклику функції.

Найбільш поширеними типами змінних у C ++ є локальні змінні всередині функції або блоку та тимчасові змінні. Загальною особливістю автоматичних змінних є те, що вони мають термін служби, обмежений сферою дії змінної. Вони створюються та потенційно ініціалізуються в точці оголошення та знищуються у зворотному порядку створення, коли область залишається. Це реалізується шляхом розподілу на стеку.

Локальні змінні створюються, коли точка виконання проходить точку оголошення. Якщо змінна має конструктор або ініціалізатор, це використовується для визначення початкового стану об'єкта. Локальні змінні знищуються, коли локальний блок або функція, в якій вони оголошені, закриті. Деструктори C ++ для локальних змінних викликаються в кінці життя об’єкта, дозволяючи дисципліну для автоматичного управління ресурсами під назвою RAII, яка широко використовується в C ++. Шаблони C ++ дозволяють загальне програмування. C ++ підтримує шаблони функцій, класів, псевдонімів та змінних. Шаблони можуть бути параметризовані за типами, константами часу компіляції та іншими шаблонами. Шаблони реалізуються за допомогою екземпляра під час компіляції. Щоб створити екземпляр шаблону, компілятори підставляють конкретні аргументи для параметрів шаблону, щоб створити конкретну функцію або екземпляр класу. Деякі заміни неможливі; вони усуваються політикою вирішення проблем із перевантаженням, описаною фразою "Помилка заміщення не є помилкою" (SFINAE). Шаблони - це потужний інструмент, який можна використовувати для загального програмування, метапрограмування шаблонів та оптимізації коду, але ця потужність передбачає витрати. Використання шаблону може збільшити розмір коду, оскільки кожна інстанція шаблону створює копію коду шаблону: по одній для кожного набору аргументів шаблону, однак це однакова або менша кількість коду, яка була б створена, якби код був написаний від руки. Це на відміну від узагальнених програм, що спостерігаються в інших мовах (наприклад, Java), де під час компіляції тип стирається, а єдине тіло шаблону зберігається.

Шаблони відрізняються від макросів: хоча обидві ці функції мови компіляції дозволяють умовне компілювання, шаблони не обмежуються лексичною заміною. Шаблони знають про семантику та систему типів супутньої мови, а також про всі визначення часу компіляції та можуть виконувати операції високого рівня, включаючи програмне управління потоком на основі оцінки строго перевірених параметрів. Макроси здатні умовно контролювати компіляцію на основі заздалегідь визначених критеріїв, але не можуть створювати нові типи, повторювати або

виконувати оцінку типів і фактично обмежуються заміною тексту до компіляції та включенням / виключенням тексту. Іншими словами, макроси можуть контролювати потік компіляції на основі заздалегідь визначених символів, але не можуть, на відміну від шаблонів, самостійно створювати екземпляри нових символів. Шаблони - це інструмент статичного поліморфізму (див. Нижче) та загального програмування.

3.1.4 Мова програмування C#

C# - мова загального призначення, призначена для розробки програм на платформі Microsoft, і для роботи потрібна платформа. NET у Windows. C # часто розглядають як гібрид, який використовує найкраще з C та C ++ для створення справді модернізованої мови. Хоча платформа. NET підтримує кілька інших мов кодування, C # швидко стала однією з найпопулярніших.

C # можна використовувати для створення майже будь-чого, але особливо сильний при створенні настільних додатків та ігор Windows. C # також може використовуватися для розробки веб-додатків і стає все більш популярним і для мобільних розробок. Крос-платформні інструменти, такі як Xamarin, дозволяють використовувати програми, написані на C #, майже на будь-яких мобільних пристроях.

C # широко використовується для створення ігор за допомогою ігрового механізму Unity, який є найпопулярнішим ігровим механізмом на сьогодні. Більше третини найкращих ігор створені за допомогою Unity, а активних користувачів ігор, створених за допомогою механізму Unity, нараховує приблизно 770 мільйонів. Unity також використовується для VR, оскільки 90% усіх Samsung Gear та 53% усіх ігор Oculus Rift VR розроблені за допомогою Unity.

C # - дуже популярний інструмент для створення цих програм, і тому робить чудовий вибір для будь-якого програміста, який сподівається проникнути в індустрію розробки ігор, або для тих, хто цікавиться віртуальною реальністю.

C # має безліч функцій (Рисунок 3.1.4), які полегшують навчання. Це мова високого рівня, порівняно легка для читання, де багато найскладніших завдань абстраговано, тому програміст не повинен про них турбуватися. Наприклад, управління пам’яттю вилучається з відповідальності користувача та обробляється схемою збору сміття.

Рисунок 3.1.4 — Можливості розробки мовою C#

Це також мова статичного типу, тому код перевіряється перед перетворенням на додаток. Це полегшує пошук помилок, що може бути особливо корисним для початківців.

Незважаючи на те, що синтаксис C # є більш послідовним та логічним, ніж C ++, все ще є чому повчитися. C # - це складна мова, і на її опанування може знадобитися більше часу, ніж на простіші мови, такі як Python. Це означає, що користувачам потрібно вивчити значну кількість коду для створення просунутих програм, що

може бути невдалим для деяких нових користувачів. C # забезпечує мовні конструкції для прямої підтримки цих концепцій, роблячи C # природною мовою, на якій можна створювати та використовувати програмні компоненти. З моменту свого зародження C # додав функції для підтримки нових навантажень та нових практик проектування програмного забезпечення.

Кілька функцій C # допомагають створювати надійні та довговічні програми. Збір сміття автоматично відновлює пам’ять, зайняту недосяжними невикористаними об’єктами. Типи, що допускають відхилення, захищають від змінних, які не посилаються на виділені об'єкти. Обробка винятків забезпечує структурований та розширюваний підхід до виявлення та відновлення помилок. Лямбда-вирази підтримують методи функціонального програмування. Синтаксис інтегрованого мовного запиту (LINQ) створює загальний шаблон роботи з даними з будь-якого джерела. Мовна підтримка асинхронних операцій забезпечує синтаксис для побудови розподілених систем. C # має уніфіковану систему типів. Усі типи C #, включаючи примітивні типи, такі як int і double, успадковуються від одного кореневого типу об'єкта. Усі типи мають спільний набір загальних операцій. Цінності будь-якого типу можна зберігати, транспортувати та використовувати послідовно. Крім того, C # підтримує як визначені користувачем типи посилань, так і типи значень. C # дозволяє динамічно розподіляти об'єкти та зберігати в лінійці легкі конструкції. C # підтримує загальні методи та типи, які забезпечують підвищену безпеку та продуктивність типу. C # надає ітератори, які дозволяють реалізаторам класів колекцій визначати власну поведінку для клієнтського коду.

C # робить акцент на встановленні версій, щоб забезпечити розвиток програм та бібліотек з часом сумісним способом. Аспекти дизайну C #, на які безпосередньо вплинули міркування щодо версій, включають окремі модифікатори віртуальних та перевизначених, правила дозволу на перевантаження методів та підтримку явних оголошень членів інтерфейсу.

3.1.5 Платформа .NET

.NET - це розробка програмного забезпечення та екосистема, розроблена та підтримувана корпорацією Майкрософт для спрощення проектування робочих столів та веб-додатків. Це популярна безкоштовна платформа, яка в даний час використовується для багатьох типів додатків, оскільки забезпечує середовище програмування для більшості етапів розробки програмного забезпечення. .NET найкраще підходить для підприємств, які шукають широкий спектр функцій, таких як веб-служби, настільне програмне забезпечення та підтримка хмарної інфраструктури

У 2014 році корпорація Майкрософт оголосила про значні зміни в існуванні. NET, представивши. NET Core (Рисунок 3.1.5), нову крос-платформну, зручну для хмар та версію з відкритим кодом. .NET Core вийшов у 2016 році, ставши основною технологією для розробки нових проектів. Поступово Microsoft почала переносити наявні служби для роботи з Core. Деякі, які не отримали офіційних портів, такі як Windows Communication Foundation (WCF), замінили альтернативи, отримані від спільноти

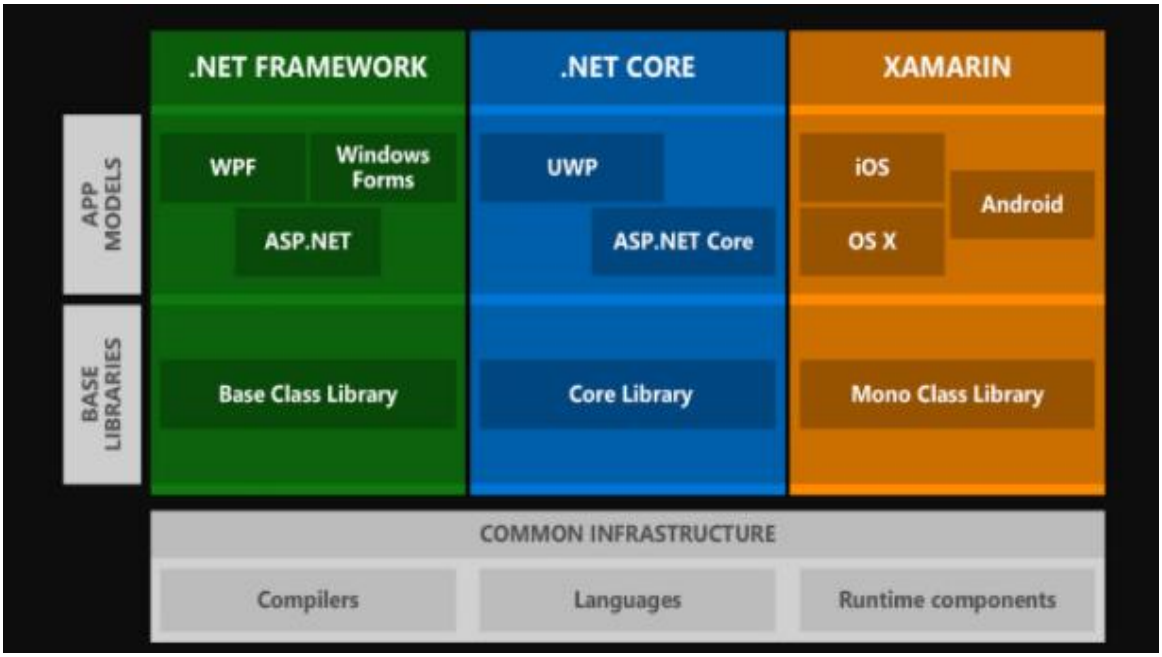

Рисунок 3.1.5 — Структура платформи

Двома основними компонентами .NET Framework є середовище загальної мови та бібліотека класів .NET Framework.

Common Language Runtime (CLR) - це механізм виконання, який обробляє запущені програми. Він надає такі послуги, як управління потоками, збір сміття, безпека типу, обробка винятків тощо.

Бібліотека класів надає набір API та типів для загальної функціональності. Він надає типи для рядків, дат, чисел тощо. Бібліотека класів включає API для читання та запису файлів, підключення до баз даних, малювання тощо.

Програми .NET написані мовою програмування C #, F # або Visual Basic. Код компілюється в мовно-агностичну спільну проміжну мову (CIL). Скомпільований код зберігається у збірках - файлах із розширенням .dll або .exe.

Ідея CLR полягає в тому, щоб полегшити життя розробника. Крім того, це дозволяє інженерам розробляти системи з кількома мовами, оскільки CLR дозволяє їм спілкуватися та інтегрувати свою поведінку. Виконавець перевіряє необхідні версії застосованих служб, щоб переконатися, що всі залежності непошкоджені і код працює належним чином. (Рисунок 3.1.6)

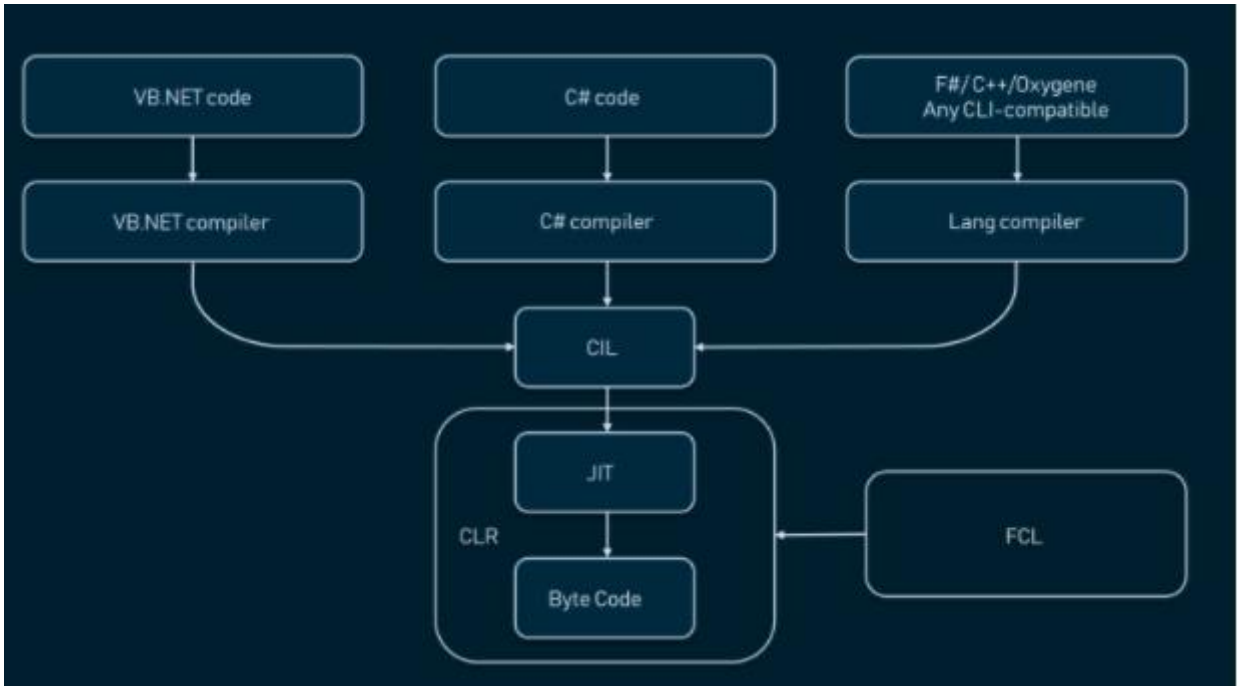

Рисунок 3.1.6 — Порядок викнонання коду

Коли програма запускається, CLR (Рисунок 3.1.7) бере збірку і використовує своєчасний компілятор (JIT), щоб перетворити її в машинний код, який може виконуватися на конкретній архітектурі комп'ютера, на якому вона працює.

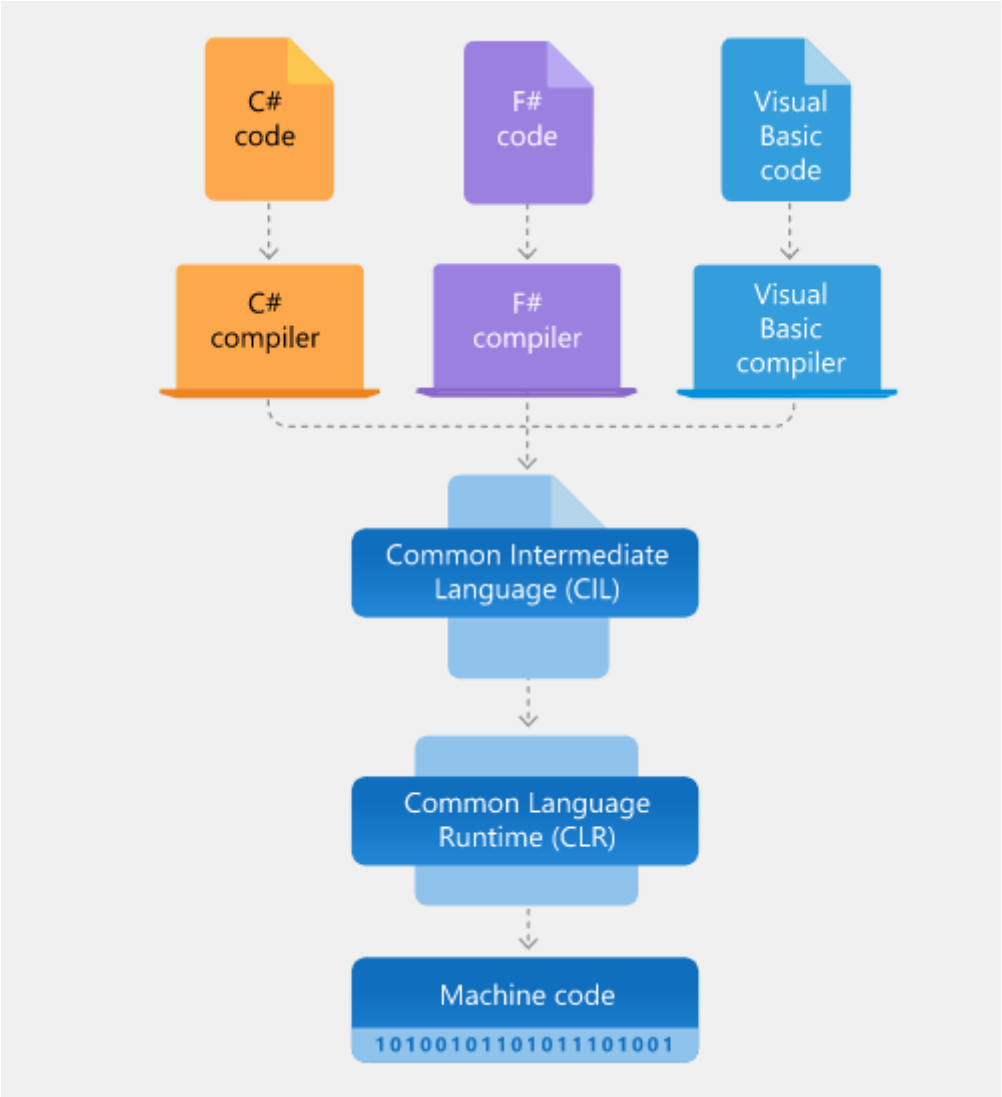

Рисунок 3.1.7 — Common Intermediate language

3.1.7 NuGet

"Сумісний" пакет означає, що він містить збірки, побудовані принаймні для однієї цільової .NET платформи, сумісної з цільовою структурою споживаючого

проекту. Розробники можуть створювати пакети, специфічні для одного фреймворку, як з елементами керування UWP, або вони можуть підтримувати ширший діапазон цілей. Щоб максимізувати сумісність пакету, розробники націлені на .NET Standard, який можуть споживати всі проекти .NET та .NET Core. Це найефективніший засіб як для творців, так і для споживачів, оскільки єдиний пакет (зазвичай містить одну збірку) працює для всіх споживаючих проектів.

З іншого боку, розробники пакетів, яким потрібні API за межами .NET Standard, створюють окремі збірки для різних цільових фреймворків, які вони хочуть підтримувати, і включають усі ці збірки в один пакет (що називається "багатоцільовим націлюванням"). Коли споживач встановлює такий пакет, NuGet витягує лише ті збірки, які необхідні проекту. (Рисунок 3.1.8) Це мінімізує розмір упаковки в кінцевій заявці та / або збірках, вироблених цим проектом. Звичайно, багатоцільовий пакет для його творця складніше підтримувати.

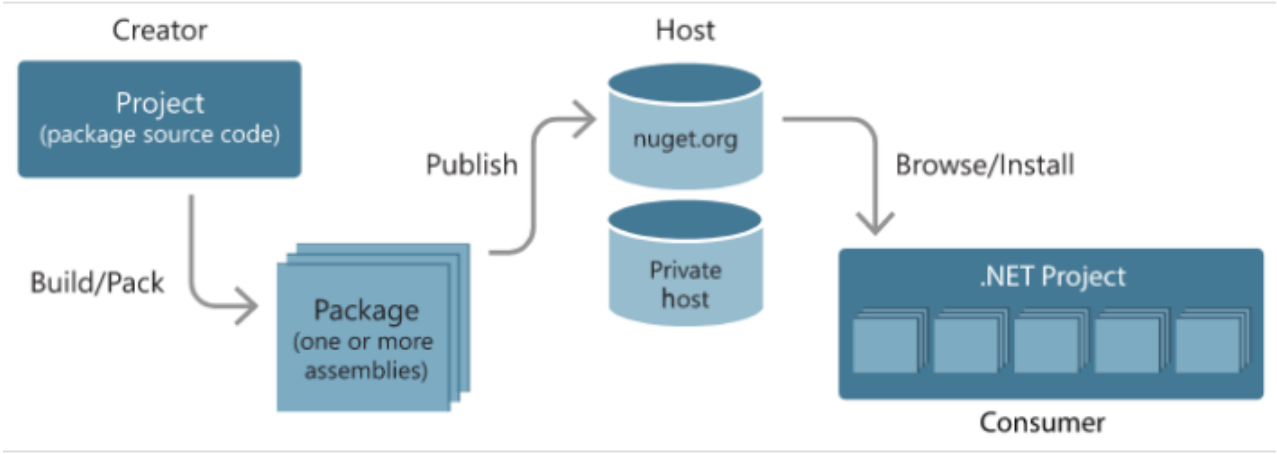

Рисунок 3.1.8 — Схема роботи пакету Nuget

Здатність легко спиратися на роботу інших - одна з найпотужніших функцій системи управління пакетами. Відповідно, більша частина того, що робить NuGet, - це управління цим деревом залежностей або "графіком" від імені проекту. Простіше кажучи, вам потрібно турбуватися лише про ті пакети, які ви безпосередньо використовуєте в проекті. Якщо будь-який з цих пакетів сам споживає інші пакети

(які, в свою чергу, можуть споживати інші), NuGet піклується про всі ці залежності нижчого рівня. (Рисунок 3.1.9)

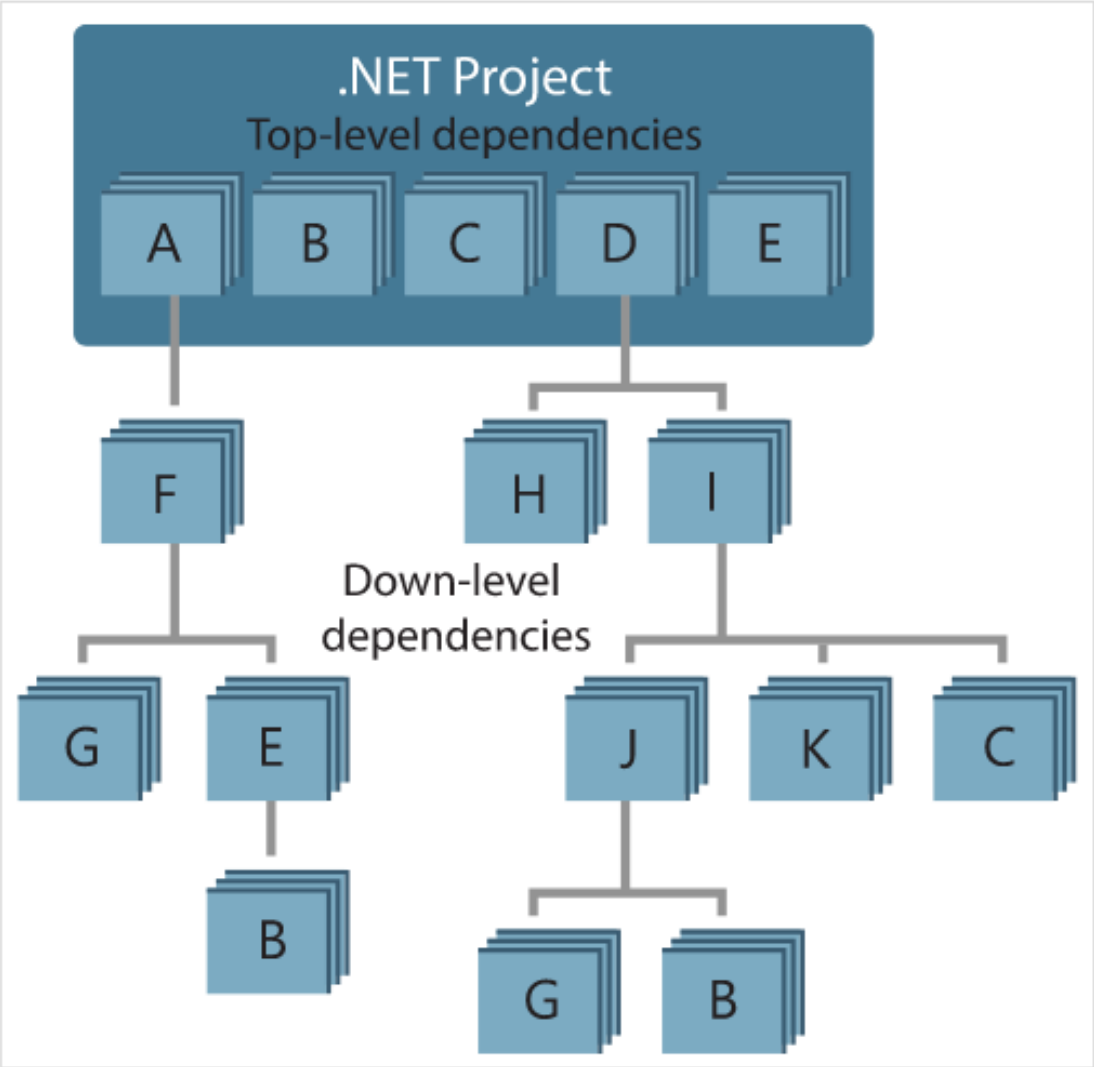

Рисунок 3.1.8 — Рівні залежностей пакетів

Комп'ютер, який отримує проект, такий як сервер збірки, який отримує копію проекту як частину автоматизованої системи розгортання, просто просить NuGet відновити залежності, коли вони потрібні. Для таких систем побудови, як Azure DevOps, передбачені кроки відновлення NuGet. Подібним чином, коли розробники отримують копію проекту (як при клонуванні сховища), вони можуть викликати такі команди, як відновлення nuget (NuGet CLI), відновлення dotnet (dotnet CLI) або Install-Package (консоль диспетчера пакетів), щоб отримати всі необхідні пакети. Visual Studio, зі свого боку, автоматично відновлює пакети під час побудови проекту

(за умови, що ввімкнено автоматичне відновлення, як описано у розділі Відновлення пакетів).

3.1.8 OpenGL

OpenGL - це відкритий і мобільний стандарт. Програми, написані за допомогою OpenGL можна переносити практично на будь-які платформи, отримуючи при цьому однаковий результат, будь це графічна станція або суперкомп'ютер. OpenGL звільняє програміста від написання програм для конкретного обладнання. Якщо пристрій підтримує якусь функцію, то ця функція виконується апаратно, якщо немає, то бібліотека виконує її програмно.

З точки зору програміста OpenGL - це програмний інтерфейс для графічних пристроїв, таких як графічні прискорювачі. Він включає в себе близько 150 різних команд, за допомогою яких програміст може визначати різні об'єкти і виробляти рендеринг. Говорячи більш простою мовою, ви визначаєте об'єкти, задаєте їх розташування в тривимірному просторі, визначаєте інші параметри задаєте властивості об'єктів, положення спостерігача, а бібліотека OpenGL подбає про те щоб відобразити все це на екрані. бібліотека OpenGL є лише відтворює (Rendering), і займається тільки відображенням 3Д об'єктів, вона не працює з пристроями введення (клавіатури, миші). Також вона не підтримує менеджер вікон.

OpenGL має добре продуману внутрішню структуру і досить простий процедурний інтерфейс. Незважаючи на це за допомогою OpenGL можна створювати складні і потужні програмні комплекси, витрачаючи при цьому мінімальний час в порівнянні з іншими графічними бібліотеками.

Основні можливості OpenGL:

- геометричні і растрові примітиви. На основі геометричних і растрових примітивів будуються всі об'єкти. З геометричних примітивів бібліотека надає: точки, лінії, полігони. З растрових: бітовий масив (bitmap) і образ (image);

- використання B-сплайнів. B-сплайни використовуються для малювання кривих по опорних точках.

- видові і модельні перетворення. За допомогою цих перетворень можна розташовувати об'єкти в просторі, обертати їх, змінювати форму, а також змінювати положення камери з якої ведеться спостереження.

Робота з кольором. OpenGL надає програмісту можливість роботи з кольором в режимі RGBA (червоний-зелений-синій-альфа) або використовуючи індексний режим, де колір вибирається з палітри.

- видалення невидимих ліній і поверхонь.

- подвійна буферизація. OpenGL надає як одинарну так і подвійну буферизацію. Подвійна буферизація використовується для того, щоб усунути мерехтіння при мультиплікації, тобто зображення кожного кадру спочатку малюється в другому (невидимому) буфері, а потім, коли кадр повністю намальований, весь буфер відображається на екрані.

- накладання текстури. Дозволяє надавати об'єктам реалістичність. На об'єкт, наприклад куля, накладається текстура (просто якесь зображення), в результаті чого наш об'єкт тепер виглядає не просто як куля, а як різнобарвний м'ячик.

Згладжування дозволяє приховати ступінчастість, властиву растровим дисплеям. Згладжування змінює інтенсивність і колір пікселів близько лінії, при цьому лінія виглядає на екрані без всяких зигзагів.

3.1.9 GLUT

Інструментарій OpenGL Utility Toolkit (GLUT) - це інтерфейс програмування з прив’язками ANSI C та FORTRAN для написання незалежних програм вікон OpenGL. Набір інструментів підтримує такі функції:

- кілька вікон для рендерінгу OpenGL.

- обробка подій, керована зворотним викликом.

- складні пристрої введення.

- режим “ простою " та таймери.

- простий, каскадний спливаюче меню.

- службові процедури для створення різних суцільних та дротяних каркасних об'єктів.

- підтримка растрових та обведення шрифтів.

- різні функції управління вікнами, включаючи управління накладами.

3.1.10 GLX

GLX 1.3 використовується в реалізації Unix OpenGL для управління взаємодією з системою X Window та кодування OpenGL у потік протоколу X для віддаленого візуалізації. Він підтримує: піксельні буфери для апаратного прискореного візуалізації поза екраном; доступні лише для читання малюнки для попередньої обробки даних у позаекранному вікні та прямого введення відео; та FBConfigs, більш потужний та гнучкий інтерфейс для вибору конфігурацій буфера кадру, що лежить в основі вікна візуалізації OpenGL.

3.1.11 GLU

GLU - це службова бібліотека OpenGL. Це набір функцій для створення міп-карт текстури з базового зображення, відображення координат між екраном та простором об’єктів, а також малювання квадричних поверхонь та NURBS.

3.1.12 Visual Studio – редактор коду

Microsoft Visual Studio застосовується для розробки комп'ютерних програм, а також веб-сайтів, веб-програм, веб-сервісів та мобільних додатків. Visual Studio використовує платформи для розробки програмного забезпечення Microsoft, такі як Windows API, Windows Forms, Windows Presentation Foundation, Windows Store та Microsoft Silverlight. Він може створювати як власний код, так і керований код.

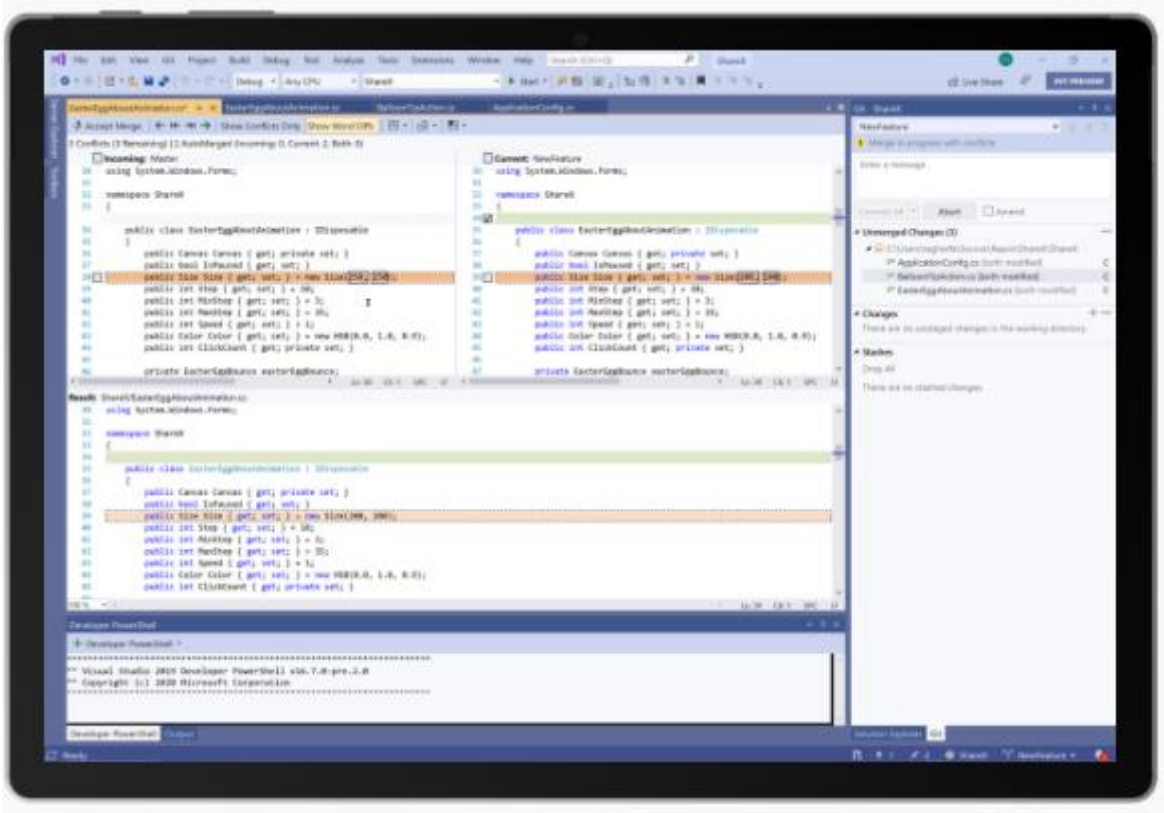

Рисунок 3.1.5.1 – Visual Studio

Visual Studio (Рисунок 3.1.5.1 – Visual Studio) включає редактор коду, що підтримує IntelliSense (компонент завершення коду), а також рефакторинг коду. Інтегрований налагоджувач працює як налагоджувач вихідного рівня, так і налагоджувач машинного рівня.

Visual Studio підтримує 36 різних мов програмування та дозволяє редактору коду та налагоджувачу підтримувати (різною мірою) майже будь-яку мову програмування за умови існування послуги, що стосується конкретної мови. До вбудованих мов належать C, C ++, C ++ / CLI, Visual Basic .NET, C #, F #, JavaScript, TypeScript, XML, XSLT, HTML та CSS. Підтримка інших мов, таких як Python, Ruby, Node.js та M, серед інших, доступна через плагіни. Раніше підтримувались Java (і J #).

4.2 Налаштування технологій та їх взаємодія

Окрім вказаних вище технологій, під час розробки “StereoAnaglyph” було використано додаткові бібліотеки та технології, створені на їх основі.

Перше, що потрібно було зробити - це створити контекст OpenGL і вікно додатків для рисування. Однак дані операцій є специфічними для кожної операційної системи, тому OpenGL цілеспрямовано намагається абстрагуватися від них. Це означає, що ми самостійно повинні створити вікно, визначити контекст та обрати користувацький ввід.

GLFW (англ. «Graphics Library FrameWork») - це бібліотека, написана на мові Сі, спеціально призначена для роботи з OpenGL. Бібліотека GLFW пропонує нам усі необхідні інструменти, які потребують рендерингу на екрані різних об'єктів. Завдяки цьому ми можемо створити контекст OpenGL, визначити параметри вікна та обробити користувацький ввід, що повністю заповнює коригування з нашими цілями.

CMake - це інструмент, який із використанням збережених певних сценаріїв може створювати колекції файлів вихідного коду, генерувати файли проектів / рішень під вибранною користувачем IDE (наприклад, для Visual Studio)

4.ОПИС ПРОГРАМНОЇ РЕАЛІЗАЦІЇ

4.1 Структурна схема

Структурна схема для системи OpenGL зображено на рисунку 4.1.

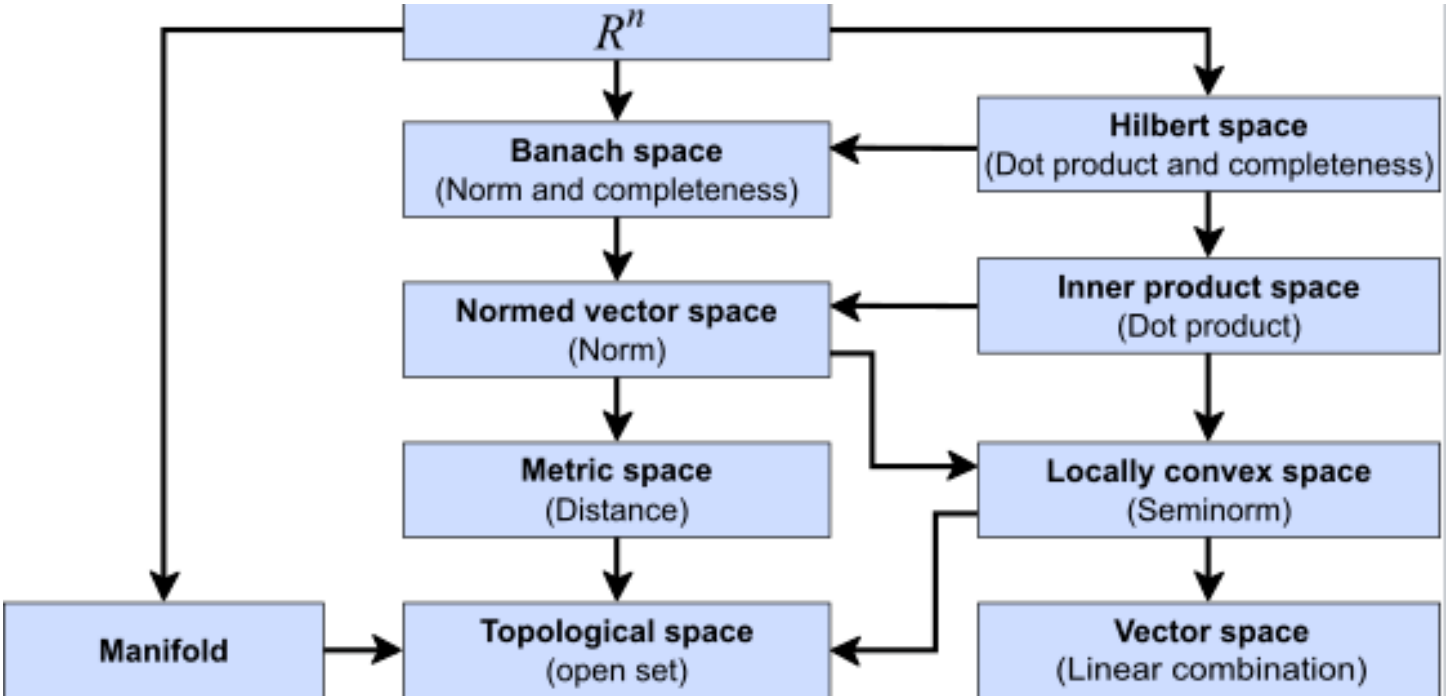

Рисунок 4.1 – Структурна схема

Користувач має змогу змінювати налаштування системи. Користувач взаємодіє з інтерфейсом та змінює початкові налаштування.

Також реалізований функціонал зміни положення фігури в 3-Д просторі за допомогою клавіатури.

Камера визначається своїм розташуванням, напрямком огляду, вектором вгору, відокремленням очей, відстанню до нульового паралакса (див. Fo нижче) та ближньою та дальньою ріжучими площинами. Положення, відрив очей, нульова відстань паралакса та площини різання найбільш зручно задавати в координатах

моделі, напрям та вектор вгору є ортонормальними векторами. Що стосується параметрів для регулювання стереоскопічного перегляду, то я б стверджував, що відстань до нульового паралакса є найбільш природним, і не тільки він безпосередньо пов'язаний з масштабом моделі та взаємним розташуванням камери, але також має пряме відношення до стереоскопічний результат

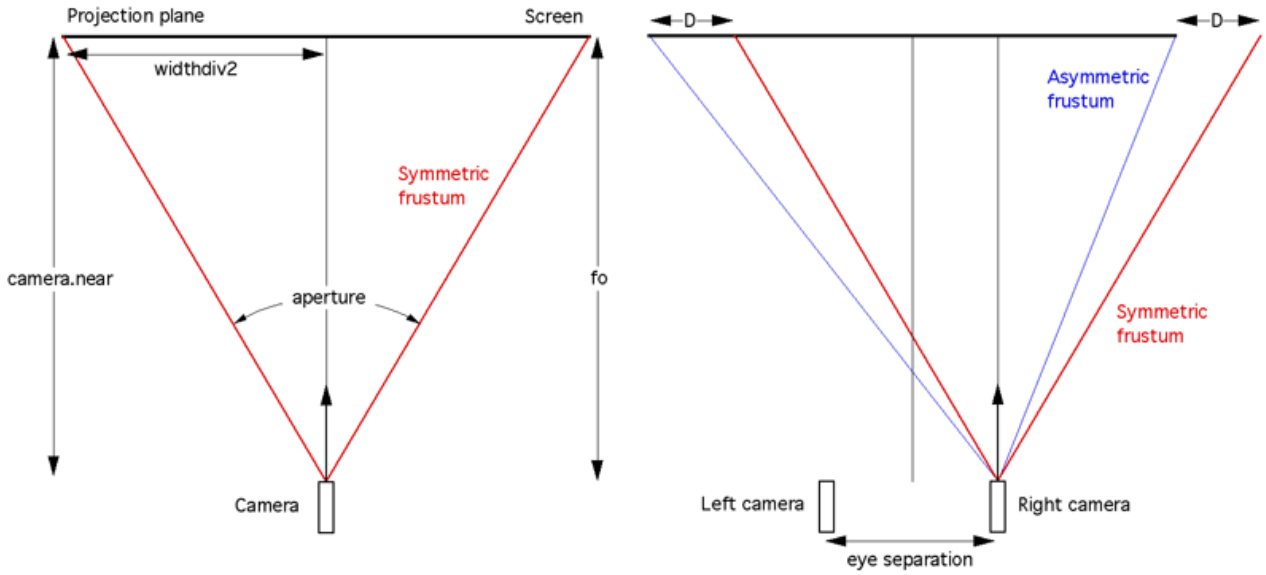

Рисунок – 4.1.1 – зміна положення камери

Наведена діаграма (вид зверху на дві камери) призначена для ілюстрації того, як розраховується сума, на яку компенсуються плоди. Зверніть увагу, що існує лише горизонтальний паралакс. Це має бути керівництвом для програмістів OpenGL, тому що існують деякі припущення, що стосуються OpenGL, які можуть не відповідати іншим API. Розділення очей перебільшено, щоб зробити діаграму чіткішою.

4.2 UML діаграма системи

UML діаграма системи наведена на рисунку 4.2.

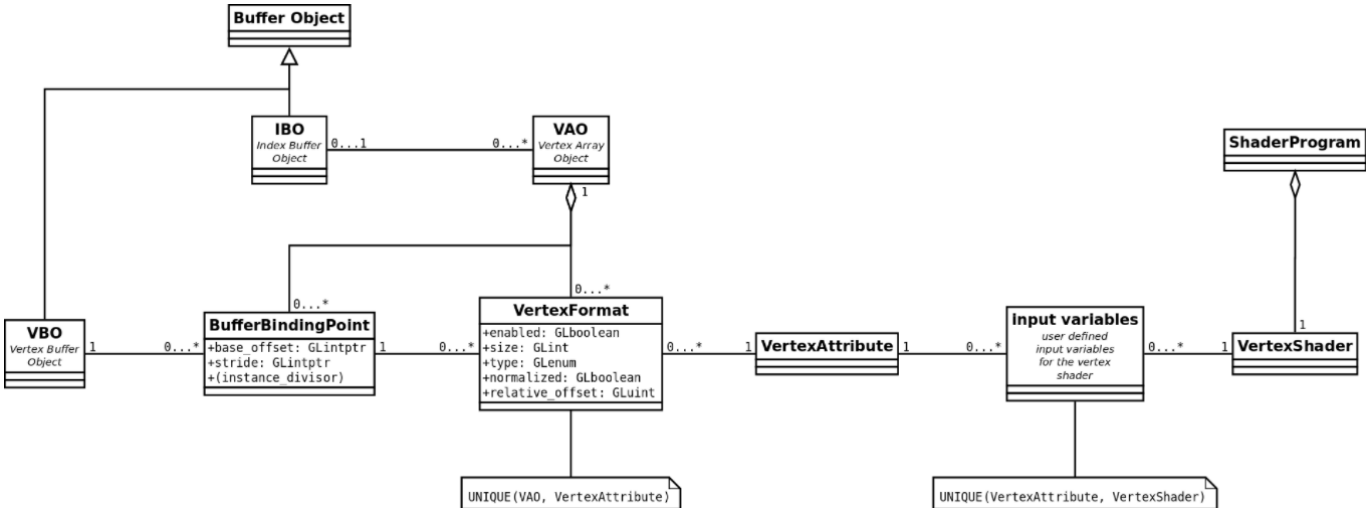

Рисунок 4.2.1 – UML діаграма

Діаграма UML показує уніфіковану візуальну презентацію системи UML (Unified Modeling Language), що має на меті дозволити розробникам або власникам бізнесу зрозуміти, проаналізувати та визначити структуру та поведінку своєї системи.

Наразі діаграма UML стала одним із найпоширеніших інструментів моделювання бізнес-процесів, що також є дуже важливим для розробки об’єктно-орієнтованого програмного забезпечення.

5. МЕТОДИКА РОБОТИ З ПРОГРАМОЮ

5.1 Інтерфейс та меню

Було прийнято рішення розробити інтерфейс якомога простим, щоб його використовували не тільки вмілі користувачі, але й будь-яка людина. Інтерфейс виконаний у вигляді таблиці та відкривається при натисканні правою кнопкою миші на простір (Рисунок 5.1.1)

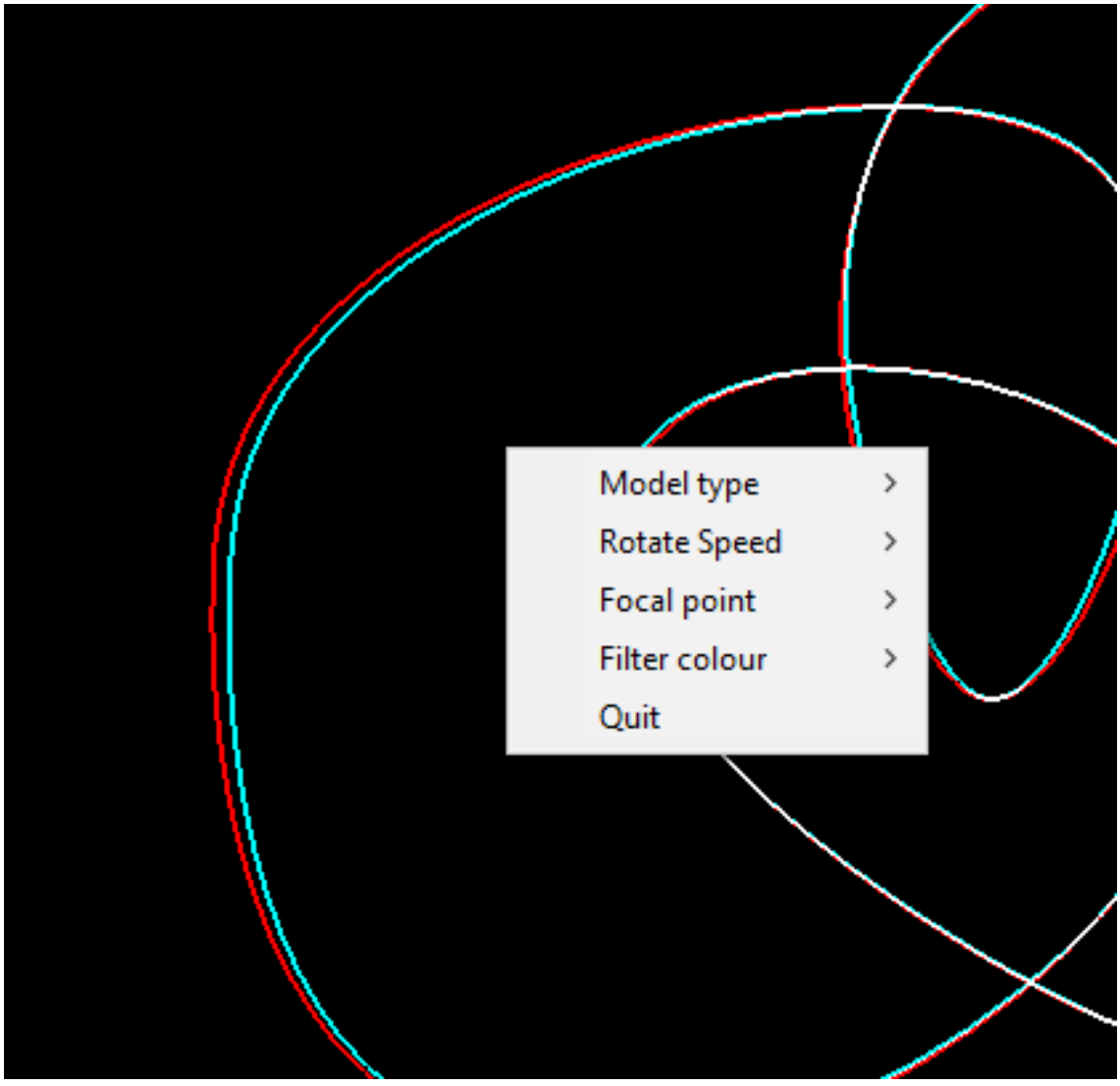

Рисунок 5.1.1 – Інтерфейс-меню користувача для роботи з системою

В меню є такі вибірки як:

- Model type – вибір каскадної моделі
- Rotate Speed – швидкість обертання

- Focal point – точка перегляду
- Filter colour – вибір кольоровогу спетрку

При наведенні на колонку таблиця розгортається (Рисунок 5.1.2)

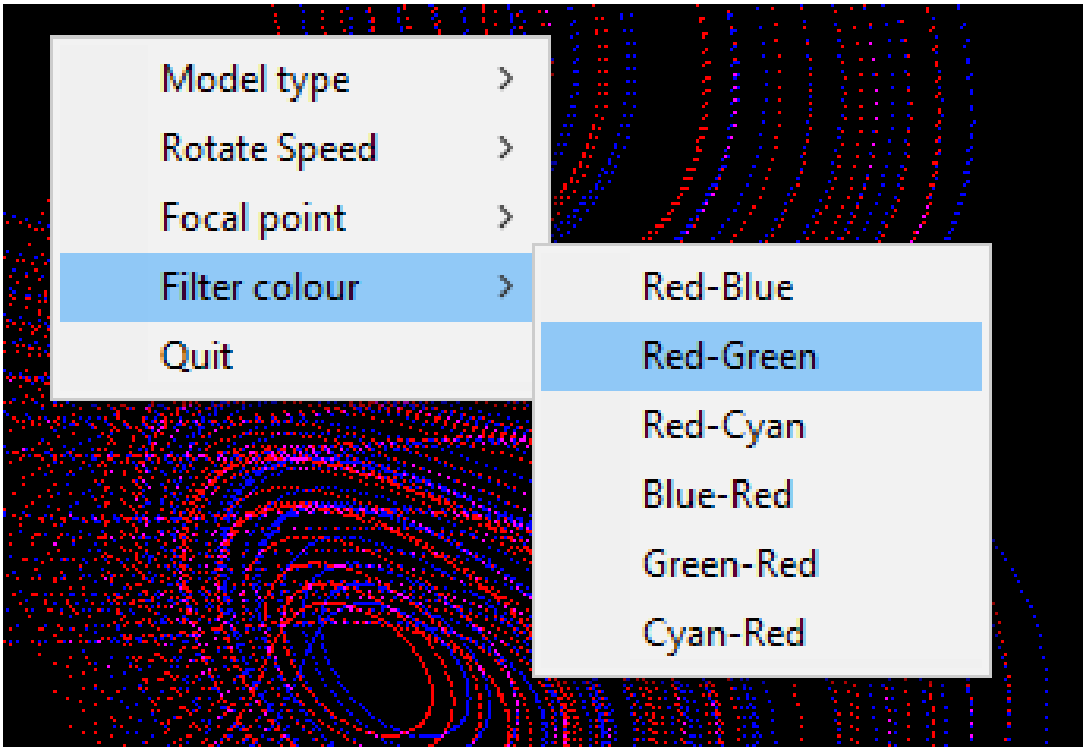

Рисунок 5.1.2 – Колонки інтерфейсу

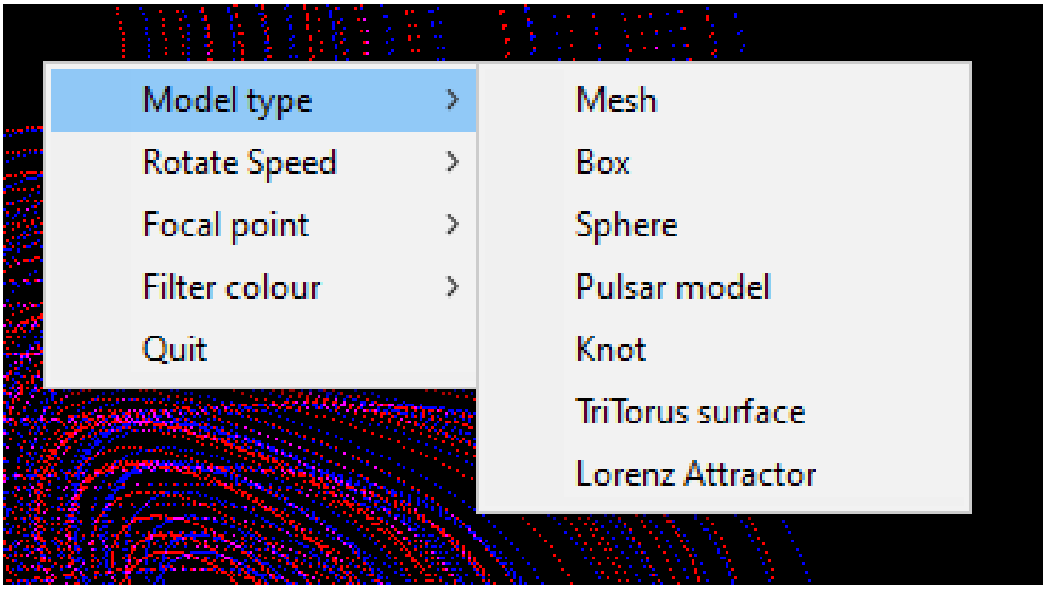

Рисунок 5.1.3 – Колонки інтерфейсу

5.2 Каркасні моделі

В програмі побудовані такі каркасні моделі, як:

- куб (Рисунок 5.2.1)

- триториус(Рисунок 5.2.2)

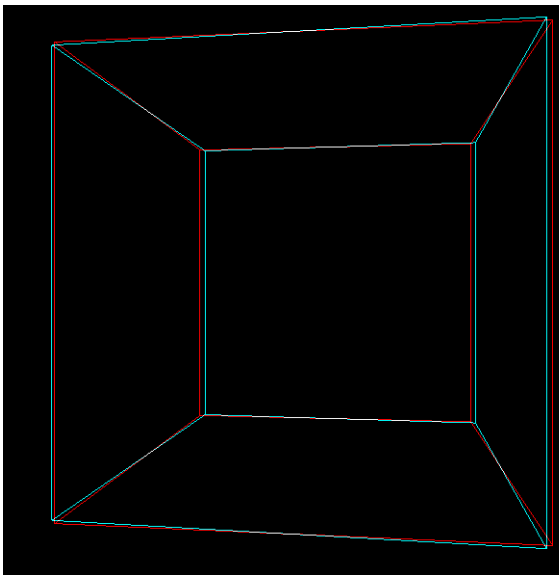

Рисунок 5.2.1 – Куб

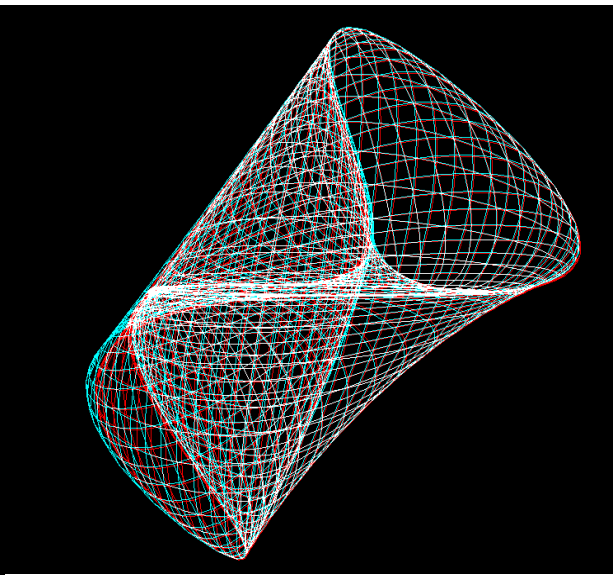

Рисунок 5.2.2 – Триториус

- сфера(Рисунок 5.2.3)

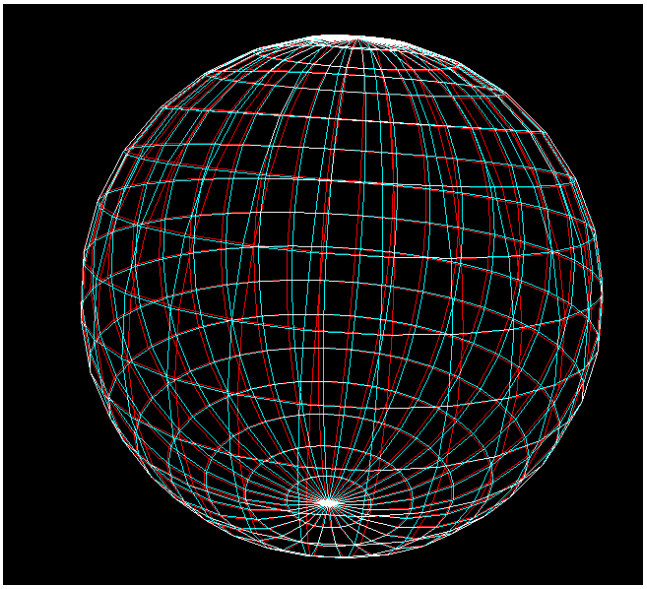

Рисунок 5.2.3 – Сфера

- кнут(Рисунок 5.2.4)

- пульсар(Рисунок 5.2.5)

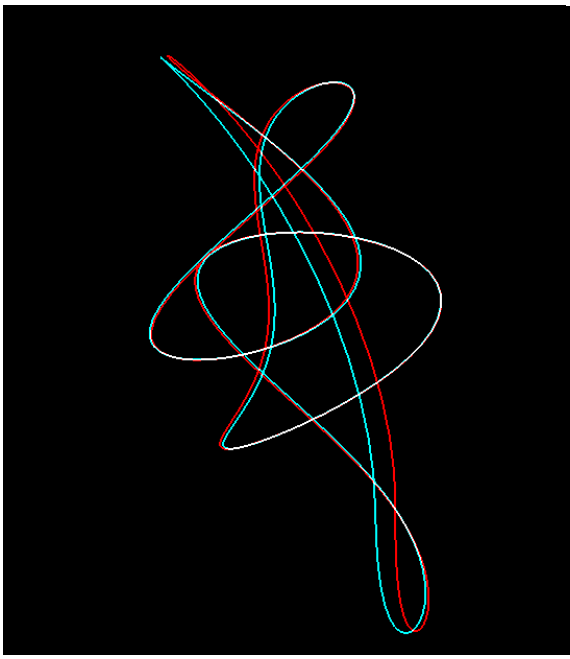

Рисунок 5.2.4 – Кнут

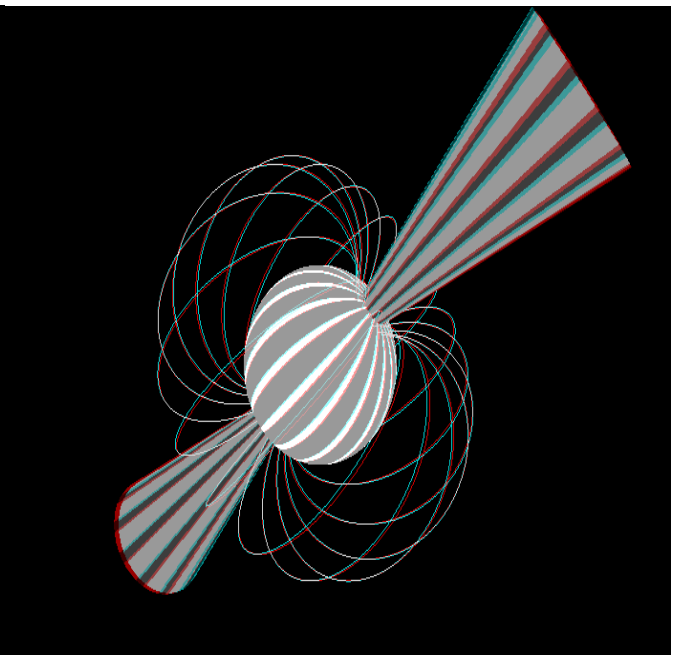

Рисунок 5.2.5 – Пульсар

- лоренз(Рисунок 5.2.6).

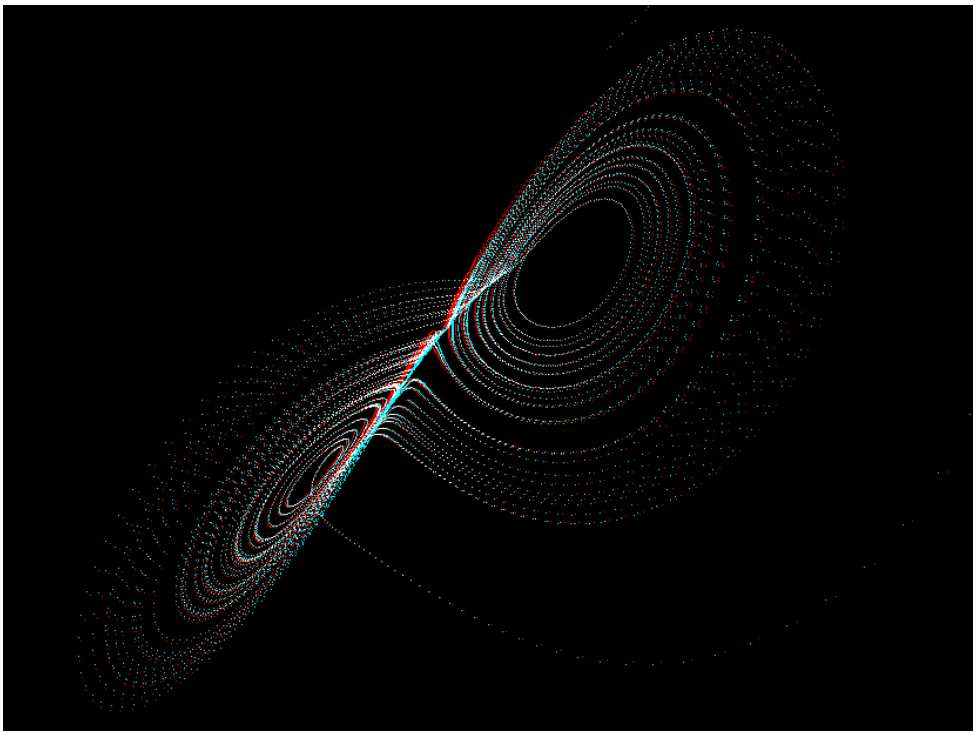

Рисунок 5.2.6 – Лоренз

5.3 Кольорові спектри

Однією з задач розробки застосунку було дослідження існуючих кольорових спетрів. Без окулярів можна побачити кольорову стереопару. В даному випадку використовується палітра «red-cyan» (Рисунок 5.3.1). При однаковому положенню точок вектору у вимірі можна побачити реалізацію сходження стереопари (білий колір) - коли один з векторів поверх іншого змінюється колір на інший, спеціально підібраний для кращого бачення стерео явища людським оком при зміні положення векторів, на прикладі сфери: (Рисунок 5.3.1)

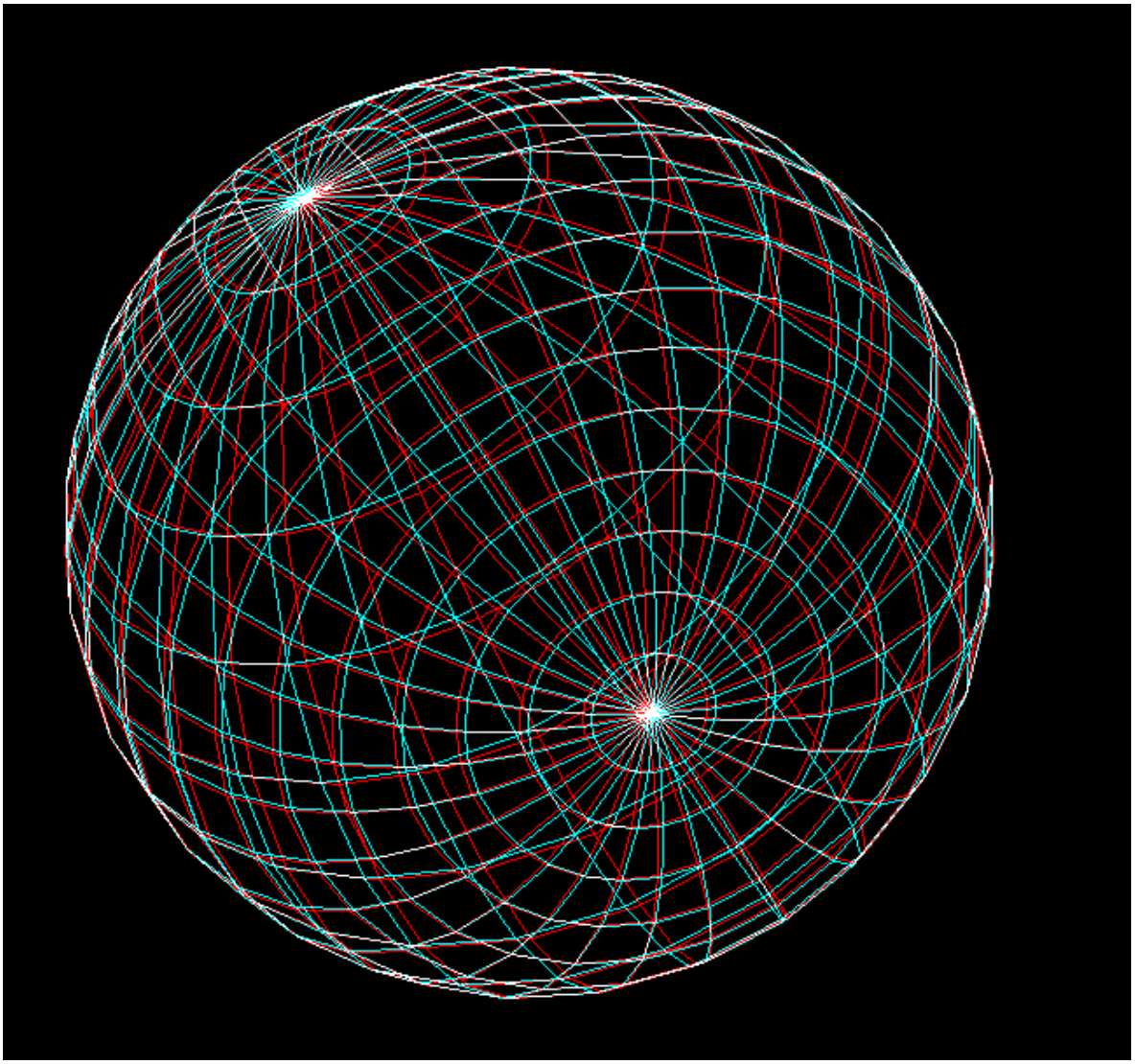

Рисунок 5.3.1 – палітра red-cyan

Існує багато варіантів анагліфних окулярів, наприклад: червоно / сині (red / blue, R / B), червоно / зелені (red / green, R / G), червоно / синьо - зелені (red / cyan, R / G + B), жовто / сині (yellow / blue, R + G / B, ColorCode 3D), зелено / червоно - сині (пурпурний) (green / magenta, G / R + B).

Однією з основних задач системи є коректне відображення для ока людини кольорової стереопари. Взаємодія та зміна кольору векторів в залежності від кольорового спектру та координат у вимірі кожного з векторів. При вільному перебуванні в 3-Д вимірі, окремо для кожного вектору стереопари, було підібрано колір під всі існуючі кольорові анагліфні окуляри. Нижче наведені приклади застосування існуючих анагліфних кольорів (Рисунок 5.3.2) (Рисунок 5.3.3) (Рисунок 5.3.4) :

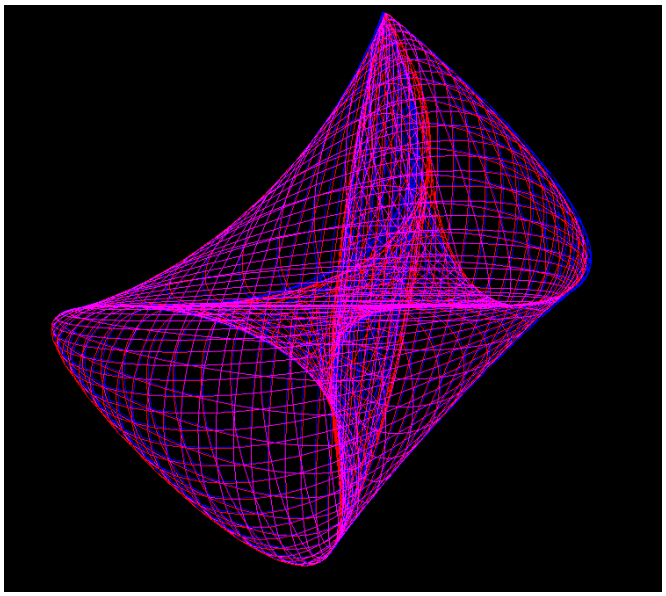

Рисунок 5.3.2 – палітра red-blue

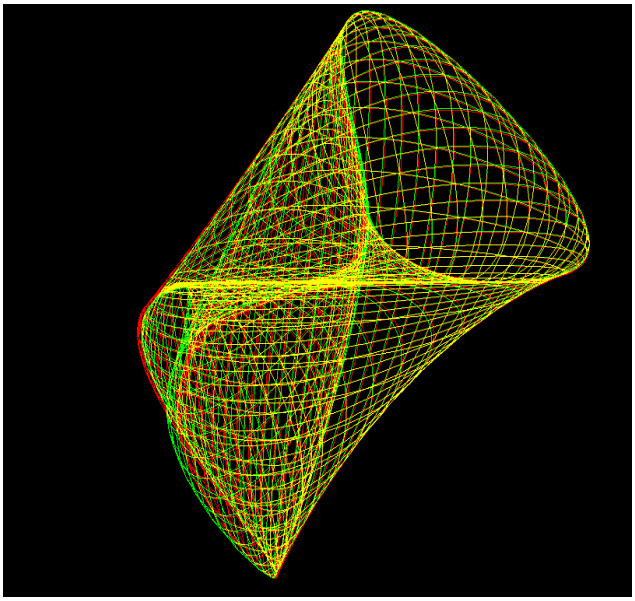

Рисунок 5.3.3 – палітра green-red

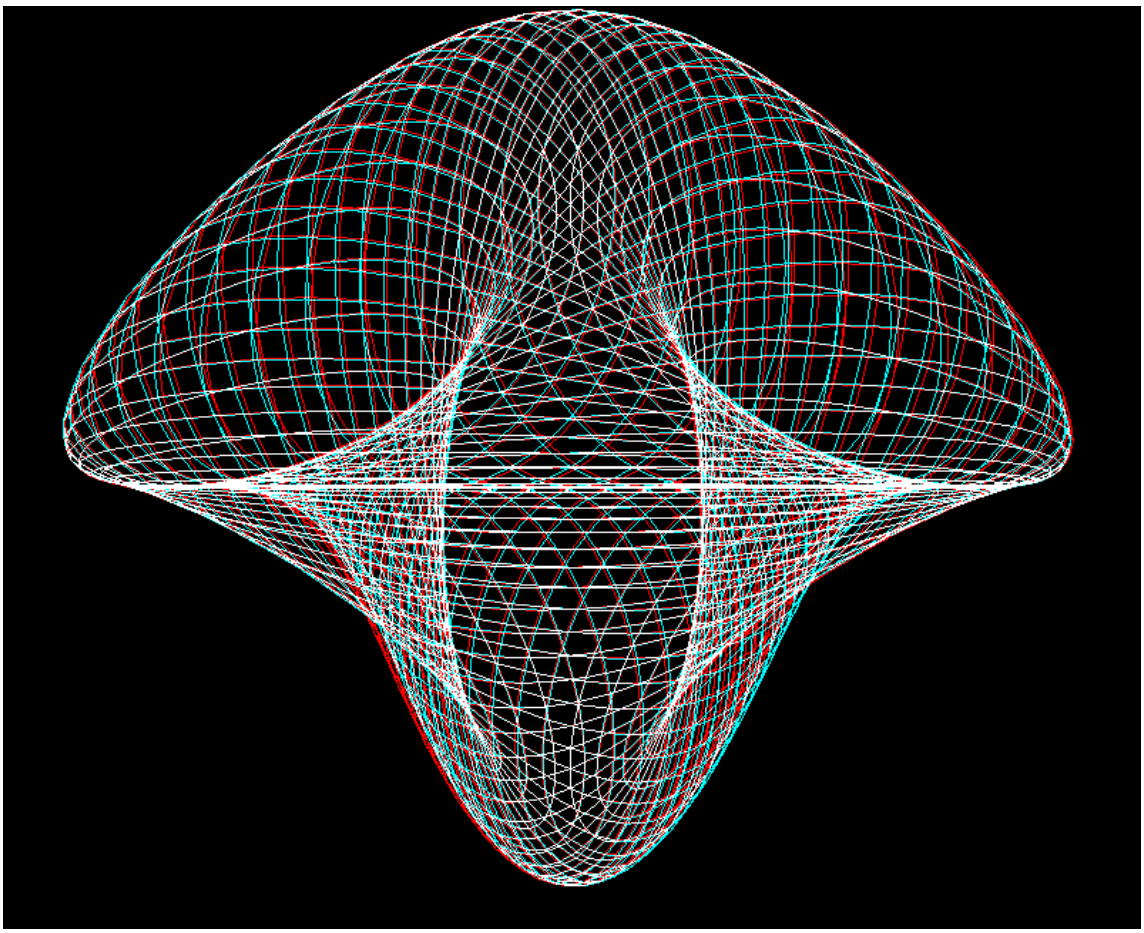

Рисунок 5.3.4 – палітра red-blue

5.4 Эффект парралаксу

Бачити предмети в обсязі дозволяє бінокулярний зір, для якого необхідні два ока. Паралакс - це зміна видимого положення об'єкту відносно віддаленого фону в залежності від положення спостерігача.

Кожне око окремо бачить плоске (двомірне) зображення. Так як ока два і вони розташовані на деякій відстані один від одного (58-72 мм у дорослих людей), в мозок надходять зображення одного і того ж предмета з двох точок зору, так званий паралакс. В результаті їх обробки формується об'ємна картинка.

Ефект паралаксу (Рисунок 5.4.1): фон зсувається повільно, а об'єкт - сильніше і швидше, тому здається, що зображення об'ємне. Щоб створити ефект паралакса, потрібно розділити картинку на кілька шарів і задати їм різну швидкість і діапазон руху в залежності від скролла або переміщення курсору

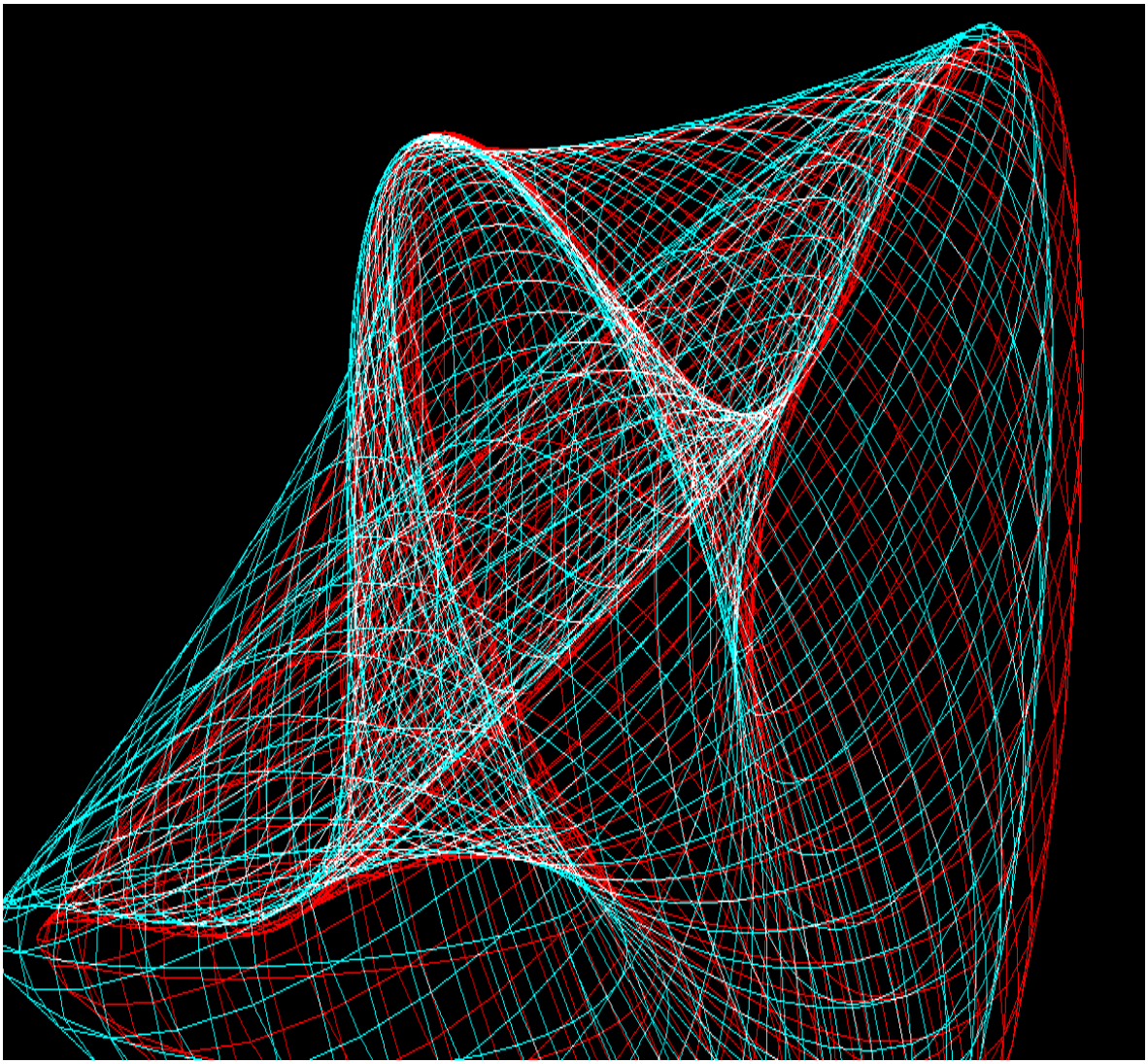

Рисунок 5.4.1 – ефект паралаксу

5.5 Впровадження зовнішніх комп’ютерних приладів

В програмі впроваджена взаємодія з мишкою та клавіатурою, а саме: обертання каркасного зображення за допомогою мишки, зсув по осям (Рисунок 5.5.1), обертання за допомогою клавіатури (Рисунок 5.5.2)

```
    case '[':
        RotateCamera(0, 0, -1);
        break;
    case ']':
        RotateCamera(0, 0, 1);
        break;
    case 'i':
    case 'I':
        TranslateCamera(0, 1);
        break;
    case 'k':
    case 'K':
        TranslateCamera(0, -1);
        break;
    case 'j':
    case 'J':
        TranslateCamera(-1, 0);
        break;
    case 'l':
    case 'L':
        TranslateCamera(1, 0);
        break;
    }
}
```

Рисунок 5.5.1 – код впровадження клавіш зсуву

```
void GiveUsage(char* command)  // changed
{
    fprintf(stderr, "          -h   this text\n");
    fprintf(stderr, "          -f   full screen mode\n");
    fprintf(stderr, "Key Strokes\n");
    fprintf(stderr, "  arrow keys   rotate left/right/up/down\n");
    fprintf(stderr, "  left mouse   rotate\n");
    fprintf(stderr, "middle mouse   roll\n");
    fprintf(stderr, " right mouse   activates the menus\n");
    fprintf(stderr, "           i   translate up\n");
    fprintf(stderr, "           k   translate down\n");
    fprintf(stderr, "           j   translate left\n");
    fprintf(stderr, "           l   translate right\n");
    fprintf(stderr, "           [   roll clockwise\n");
    fprintf(stderr, "           ]   roll anti clockwise\n");
    fprintf(stderr, "           w   write a frame as a PPM file\n");
    fprintf(stderr, "           r   toggle movie recording on and off\n");
    fprintf(stderr, "           q   quit\n");
    exit(-1);
}
```

Рисунок 5.5.2 – код впровадження клавіш обертання

ВИСНОВКИ

Під час проходження практики було покращено навички розробки 3-Д моделей, починаючи від створення 3-Д простору до побудови векторів для стереопари. Освоїв навички побудови стереопари для візуалізації методу анагліфної сепарації. Ознайомився з такою технологією, як OpenGL та її основними технологіями. Навчився інтегрувати в програму взаємодію користувача с 3-Д простором, інтегрував налагодження системи через користувача. Пізнав для себе явище «сепарації» , дослідив існуючі методи.

Я створив програму , яка вирішує задачі поставлені мною:
візуалізація анагліфного методу сепарації, обертання геометричних фігур в 3-Д просторі, налагодження інтерфейсу для користувача, динамічна зміна кольору стереопари, інтеграція виду «перед екраном», «за екраном» та «на екрані», доповнив програму взаємодією з зовнішніми комп’ютерними приладами для обертання та зсуву моделі, які в подальшому можуть бути використані в навчальних цілях в наукових закладах.

Отже, практика покращила знання різноманітних технологій, що використовуються під час розробки різноманітних 3-Д додатків.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Gruts Yu.N., Jung-Young Son Stereoscopic operators and its application // Proccedings of The 6th International Workshop on 3-D Imaging Media Technology and The 5th Photonic Information Processing Conference.-2000.-Vol.6,N1,pp.34-38.
2. Yu.N.GRUTS Method of Model-Computer Stereomodelling // Engineering Simulation, 1997,Vol. 14, pp. 681-690.
3. OpenGL [Електронний ресурс] – Режим доступу: https://www.opengl.org//
4. C [Електронний ресурс] – Режим доступу: https://www.programiz.com/c-programming
5. Налаштування OpenGL [Електронний ресурс] – Режим доступу: https://ravesli.com/urok-2-podgotovka-k-pervomu-proektu-opengl-nastrojka-glfw-cmake-i-glad/
6. Наукова стаття [Електронний ресурс] – Режим доступу: http://paulbourke.net/stereographics/stereorender/
7. Visual Studio [Електронний ресурс] – Режим доступу: https://visualstudio.microsoft.com/
8. Nvidia [Електронний ресурс] – Режим доступу: https://www.ixbt.com/video3/3dvision.shtml
9. Windows 10 [Електронний ресурс] – Режим доступу: https://www.microsoft.com/en-us/windows/features?activetab=NewPopular