МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ОДЕСЬКИЙ ДЕРЖАВНИЙ ЕКОЛОГІЧНИЙ УНІВЕРСИТЕТ

Факультет комп'ютерних наук управ-
ління та адміністрування
Кафедра інформаційних технологій

Бакалаврська кваліфікаційна робота

на тему: Візуалізація 3D моделі ОДЕКУ

Виконав студент 4 курсу групи К-42
Напрямок 6.05.01.01 Комп'ютерні науки
Воробйова Оксана Анатоліївна

Керівник к.т.н., доцент
Трегубова Ірина Анатоліївна

Консультант

Рецензент к.ф-м.н.
Витавецкая Лариса Анатоліївна

Одеса 2019

ЗМІСТ

ВСТУП

Актуальність теми. Візуалізація є актуальною проблемою сучасності, так як вона надає змогу людині заочно побачити об'єкт. Концепт об'єкта або проекту, в основному, демонструють за допомогою відеороликів і картинок, зроблених на основі 3D-графіки. Сценарії в готовому вигляді і демонструють лише те, що компанія-представник вважає за необхідне. Це обмежує перегляд, так як в статичних зображеннях немає можливості відхилитися від сюжету і уважно оглянути деталі. Тривимірна графіка реального часу на сьогоднішній день ефективно застосовується в багатьох предметних областях.

Потужність комп'ютерних обчислень дозволяє обробляти досить складні сцени в режимі реального часу без втрати швидкості і якості відображення. Ці можливості привели до появи інтересу до тривимірної візуалізації з боку фахівців з різних сфер діяльності. Так, в області архітектури та містобудування все більш широке застосування знаходять віртуальні прогулянки по місту та приміщенням будвель.

Фотореалістічна реконструкція об'єктів, що дозволяє на етапі проектування ефективно працювати із замовником, використовувати 3D моделювання в процесі навчання, в музейних, реставраційних, рекламних, комерційних проектах також є сучасним і перспективним.

Особливе значення 3D технологія набуває в задачах інтерактивного проектування інженерних підсистем в системах автоматизації, іменованих «розумний будинок» (smart house). Наприклад, нещодавно сгорівші частини на весь світ відомого собору Нотр Дам будуть відновлювати за розробками детальних частин американського професора-архітектора Ендрю Уілсон, та за детальною моделью собору із іметиної гри Assassin’s Creed: Unity.

Також багато зображенть, які ми бачимо по телевізору чи в кінотеатрах є роботами 3D моделювання, 3D анімацій і рендрінгу 3D моделей. Треба підкреслити яскраву та професійну розробку у фільмі Аватар більші частини

якого були побудовані з допомогою 3D графіки. Зараз компанія Microsoft розроблює софт для лікарів, щоб полегшити проведення складних операцій; для пілотів, щоб удосконалити управління літаками; для архітекторів, щоб спростити роботу з клієнтами та інших. Більшість заставок для телевізійних программ також зроблені завдяки 3D. Винаходцем Скотом Крампом створені 3D принтери, що надали можливісті одержувати реальни 3D об'єкти з використанням різноманітних матеріалів. Цей потужний винахід став частиною сучасності бо дав можливість не тільки одержувати потрібні технічні об'єкти а також врятовувати життя та здоров'я багатьом хворим людям, здійснюючи операції по трансплантації власних не працюючих органів на штучні органи, вироблені з живих тканин. Тому можна сказати, що 3D моделювання далеко за межі мультфільмів. Воно є вагомою частиною нашого сучасного життя.

Метою роботи є розробка інтерактивної візуалізації для Одеського Державного Екологічного Університету.

Для досягнення мети треба вирішити наступні поставлені задачі:

- змога для студентів першокурсників, чи тих же хто перевівся з коледжів дізнатися місце знаходженні аудиторій для лекцій та практик;
- місце проведення загально університетських зборів та заходів;
- розробити плани пожежної безпеки в інтерактивних форматах для кращого запам'ятовування в порівнянні з вже існуючими стандартними схемами.

Об'єкт дослідженні – процесс використання сучасних графічного софту для сворення інтерактивної візуалізації Одесського Державного Екологічного Университету.

Предмет дослідження – теоритичні, методологічні, программні та прикладні аспекти 3D для створення інтерактивної візуалізації Одесського Державного Екологічного Университету.

Методами дослідження є аналітичні, програмні, порівняльні методи аналізу та синтезу інтерактивної візуалізації об'єкта

Практична значущість отриманих результатів полягає у тому що створена інтерактивна візуалізація демонструє більш наглядніше структуру розміщення адміністративних приміщень, кафедр, лекційних та учбових аудиторій.

1 АНАЛІТИЧНА ЧАСТИНА

1.1 Аналіз програмного забезпечення

У наш час CGI-образи (від слів Computer Graphics Imagery – зображення створене на комп'ютері) оточують нас на кожному кроці: САПР та ділова графіка, мультимедіа, анімація, кіно, комп’ютерні ігри, web-дизайн, поліграфії та в інших сферах. Комп'ютерна графіка перетворилася з вузькоспеціальної області інтересів вчених-комп'ютерників в справу, якій прагнуть присвятити себе безліч людей. Серед програмних комплексів тривимірної графіки, призначених для роботи на комп'ютерах типу PC, одною з таких є графічне програмне забезпечення Blender, яке було використано для створення даної 3D моделі. Популярними також є AutoCAD 3D, 3D Studio MAX, Autodesk Maya, Cinema 4D, Metasequoia та інші.

Наприклад у самій назві 3D Studio MAX – "тривимірна графіка" – закладено вказівку на те, що нам доведеться мати справу з трьома просторовими вимірами: шириною, висотою і глибиною. Якщо поглянути навколо: все, що нас оточує, є 3D об’єктами – побутові предмети, транспортні засоби, житлові будівлі, промислові корпуси, тіла живих істот. Однак термін "тривимірна графіка" все ж це спотворення істини. В реальності тривимірна комп'ютерна графіка має справу лише з двовимірними проекціями об'єктів уявного тривимірного світу.

Щоб проілюструвати сказане, можна уявити оператора з відеокамерою, за допомогою якої він знімає об'єкти, розташовані в кімнаті. Коли під час зйомок він переміщається по кімнаті, то в об'єктив потрапляють різні тривимірні об'єкти, але при відтворенні відзнятого відеозапису на екрані телевізора буде видно лише плоскі двовимірні зображення, що представляють собою відображені образи, знятих кілька хвилин тому тривимірних об'єктів. Сцена на екрані виглядає цілком реально завдяки наявності джерел світла, природного забарвлення всіх об'єктів і присутності

тіней, що додають зображенню глибину і що роблять його візуально правдоподібними, хоча воно і залишається всього лише двовимірним.

У комп'ютерній графіці об'єкти існують лише в пам'яті комп'ютера. Вони не мають фізичної форми – це не більше ніж сукупність математичних рівнянь і рух електронів в мікросхемах. Оскільки об'єкти, про які йде мова, не можуть існувати поза комп'ютером то єдиним способом побачити їх є додавання нових математичних рівнянь, що описують джерела світла і знімальні камери. Програмний комплекс Blender дозволяє виконувати всі перераховані вище операції.

Використання програми, подібної Blender, багато в чому схоже зі зйомкою за допомогою відеокамери кімнати, повної сконструйованих об'єктів. Програмний комплекс Blender дозволяє змоделювати кімнату і її вміст з використанням різноманітних базових об'єктів, таких як куби, сфери, циліндри і конуси, а також з використанням інструментів, необхідних для реалізації різноманітних методів створення більш складних об'єктів .

Після того як моделі всіх об'єктів створені та належним чином розміщені у складі сцени, можна вибрати з бібліотеки будь-які готові матеріали, такі як пластик, дерево, камінь та інше, і застосувати ці матеріали до об'єктів сцени. Можна створити і власні матеріали, користуючись засобами редактора матеріалів, за допомогою яких можна керувати кольором, глянцем, прозорістю і навіть застосовувати скановані фотографії або намальовані зображення, щоб поверхня об'єкта виглядала так, як це було задумано.

Застосувавши до об'єктів матеріали, необхідно створити уявні знімальні камери, через об'єктиви яких буде спостерігатися віртуальний тривимірний світ і вироблятися зйомка наповнюючих його об'єктів. За рахунок налаштування параметрів віртуальних камер можна отримати ширококутну панораму сцени або збільшити план зйомки, щоб зосередити свою увагу на окремих дрібних деталях. Пакет 3D Studio MAX підтримує моделі камер з набором параметрів властивих справжнім фото-або

відеокамер, за допомогою яких можна спостерігати сцену саме в тому вигляді, який потрібно за задумом сценарію.

Щоб зробити сцену ще більш реалістичною, можна додати до її складу джерела світла. Blender дозволяє включати в сцену джерела світла різних типів, а також налаштовувати параметри цих джерел.

1.2 Blender

Blender використовується для створення тривимірних візуалізацій, таких як статичні картинки, відео та інтерактивні відео-ігри.[9][1)]

Blender відмінно підходить як індивідуальним розробникам, так і невеликим студіям, які отримують користь від єдиного виробничого конвеєра і чуйного процесу розробки.

Blender є кроссплатформенним додатком і запускається на Linux, macOS і MS-Windows. Також він споживає відносно малу кількість пам'яті і вимагає менше місця на диску, в порівнянні з іншими програмами для 3D-моделювання. На рис.1.1 показано Інтерфейс Blender 2.79.

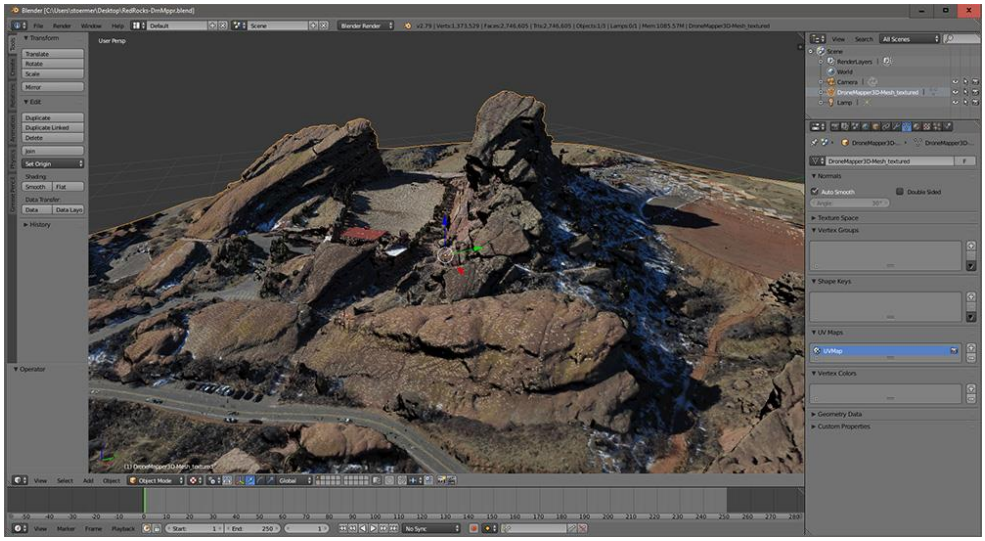

Рисунок 1.1 – Інтерфейс Blender 2.79

[1)] [9] Blender Developer wiki URL: https://wiki.blender.org/wiki/Main_Page (дата звернення 20.04.2019).

Для відтворення свого інтерфейсу Blender використовує OpenGL, що дозволяє йому виглядати однаково на всіх підтримуваних платформах. Blender містить широкий набір інструментів, що робить його придатним для виробництва майже будь-якого роду медіа-продукції. Великі студії, фрілансери та прості користувачі люди з різних країн використовують його для створення хобі-проектів, рекламних роликів, художніх фільмів, ігор та інших інтерактивних додатків, наприклад, для наукових досліджень.

У 1988 році Тон Розендаль (Ton Roosendaal) заснував голландську анімаційну студію NeoGeo. NeoGeo швидко стала найбільшою студією 3D анімації в Нідерландах і одним з лідируючих анімаційних будинків в Європі. NeoGeo створювала відзначену призами продукцію (European Corporate Video Awards 1993 і 1995 роки) для великих корпоративних клієнтів, таких як багатонаціональна компанія виробник електроніки Philips. В межах NeoGeo Тон Розендаль займався як художнім напрямом, так і питаннями розробки внутрішнього програмного забезпечення. Після ретельного вивчення питання він вирішив, що наявний в розпорядженні у NeoGeo набір 3D інструментів порядком застарів і занадто громіздкий для подальшої підтримки та модернізації, і його необхідно переписати з нуля. У 1995 році цей процес почався, а його результатом став пакет 3D моделювання, добре відомий нам як Blender. В процесі того, як NeoGeo продовжувала налагоджувати і вдосконалювати Blender, Тону стало очевидно, що програма може використовуватися в якості робочого інструменту та іншими художниками, поза компанії NeoGeo.

У 1998 році Тон вирішує заснувати нову компанію Not a Number (NaN), як дочірню компанію NeoGeo, для подальшого просування і розробки Blender'a. У серці NaN було бажання розробляти і поширювати компактний, крос-платформний пакет 3D моделювання безкоштовно. У той час це була революційна концепція, оскільки більшість комерційних пакетів коштувало кілька тисяч американських доларів. NaN сподівалася зробити доступними інструменти професійного рівня для 3D моделювання та анімації широкому

колу комп'ютерної публіки. Ділова модель NaN будувалася на забезпеченні комерційних продуктів і сервісів для Blender'a. У 1999 році NaN відвідує свою першу конференцію SIGGRAPH, з метою представити Blender ще більш широкому колу користувачів. Перша конференція Blender'a на SIGGRAPH 1999 року мала величезний успіх і пробудила величезний інтерес як у преси, так і у відвідувачів. Blender став хітом і його величезний потенціал був підтверджений!

На крилах успішного SIGGRAPH на початку 2000 року, NaN забезпечила собі фінансування від венчурного капіталу в розмірі € 4,5млн. Таке серйозне грошове вливання дозволило NaN різко розширити свої операції. І незабаром NaN налічувала понад п'ятдесят співробітників по всьому світу, які працюють над поліпшенням і просуванням Blender'a. Влітку 2000 року був випущений Blender версії 2.0. У цій версії в 3D пакет був доданий інтегрований ігровий движок. І на кінець 2000 року кількість користувачів, що зареєструвалися на веб-сайті NaN, перевалило за 250 000.

На жаль, амбіції і можливості NaN не співпали зі здібностями компанії і ринковими реаліями того часу. Таке «розширення» призвело до того, що NaN починає все спочатку з новим інвестором, заснувавши меншу компанію в квітні 2001 року. Шість місяців потому NaN випускає свій перший комерційний програмний продукт Blender Publisher. Цей продукт був націлений на зароджується ринок інтерактивної веб-орієнтованої 3D медіапродукції. Але через незадовільні продажів і не проходить важкого економічного клімату, нові інвестори вирішують зупинити всі операції NaN. Що мало на увазі, також, припинення розробки Blender. Незважаючи на те, що поточна версія Blender мала явні недоліки, складну програмну архітектуру, недоробки і нестандартно виконаний призначений для користувача інтерфейс, активна підтримка спільноти користувачів і клієнтів, які купили Blender Publisher в минулому, була така сильна, що Тон просто не міг залишити Blender і зрадити його забуттю. Оскільки створити нову компанію з досить великою командою розробників не представлялося

можливим, в березні 2002 року Тон Розендаль засновує некомерційну організацію Фонд Blender.

Основною метою Фонду Blender було знайти спосіб продовжити розробку і просування Blender'a у вигляді громадського проекту з відкритим вихідним кодом. У липні 2002 року Тону вдалося умовити інвесторів NaN погодитися на унікальний план Фонду Blender, що пропонує відкрити вихідний коду Blender'a. Щоб Фонд зміг купити права на вихідний код Blender'a і інтелектуальну власність у інвесторів NaN, а потім випустити Blender під ліцензією з відкритим вихідним кодом, організовується кампанія «Звільніть Blender» («Free Blender»), метою якої був збір необхідних для цього € 100 000. Разом з групою ентузіастів-добровольців, серед яких було кілька колишніх співробітників NaN, кампанія була запущена. На загальний подив і захоплення кампанія досягла поставленої мети в € 100 000 за все за сім тижнів. У неділю, 13 жовтня 2002 року, Blender був випущений в світ на умовах універсальної громадської ліцензії GNU. Розробка Blender'a і донині триває командою відданих добровольців з усього світу на чолі з творцем оригінального Blender'a Тоном Розендалем.

1.3 ZBrush

Програма ZBrush від компанії Pixologic – це потужний професійний інструмент для створення і редагування тривимірної графіки. Для того, щоб краще підготувати новачків до роботи в додатку, ми підготували огляд програми ZBrush. В першу чергу програма спрямована на роботу з так званої «цифрової глиною», з якої можна буквально виліплює об'єкти за допомогою різноманітних інструментів. Аналогів такому підходу практично немає в інших пакетах для 3D моделювання, хоча деякі додатки (наприклад, Maya) пропонують спеціальні інструменти для скульптінга.

Така цифрова ліпнина ідеальна для створення людей, тварин, і взагалі всього органічного. Проте, ZBrush може використовуватися для

твердотільного 3D моделювання і оснащений для цього спеціальними інструментами. Величезний набір спеціальних кистей спрямований на досягнення максимальної реалістичності при створенні 3D моделей, а інструменти накладення текстур і візуалізації доповнюють функціонал програми. На рис.1.2 показано Інтерфейс ZBrush.

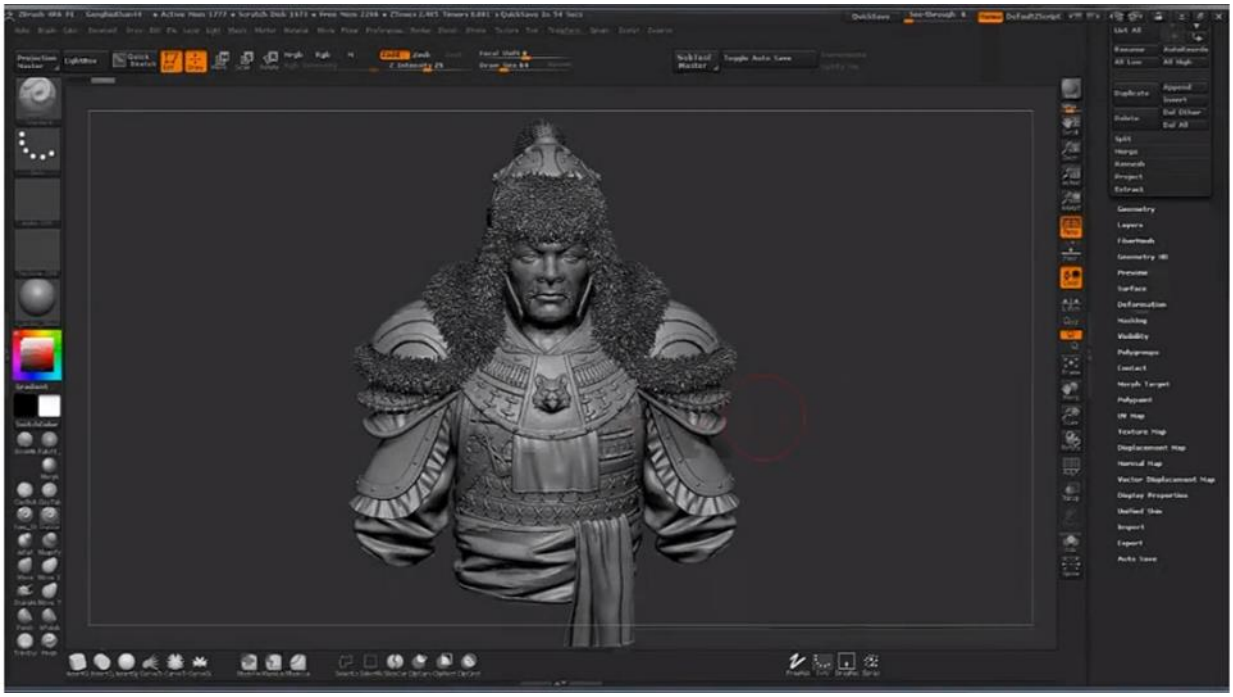

Рисунок 1.2 – Інтерфейс ZBrush

3D моделі, створені в цій програмі, затребувані в першу чергу в кіно і ігрової індустрії, де надзвичайно важливі деталізація і реалістичність. З її допомогою створювалися персонажі й атрибути багатьох знаменитих комп'ютерних ігор і фільмів, в тому числі анімаційних.

1.3 Autodesk Maya

Maya названа в честь санскритського слова माया māyā, майа, яке означає ілюзія. Maya існувала в трьох версіях:

Maya Unlimited – найповніший і найдорожчий пакет. Містить розширення Hair, Fur, Maya Muscule, Fluid Effects, Cloth і деякі інші.

Maya Complete – базова версія пакета, в якій присутній повноцінні блок моделювання та анімації, але відсутні модулі фізичної симуляції.

Maya Personal Learning Edition – безкоштовний пакет для некомерційного використання. Є функціональні обмеження, обмеження на розмір визуализированного зображення, позначка водяними знаками фінальних изображнием.

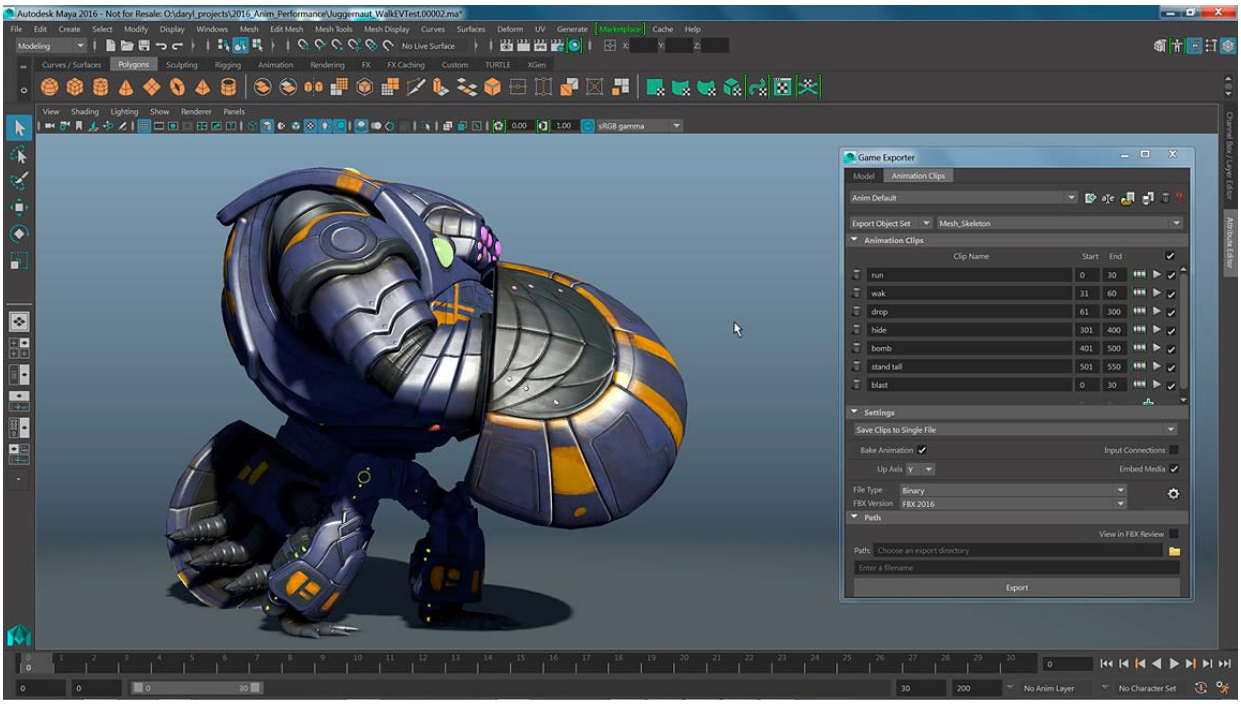

Рисунок 1.3 – Інтерфейс Autodesk Maya

Однак на виставці SIGGRAPH 2009 року компанія Autodesk представила нову версію свого 3D-редактора Autodesk Maya 2010. Починаючи з цього релізу, розробники відмовилися від поділу програми на Maya Complete і Maya Unlimited – тепер пропонується одне рішення Maya 2010. Maya 2010 містить всі можливості Maya Unlimited 2009 і Maya Complete 2009 включаючи Maya Nucleus Unified Simulation Framework, Maya nCloth, Maya nParticles, Maya Fluid Effects, Maya Hair, Maya Fur. У новій версії представлена нова система композітінга Maya Composite, заснована на програмі Autodesk Toxic, яка більше не буде доступна у вигляді окремого

додатка. Крім цього, в Maya 2010 включена система Autodesk MatchMover, менеджер для складання завдань мережевої візуалізації Autodesk Backburner, п'ять вузлів візуалізації для пакетного рендеринга засобами mental ray.

Спочатку Maya була розроблена Alias Systems Corporation і випущена для операційних систем Microsoft Windows, Linux, IRIX і Mac OS X. У вересні 2007 року, була випущена нова версія, що отримала ім'я Maya 2008. Для платформи IRIX останньою версією була 6.5, в зв'язку з зменшується популярністю ОС у останні роки. У жовтні 2005 року компанія Alias влилася в Autodesk. Представники компанії в різних інтерв'ю підтвердили, що не будуть зливати Maya і 3ds Max в один продукт.

1.4 AutoCAD

Програма AutoCAD – це справжній мастодонт серед САПР, адже перша версія була продемонстрована в далекому 1982 році на виставці COMDEX в Атлантік Сіті. У той славний час автомобілі були карбюраторними, мікросхеми – великими, а персональні комп'ютери тільки-тільки виходили на широкий ринок.

Більшість програм створювалося для величезних мейнфреймів IBM. Проте, заповзятлива група з тринадцяти програмістів зуміла побачити перспективу розвитку обчислювальної техніки і створила продукт саме для персональних комп'ютерів. Програма привернула загальну увагу вже після першої демонстрації, і розробники зрозуміли, що тепер їх головне завдання – удосконалювати своє «дітище».

Компанія Autodesk досі успішно і своєчасно цим займається. В результаті AutoCAD є найпопулярнішою програмою для роботи як з двовимірними кресленнями, так і з 3D-моделями вже протягом більш ніж тридцяти років.

Якщо говорити про основні можливості AutoCAD, то вони безумовно дуже широкі. Не будемо детально зупинятися на надаються програмою

функціях і інструментах – для цього є відповідні підручники. Тим більше, для опису всіх функцій AutoCAD знадобиться просто сила-силенна часу. Скажемо лише, що сучасний AutoCAD багато в чому перевершує по функціоналу свої ранні версії, які заслужено називали «електронними кульманами».

Крім використання графічних примітивів для отримання більш складних об'єктів в двовимірному просторі AutoCAD дає можливість створювати повноцінні тривимірні моделі з використанням твердотільного полігонального і поверхневого моделювання. Проте, в певних моментах AutoCAD в області 3D-моделювання поступається таким спеціалізованим САПР і, наприклад, SolidWorks.

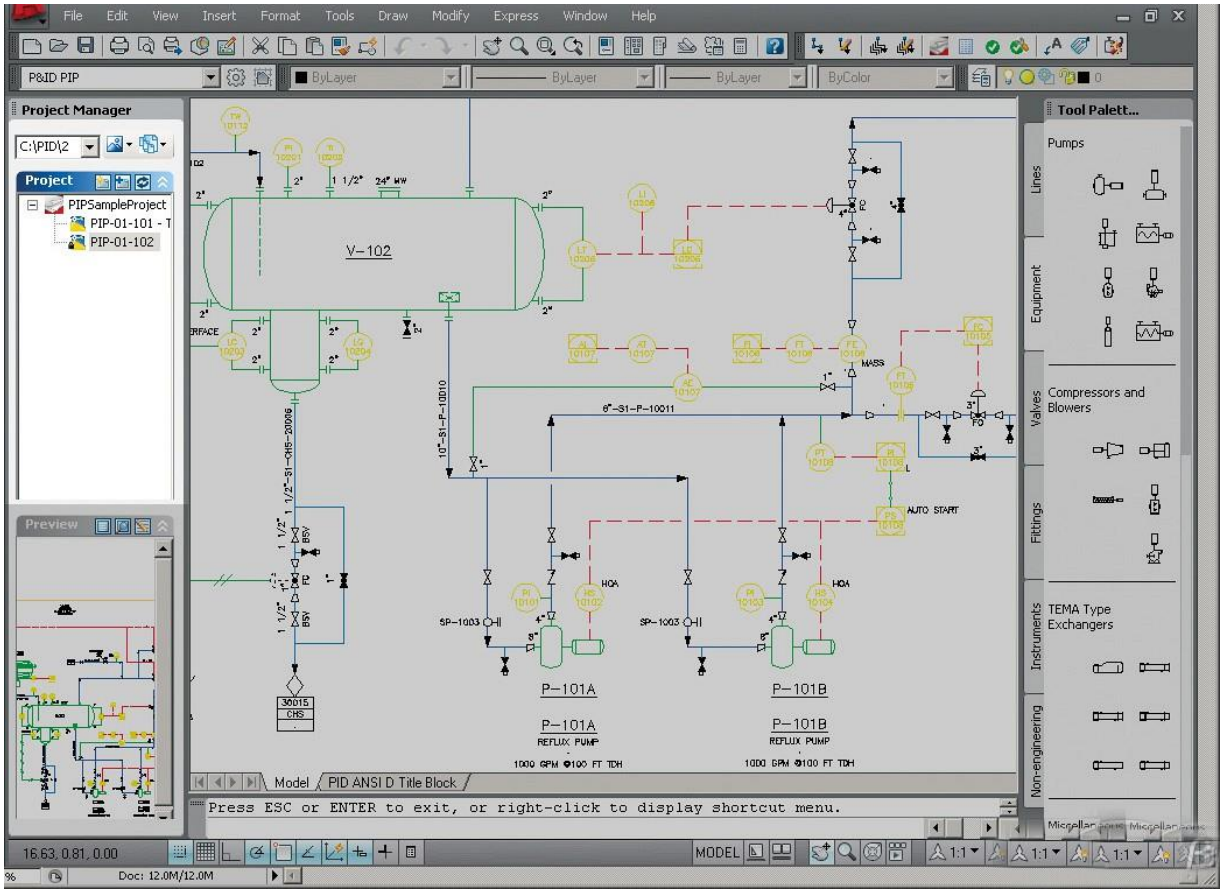

Рисунок 1.4 – Інтерфейс AutoCAD

Крім того, AutoCAD надає у Ваше розпорядження широкі можливості роботи з шарами і аннотатівнимі об'єктами. Останні версії програми припускають можливість динамічної зв'язку креслення з реальними картографічними даними та роздруківки моделей на 3D-принтері. Автокад працює з декількома форматами файлів. Основні з них – DWG і DWT.

DWG – це формат файлу, в якому зберігається безпосередньо сам креслення. Даний формат дозволяє зберігати як двовимірні, так і тривимірні об'єкти, а також підтримується іншими додатками компанії Autodesk.

DWT – файл з даними розширенням є шаблоном. Так, наприклад, Ви можете зберегти якийсь проект з усіма виставленими Вами настройками у вигляді шаблону і використовувати його в майбутньому.

AutoCAD це набір геометричних примітивів, але не всяке зображення не можна уявити як набір із примітивів. Такий спосіб уявлення діє для ділової графіки а також використовуються але не для створення мультимедійної продукції.

AutoCAD часто використовується, як база для створення прикладних програм. Так, на його основі вже випущені такі програми, як AutoCAD Mechanical, AutoCAD Architecture, AutoCAD Electrical, Promis-e, PLANT-4D, AutoPLANT, GeoniCS і ін.

Зображення в AutoCAD дають простір для редагування: зображення може без втрат масштабуватися, обертатися, деформуватися та імітувати до трьох вимірності.

2 ВИБІР ПРОГРАМНИХ ЗАСОБІВ ДЛЯ ВІЗУАЛІЗАЦІЇ ОДЕСЬКОГО ДЕРЖАВНОГО ЕКОЛОГІЧНОГО УНІВЕРСИТЕТУ

2.1 Робота у програмі Blender

Методи моделювання можуть поєднуватися один з одним. Моделювання на основі стандартних об'єктів, як правило, є основним методом моделювання і служать відправною точкою для створення об'єктів складної структури, що пов'язано з використанням примітивів в поєднанні 21 один з одним як елементарних частин складових об'єктів. Кожен з них має набір параметрів, однозначно визначають форму тривимірного тіла. Наприклад, об'єкт «Труба» визначається такими основними параметрами як внутрішній і зовнішній радіуси, висота; крім того існує ряд параметрів, що дозволяють управляти точністю побудови. Після створення об'єкта кожен з параметрів може бути змінений так, що це моментально відіб'ється на зовнішньому вигляді об'єкта у вікні редагування. Переважна більшість параметрів також можуть бути згодом піддані анімації. Стандартний об'єкт «Мавпа» входить в цей набір в силу історичних причин: він використовується для тестів матеріалів і освітлення в сцені, і, крім того, давно став своєрідним символом тривимірної графіки у Blender. Будь-яка сцена формується з використанням стандартного алгоритму, який може бути описаний таким чином: створення геометрії, налагодження джерел світла, знімальних камер і матеріалів, налаштування анімації, візуалізація. Кінцевим результатом, завершальним роботу над статичної тривимірною сценою, є «картинка» – графічний файл зображення. Динамічна сцена дає на виході набір «картинок» або анімаційну послідовність, де кожен кадр відображає зміни, що відбувалися з об'єктами сцени. Результати візуалізації можуть бути перенесені на папір, плівку, тканину або записані на відеострічку та інше. Коротко зупинимося на основних пунктах алгоритму роботи зі створення, налагодження та візуалізації тривимірної сцени. Це один з основних етапів роботи, що характеризується вимогами значних навиків і знань основних

команд і інструментів середовища Blender. Причому реально враховується саме геометрія тіл, а не їх фізичні властивості або взаємодії – ці поняття лише імітуються. Освоюючи роботу по моделюванню сцени, можна переконатися, що обсяг первинних знань доступний для запам'ятовування будь-яким початківцям користувачем, і кінцевий результат може бути досягнутий досить швидко.

Всі геометричні об'єкти програми Blender можна умовно розділити на дві категорії: параметричні та редаговані. Більшість об'єктів в Blender є параметричними. Параметричні об'єкти – це об'єкти, які визначаються сукупністю установок або параметрів, а не є описом його форми. Простіше кажучи, такі об'єкти можна контролювати за допомогою параметрів (Parameters (Параметри) на командній панелі). Зміна значень параметрів модифікує геометрію самого об'єкта. Такий підхід дозволяє гнучко деформувати розміри і форми об'єктів. Параметричними об'єктами в Blender є всі об'єкти, які можна побудувати за допомогою меню Create (Створення). Вони мають важливі настройки моделювання та анімації, тому в загальному випадку необхідно якомога довше зберігати параметричні визначення об'єкта. Однак збереження параметричних властивостей об'єктів витрачає велику кількість ресурсів комп'ютера і уповільнює роботу з об'єктами, так як всі параметри, настройки і модифікатори зберігаються в пам'яті комп'ютера. Якщо ви не припускаєте надалі використовувати параметричні властивості об'єкта, перетворіть його в Editable Mesh (Редагована поверхню). Зміна редагованих об'єктів відбувається за рахунок під об’єктів (вершини, ребра, грані, полігони) або функцій. До складу редагованих об'єктів входять: Editable Spline (редагований сплайн), Editable Mesh (редагована поверхню), Editable Poly (редагована полігональна поверхню), Editable Patch (редагована патч-поверхня) і NURBS (NURBS-поверхність). Редаговані об'єкти в стеку модифікаторів містять ключове слово Editable (редагується). Виняток становлять NURBS-об'єкти, які називаються NURBS Surfaces (NURBS-поверхні). Редаговані об'єкти виходять шляхом перетворення інших типів

об'єктів. Після перетворення параметричного об'єкта в інший тип (наприклад, в Editable Mesh (Редагована поверхня)) він втрачає всі свої параметричні властивості і не може бути змінений шляхом вказівки параметрів. У той же час редагований об'єкт набуває властивостей, недоступні параметричного, – можливість редагування на рівні під об'єктів.

Подібно величезному будинку, побудованому з маленьких цеглинок, програма Blender дозволяє створювати різнопланові сцени, використовуючи в якості будівельних блоків примітиви (параметричні об'єкти). Ви можете використовувати стандартні параметричні об'єкти для початку будь-якої роботи. Після створення до них можна застосовувати модифікатори, будувати складені об'єкти, розрізати, редагувати на рівні під об'єктів і виконувати багато інших операцій.

Процес створення і перетворення будь-яких об'єктів в цілому однаковий: об'єкт створюється за допомогою меню Create (Створення), вкладки Create (Створення) командній панелі або кнопок панелі інструментів, потім вибирається інструмент для його зміни.

Одне з основних призначень Blender – моделювання тривимірних об'єктів. Уява дизайнера тривимірної графіки дуже часто малює сцени, які неможливо створити, використовуючи тільки примітиви. Багато об'єктів, які оточують нас у повсякденному житті, мають несиметричну поверхню, відтворити яку в тривимірній графіці досить складно, але більш ніж можливо.

Об'єкти категорії Geometry (Геометрія) в 3D Max є базовим матеріалом для створення більш складних моделей. Для редагування поверхні примітивів використовуються різні інструменти моделювання.

Існують різні підходи до тривимірного моделювання:

- моделювання на основі примітивів;
- використання модифікаторів;
- моделювання сплайна;

- правка редагованих поверхонь: Editable Mesh (Редагована поверхня), Editable Poly (Редагована полігональна поверхня), Editable Patch (Редагована патч-поверхня);
- створення об'єктів за допомогою булевих операцій;
- створення тривимірних сцен з використанням частинок;
- використання N-гонів;
- NURBS-моделювання (NURBS - Non Uniform Rational B-Splines, неоднорідні нераціональні B-сплайни).

Ринок програм для 3D-моделювання дуже розвинений. На ньому бере участь багато компаній, які виробляють масу різних продуктів, пов'язаних з 3D-моделюванням. Кожен з них має щось своє – те, чого немає в її аналогах. Blender – повнофункціональна і професійна система для роботи з 3D-графікою і анімацією та скульптингом. Це означає, що вона включає в себе повний список інструментів, який не поступається менш доступним коштовним аналогам, необхідних для роботи з тривимірною графікою. Зупинимося ж докладніше на кожному з них.

У Blender використовується як полігональний метод моделювання так і метод скульптингу. Полігональний метод має на увазі використання редагованої поверхні і редагованого полігону. Такий метод вважається найбільш зручним і прогресивним. Він підходить як для створення простих моделей, так і дуже-дуже складних.

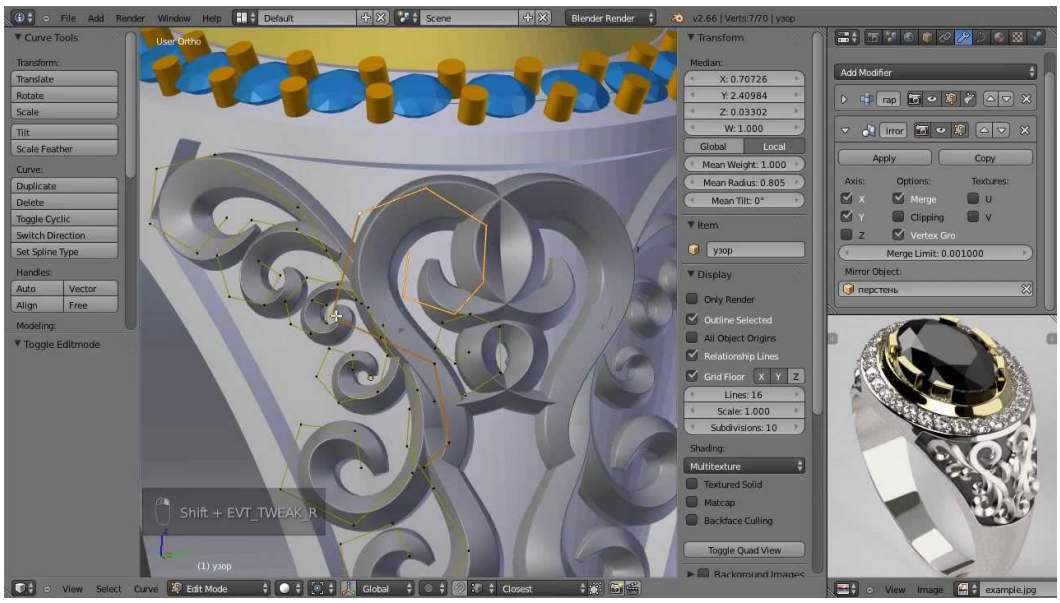

Рисунок 1.5 – Приклад складної моделі

Blender підтримує найрізноманітніші методи моделювання, так би мовити, на кожен «смак і колір». Серед цих методів, такі як моделювання за допомогою неоднорідних B-сплайнів (NURBS), моделювання за допомогою простих сплайнів і модифікатора Surface, моделювання за допомогою різних вбудованих бібліотек стандартних об'єктів, моделювання за допомогою Editable path, а також безліч інших менш відомих методів. Всі ці способи можна комбінувати між собою для досягнення кращих результатів.

Система частинок – безліч невеликих об'єктів, що мають спільні поведінку і форму, які задаються рядом параметрів. Простіше кажучи – спосіб створення таких ефектів, як сніг, дощ, дим, зірки, інше. З недавніх пір за допомогою системи частинок можна створювати не тільки ефекти, а й масиви повноцінних об'єктів, такі як, наприклад, зграя птахів, косяки риб і аналогічні їм.

Працюючи з Blender користувач має справу з уявним тривимірним простором. Тривимірний простір – це куб в кібернетичному просторі, що створюється в пам'яті комп'ютера. Кібернетичний простір відрізняється від

реального фізичного світу тим, що створюється і існує тільки в пам'яті комп'ютера завдяки дії спеціального програмного забезпечення.

Однак подібно реального простору, тривимірний простір також необмежено великий. Завдання пошуку об'єктів та орієнтації легко вирішується завдяки використанню декартової координат.

Найменшою областю простору, яка може бути зайнята якимось об'єктом, є точка (point). Положення кожної точки визначається трійкою чисел, які називаються координатами (coordinates). Прикладом координат може служити трійка (0; 0; 0), що визначає центральну точку тривимірного простору, яка також називається початком координат (origin point). Іншими прикладами координат можуть бути трійки (200; 674; 96) або (23; 67; 12).

Кожна точка тривимірного простору має три координати, з яких одна визначає висоту, інша – ширину, третя – глибину положення точки. Таким чином, через кожну точку можна провести три координатних осі кіберпростору.

Координатна вісь (axis) – це уявна лінія кіберпростору, що визначає напрямок зміни координати. У Blender є три стандартні осі, звані осями X, Y і Z. Можна умовно вважати, що вісь X представляє координату ширини, вісь Y – висоти, а вісь Z – глибини.

Якщо з'єднати дві точки в кіберпросторі, то буде створена лінія (line). Наприклад, поєднуючи точки (0; 0; 0) і (5; 5; 0) виходить лінія. Якщо продовжити цю лінію, з'єднавши її кінець з точкою (9; 3; 0) то вийти полілінія (poliline), тобто лінія, що складається з декількох сегментів. Якщо з'єднати останню крапку з першої, то вийти замкнута форма (closed shape), тобто форма, у якій є внутрішня і зовнішня області. Намальована форма являє собою простий тристоронній багатокутник (polygon), званий також гранню (face), і становить основу об'єктів, що створюються у віртуальному тривимірному просторі. У багатогранника є наступні базові елементи: вершина, ребро, грань.

Вершина (vertex) – це точка в якій з'єднується будь-яку кількість ліній. Грань (face) – це фрагмент простору, обмежений ребрами багатокутника. Ребро (edge) – це лінія, яка формує кордон грані.

В Blender об'єкти складаються з багатокутників, шматків або поверхні типу NURBS, причому найчастіше використовуються багатокутники, розташовані таким чином, щоб утворити оболонку потрібної форми. У ряді випадків для формування об'єкта потрібно всього кілька багатокутників. Однак в більшості випадків формування об'єктів вимагає використання сотень і тисяч багатокутників, що утворюють величезний масив даних. Так, наприклад, в процесі роботи з кубом комп'ютер має відстежувати положення восьми вершин, шести граней і дванадцяти видимих ребер. Для більш складних об'єктів число елементів складаються з багатокутників може досягати десятків і сотень тисяч.

Точка спостереження (viewpoint) – це позиція в тривимірному просторі, яка визначає положення спостерігача. Точки спостереження є основою формування в Blender вікон проекцій (viewports), кожне з яких демонструє результат проекції об'єктів тривимірної сцени на площину, перпендикулярну напрямку спостереження з певної точки.

Уявна площина, що проходить через точку спостереження перпендикулярно лінії погляду, називається площиною відображення, яка визначає межі області видимої спостерігачеві. Площина відображення інколи називають площиною відсічення.

Щоб побачити об'єкти, розташовані позаду площині відображення, необхідно змінити положення точки спостереження. Або "відсувати" площину відсічення, поки що цікавлять нас об'єкти не виявляться попереду площині.

У вікнах, що дозволяють заглянути в віртуальний тривимірний світ, називаються вікнами проекцій (viewports). Екран монітора сам по собі є площиною відображення, оскільки користувач може бачити тільки те, що розташовується в кіберпросторі "за площиною" екрану монітора. Бічні межі

ділянки показується у вікні проекції, визначаються межами вікна. Три з чотирьох демонструються за замовчуванням вікон проекцій в Blender є вікнами ортографічних проекцій. При побудові зображень у цих вікнах вважається, що точка спостереження віддалена від сцени на нескінченну відстань, а все промені, що йдуть від точки спостереження до об'єктів, паралельні відповідної осі координат. Четверте вікно проекції з числа прийнятих за замовчуванням, Perspective (Перспектива), є вікном не ортографической, а центральної проекції і демонструє більш реалістичне на вигляд зображення тривимірної сцени, при побудові якого промені вважаються такими, що виходять розбіжним пучком з точки спостереження, як це відбувається в реальному життя.

Тривимірні примітиви становлять основу багатьох програмних пакетів комп'ютерної графіки і забезпечують можливість створення різноманітних об'єктів простої форми. У багатьох випадках для формування потрібної моделі тривимірні примітиви доводиться об'єднувати або модифікувати. Blender надає вам два набору примітивів: стандартні (Standard Primitives) і поліпшені (Extended Primitives). До числа стандартних примітивів належать паралелепіпед, сфера, геосфера, конус, циліндр, труба, кільце, піраміда, чайник, призма. Поліпшеними називаються примітиви багатогранник, тороїдальний вузол, паралелепіпед з фаскою, цистерна, капсула, веретено, тіло L-екструзії, узагальнений багатокутник. Працюючи з примітивами майже завжди необхідно вдаватися до їх перетворення або модифікації для створення потрібних об'єктів. Наприклад, можна змоделювати стіни будівлі набором довгих і високих паралелепіпедів малої товщини. Створюючи додаткові прямокутні блоки меншого розміру і віднімаючи їх з блоків стін, можна створити отвори для вікон і дверей. Самі по собі примітиви використовуються досить рідко.

Складові об'єкти – це тіла, складені з двох або більше простих об'єктів (як правило об'єктів примітивів). Створення складених об'єктів є продуктивний метод моделювання багатьох реальних об'єктів, таких як

морська міна, стіни з прорізами для дверей і вікон, а також фантастичних тіл, перетікають з однієї форми в іншу як рідина. Blender надає можливість використовувати шість типів складових об'єктів:

-морфінг: об'єкти даного типу дозволяють виконувати анімацію плавного перетворення одного тіла в інше;

-булеві: об'єкти цього типу дозволяють об'єднувати два або кілька тривимірних тіл для отримання одного нового. Застосовуються для створення отворів або прорізів в об'ємних тілах або для з'єднання декількох об'єктів в один. Цей тип ідеально підходить для архітектурного моделювання, або будь-яких інших завдань, в яких необхідно виключити (відняти) об'єм, який займає одние тілом, при пересіканні іншим;

- розподілені: об'єкти цього типу являють собою результат розподілу дублікатів одного тривимірного тіла по поверхні іншого. Можуть використовуватися для імітації стебел трави, ямочок на поверхні м'яча для гольфу або дерев на моделі ландшафту;
- відповідні: даний тип об'єктів дозволяє змусити одне тривимірне тіло прийняти форму іншого. Це відмінно підходить для створення таких ефектів, як плавлення, танення або розтікання;
- поєднування: цей тип об'єктів дозволяє з'єднати між собою отвори в двох вихідних тілах своєрідним тунелем.
- злиття з формою: об'єкти цього типу дозволяють з'єднувати форму сплайна з поверхнею тривимірного тіла. Фактично, це дозволяє малювати на поверхнях тривимірних тіл.

Розробка системи для роботи з візуальними макетами, зокрема тривимірними моделями, потребує ретельного дослідження математичного апарату в даній області і створення функціоналу для роботи з даними та геометрією, з якої складається модель.

В основі будь-якої тривимірної моделі лежить по-перше структура геометричних сутностей, які взаємопов’язані між собою певним чином. Ця структура сутностей називається топологією моделі, і комбінація геометрі та

топології складає граничне представлення (Boundary Representation, B-Rep) тривимірного об’єкту.

В процесі створення тривимірної моделі користувач використовує різноманітні функції системи автоматизованого проектування (САПР) для додання та модифікації елементів граничного представлення. Зазвичай геометричні та топологічні елементи поділяються на декілька класів, в залежності від кількості вимірів в яких вони існують, способу задання та способу визначення топологічних категорій.

Таким чином, геометричні елементи можуть бути або точками, або лініями, або поверхнями. Лінії та поверхні здебільшого поділяються на об’єкти аналітичної та параметричної геометрії. До аналітичної геометрії відносяться ті сутності, що задаються формулою напряму: круги, еліпси, сфери, конуси, циліндри тощо. Параметрична геометрія – це або геометрія, задана в параметричному просторі іншої сутності (аналітичної або геометричної), або сплайнові лінії та поверхні, до яких відносяться сплайни Безье, B-сплайни, NURBS, та T-сплайни. Серед інших параметричних поверхонь можуть також бути поверхні обертання, поверхні округлення, фаскові поверхні та інші, що визначаються генеруючою операцією. Аналітичну геометрію можна легко привести до представлення параметричної без втрати точності, проте параметричну геометрію можна представити аналітичною тільки за допомогою подекуди малоточною апроксимації.

Топологічні елементи не мають власної форми та фігури, і вони використовуються тільки для того, щоб задати взаємозв’язок між геометричними сутностями в моделі, створюючи графоподібну структуру, завдяки якій можна відповісти на питання про модель, на які неможливо відповісти лише по геометрії, наприклад, яка частина моделі є внутрішньою, а яка зовнішньою, які грані з’єднані неперервно, які можуть від’єднуватись одна від одної, та інші питання, пов’язані зі структурою об’єкта, а не його формою.

До класів топологічних елементів завжди відносяться вершини, кромки і грані. Вершини вважаються 0-мірними елементами, кромки 1-мірними, та грані 2-мірними, оскільки саме в таких просторах існують їх параметричні функції. Інші типи елементів можуть відрізнятись в різних системах, але зазвичай вони зводяться до того, що існують також компоненти, або оболонки – зв'язані комбінації кромок або граней. Компонент може також містити вершину, але тільки одну, оскільки вершини не можуть бути неперервно пов'язаними між собою. В свою чергу, компоненти групуються в регіони, і саме регіони визначають, яка частина тривимірного простору належить об'єкту, а яка знаходиться поза ним. Наприклад, куб, який має порожнину всередині, може складатися з двох компонентів, один з яких знаходиться всередині іншої. Обидва компоненти складаються з шести пласких граней, і в такій комбінації вони утворюють три регіони – порожнеча поза кубом, твердотіла частина куба між двома компонентами, та порожнеча всередині меншого компонента. Всі регіони в свою чергу групуються в клас топології «тіло», якщо модель не композитна, або «тіло-частину», якщо модель складається з декількох тіл. В цьому випадку всі частини в результаті групуються в збірку, яка окрім частин може також містити інші збірки, і збірка найвищого рівня і є загальним класом-контейнером всього тривимірного представлення моделі.

Прив'язка геометрії до топології проходить на рівні вершин, кромок та граней. Одна точка може відповідати одній вершині, проте геометрія може параметризуватись топологією різними способами. Наприклад, кромка в вигляді кругової арки може мати цілий геометричний круг як свою базову геометричну сутність, проте використовувати лише ту його частину, яка відповідає певному параметричному інтервалу, наприклад [0, Pi/2]. Те саме стосується поверхонь, які можуть параметризуватися в двох напрямках. Якщо взяти звичайний циліндр, то один з параметричних вимірів його поверхні буде відповідати за висоту циліндру, а інший – за круг в його основі. Як окремий випадок частково використовуваних поверхонь можна

також зазначити обрізані поверхні, тобто ті, робоча частина яких обмежується набором циклічно пов’язаних ліній. Такий підхід також означає, що різні кромки або грані одного тіла можуть використовувати одні і ті самі лінії та поверхні.

Після створення граничного представлення користувач отримує повну параметричну модель, і вона може після цього використовуватись для різних цілей – проведення інженерних аналізів, виведення продукту на виробництво через тривимірний друк або CNC машини, тощо. Проте перед цим, користувачу потрібно переконатись, що модель є правильною, побачивши її повну візуалізацію. Візуалізація тривимірних моделей будується за допомогою чотирьох основних процесів: теселяції, проекції, растеризації та шейдингу.

Теселяція – це процес апроксимації моделі, що складається з різних типів ліній та поверхонь, до стану, коли вона складається лише з пласких полігонів та прямих ліній. Цей крок необхідний, тому що системи для візуалізації традиційно працюють тільки з цими апроксимованими даними, які значно швидше та простіше растеризуються, ніж будь-які інші. В результаті теселяції отримується або повністю полігональне тіло, також відоме як «меш», або набір ламаних ліній. Саме вони надалі будуть використовуватися для візуалізації.

Наступним кроком є проекція тривимірної моделі на екран користувача. На цьому етапі екран користувача приймається за неперервну площину, обмежену чотирма гранями. На основі даних про екран користувача та певних інших параметрів, створюється обрізана піраміда камери та матриця проекції. Особливості цього процесу будуть розглянуті нижче.

Коли була отримана проекція моделі на площину, що відповідає екрану користувача, проходить процес растеризації. Потреба в цьому процесі витікає з того, що тривимірні моделі існують в неперервному векторному просторі, а екран користувача в будь-якому випадку складається зі скінченного набору

дискретних елементів-точок, також відомих як пікселі. Виходячи з цього, необхідно визначити, яка саме частина якої геометрії буде видима в місці знаходження кожного пікселя, і визначити його відповідний колір.

Тут мають вплив також інші дані про тривимірний простір, не пов’язані з моделлю напряму, наприклад визначення та розташування абстрактних джерел світла, необхідність прорахування тіней, що можуть кидатись на модель іншими моделями, або самою цією моделлю. Весь цей процес називається шейдингом, і в залежності від архітектури модуля візуалізації, він може виконуватись або до, а після растеризації. До растеризації шейдинг на моделі виконується повершинно, а після неї – пофрагментно, де кожен фрагмент відповідає одному пікселю на екрані користувача.

Основним елементом абстрактних негеометричних даних тривимірної сцени є поняття камери. Процес візуалізації тривимірного простору на умовно двовимірному екрані користувача потребує вибору певної площини, яка відповідатиме оглядовому вікну екрана (viewport), і на яку буде створюватись проекція решти тривимірного простору. Ця площина може змінювати свій розмір та розміщення в залежності від налаштувань користувача, тому для її визначення використовується абстрактне поняття камери. Камера складається з точки в просторі, яка визначає її положення, трьох векторів, які визначають її орієнтацію (відомі як напрям ока, напрям верху, та напрям лівої сторони, або eye, up, left), а також кута огляду. В більшості випадків в візуалізації тривимірних моделей також визначається відстань від камери до найближчої та найдальшої точки в полі зору, щоб не відображати на екрані моделі, які знаходяться надто далеко від камери або надто близько до неї.

На основі перелічених вище даних створюється обрізана піраміда поля зору, менша основа якої відповідає площині екрану користувача, а весь простір всередині неї буде в результаті спроектований на цю площину. Таким чином, вся геометрія, що не попадає в цю обрізану піраміду, може не оброблятися для візуалізації, значно покращуючи швидкість обробки решти

даних. Процес відкидання геометрії відповідно до поля зору називається обрізанням (clipping).

Після визначення геометрії, яку необхідно відобразити на екрані, вона проектується на двовимірну площину за допомогою матриці перспективи. Матриця перспективи відповідає трансформації проекції, тому важливо, щоб вся геометрія була інваріантна відповідно до таких трансформацій. Отримана проекція далі переводиться в однорідний простір, де координати, що відповідають розмірності екрану користувача, нормалізуються для подальшого використання процесором відеокарти і відображення на фізичному дисплеї.

Існує декілька підходів до організації керування камерою в тривимірному просторі за допомогою миші та клавіатури, і серед них є два типових простих рішення – це трекболове керування та орбітне керування. Обидві ці схеми керування надають можливість переміщати та обертати камеру, а також змінювати її кут огляду для приближення або віддалення проекції. Різниця між ними полягає більшою мірою в методі обертання – трекболове керування обертає камеру відповідно до її поточного напрямку, а орбітне керування обертає камеру навколо однієї визначеної точки в просторі. Це також означає, що при використанні орбітного керування будь-яке обертання також призводить до переміщення камери на умовній сфері для того, щоб зберегти залежність між розміщенням центральної точки обертання та самої камери, що відрізняється від трекболового керування, де камера при обертанні завжди залишається на місці.

Перевага орбітного керування полягає в тому, що воно також завжди зберігає вектор напряму «вверх», і тому камеру неможливо перевернути догори низом або нахилити її вліво або вправо, тобто задати їй крен. Це лише призводить до незручності орієнтації камери, тому використання орбітного керування рекомендується в умовах, коли тривимірна сцена містить одну центральну модель, інспекція якої виконується користувачем. Якщо ж моделей багато, має сенс розглянути інші підходи до керування камерою.

Отже, на основі приведеного вище математичного апарату можна побудувати систему для роботи з тривимірною геометрією та її візуалізації на екрані користувача. В якості моделі для візуалізації використовуватимуться полігональні меші, які найбільш оптимізовані для проекції та растеризації, для шейдингу буде використане стандартизоване визначення фізичного освітлення та матеріалу, а для перегляду сцени буде використане визначення камери з орбітною схемою керування.

Незважаючи на те, що програмний пакет Blender, як правило, використовується професіоналами в світі 3D моделювання, навіть непідготовлені люди можуть використовувати програму для своїх завдань. Простий і інтуїтивно зрозумілий інтерфейс освоюється досить швидко. Всю найбільш складну роботу Blender пророблює за користувача. Наприклад, поставити зіткнення тисяч об'єктів один з одним можна все в кілька кліків мишкою. Інструменти CINEMA дають можливість створювати анімацію і зображення для абсолютно будь-якого застосування. Ілюстрації, зображення будівель, зорельотів – ви можете втілити в «життя» будь-які ваші фантазії і створити все що завгодно. Дуже важливо відзначити той факт, що Blender є досить простим і, що важливо, економічним рішенням для тих, хто тільки починає працювати з 3D. У той же час дана програма зможе вирішити, як прості, так і найскладніші завдання. Кожен користувач може підібрати ту версію програми, яка максимально підходить саме для його конкретних цілей

2.2 Проектування моделі

Робочий процес починається з створення моделі куба у Blender, що зображено на рис. 1.6 показано створення кубу у Blender 2.8(beta). Методи моделювання та текстурування відрізняються між митцями, але не враховують цільове 3D ігрове середовище, основна структура моделей як і раніше залишається незмінною.

Існують три правила, які слід пам'ятати при створенні ігрових моделей.

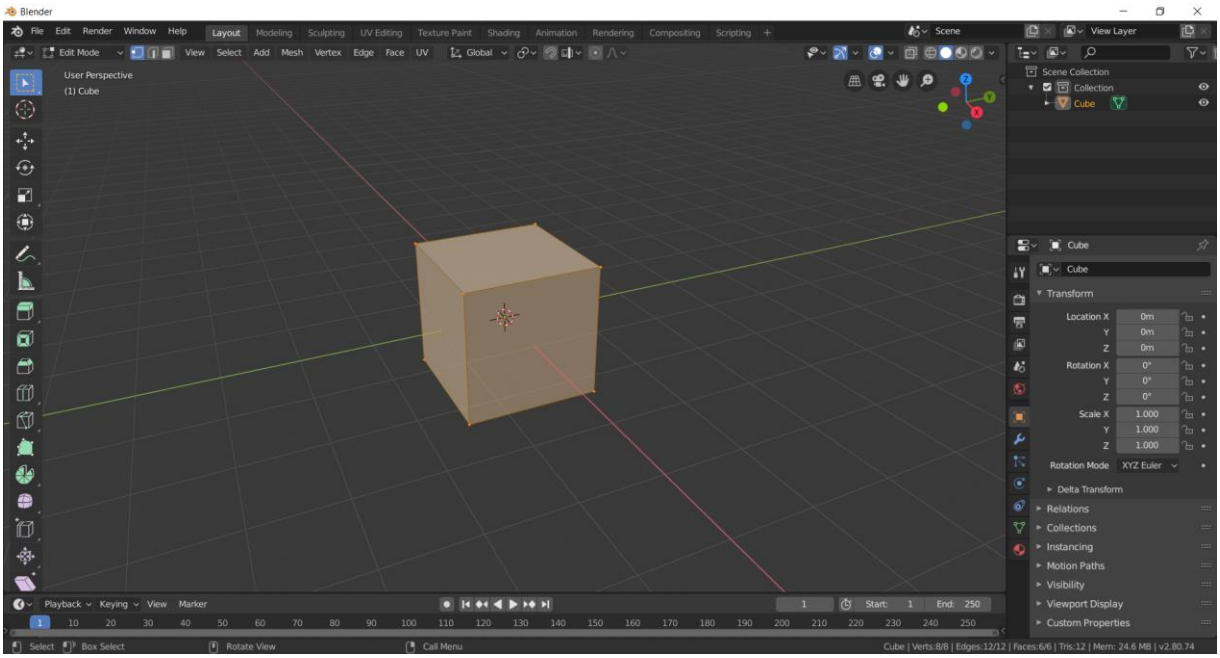

Рисунок 1.6 – Створення кубу у Blender 2.8(beta)

Перший полягає в тому, щоб мати мінімальну кількість полігонів. По-друге, використовувати текстури ефективно, і, нарешті, повторно використовувати ті ж самі матеріали кілька разів, не даючи користувачеві це зрозуміти. Всі інтерактивні віртуальні Середовища демонструють ці три фактори і тільки, слідуючи за ними, найкращі результати можуть бути досягнуто.

2.3 Логіка створення моделі

3D графіка вивчає прийоми і методи створення об’ємних моделей об’єктів які максимально відповідають дійсності. Такі об’ємні зображення можна розглядати з усіх сторін за допомогою функції обертання. Для створення об’ємних зображень використовують різні графічні фігури та гладкі поверхні. З їхньою допомогою спочатку створюються каркас об’єкта та розроблюється його деталізація. Потім його поверхню покривають матеріалами візуально схожими на реальні. Після цього роблять освітлення гравітацію властивості атмосфери та інші параметри простору в якому

знаходиться об’єкт, для об’єктів що рухаються вказують траєкторію руху та швидкість. Моделі спочатку створюються за допомогою малого числа полігонів і в залежності від елемента можуть бути перетворені в дуже високі – до декількох мільйонів полігонів, а потім спроектовані так щоб повернутися до версії з низьким рівнем полігонів. Ідея полягає в тому, щоб представити фігури в текстурах, а не в фактичній геометрії об'єкта. Це пов'язано з тим, що ігри запускаються в реальному часі і потребують збереження досить високої частоти оновлення (зазвичай більше 24 кадрів в секунду). Сучасні комп'ютери можуть працювати з мільйонами полігонів з гідними або хорошими результатами, але комп'ютери з низьким рівнем спектра повинні завжди бути також враховані для створення даних операцій.

2.3.1 Текстури та UV

Перш ніж на моделі можуть бути застосовані будь-які текстури, вони повинні мати UV-дані. Це означає, що 3D-поверхня сплющується у звичайний 2D-формат, а зображення (текстури) можуть бути над ним, створюючи візуальний вигляд. Об'єднуючи різні текстури, застосовуючи матеріальні опції і правильно налаштовуючи їх, нарешті будуть створені матеріали, які працюють в подібному ніж у кінематографічних анімаціях.

Зазвичай використовуються текстури (карти) в ігрових движках:

- Color map є єдиною текстурою, яка є обов'язковою, і яка визначає фізичну матеріал об'єкта. Зображення може містити будь-яку комбінацію різних матеріалів і в ігрових середовищах повинно бути розміщене освітлення (AO), випечене на ньому;
- Specular map визначає блиск поверхні. Ефект досягається за допомогою чорно-білого зображення - там, де чорне відображає області, які не відображають будь-яке світло і біле протилежне;
- Normal map реагує на світло і створює ілюзію складної поверхні. Таким чином модель може виглядати дуже переконливо і

реалістична, особливо в поєднанні з дзеркальним карта, що посилює ефект;

- Parallax map створює ілюзію глибшої глибини шляхом зміни текстур текстури, яка залежить від кута камери. Ця карта найчастіше використовується зі статичними об'єктами стіни, як цегляні або кам'яні стіни.
- Карта відображення є відображенням, яке показує поверхня. У поєднанні з іншою текстурою, що визначає зони впливу, результат може бути дуже переконливим і реалістичним.

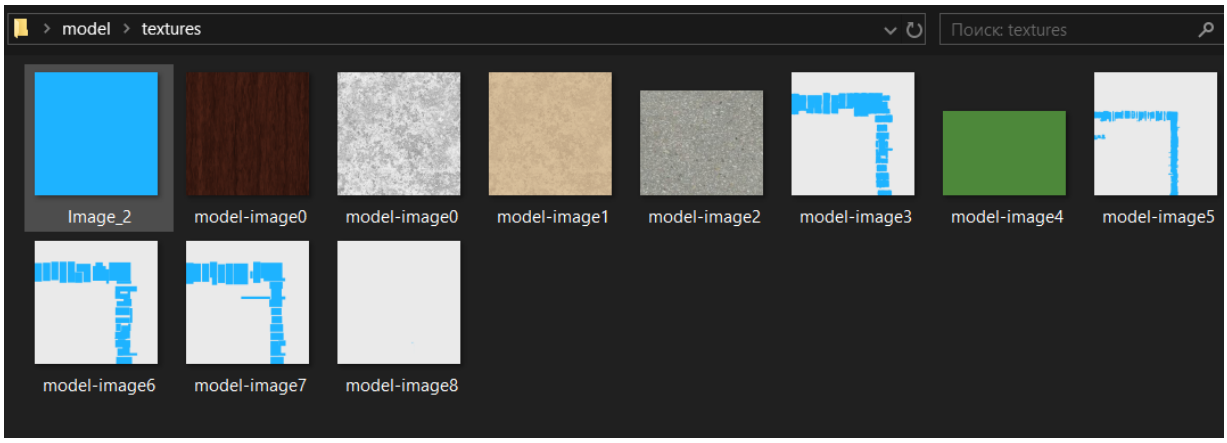

Рисунок 1.7 – Текстури що були використані для моделі

Щоб надати об'єкту фіналізований вигляд, в колірну карту наносять зовнішню оклюзію. Це означає що глобальне освітлення відображається в текстурі об'єкта. Обдурюючи тіні, об'єкт може виглядають більш автентичними і мають відчуття тяжкості, що впливає на них. Це проста операція які можуть бути зроблені в більшості програмного забезпечення 3D. У Blender функція доступна з меню візуалізації, під "bake" і функція добре працює без будь-яких настроювань, як довгий час, як увімкнено Окружність навколишнього середовища.

Звичайні карти можуть бути створені аналогічно AO, про які згадувалося раніше, але різниця полягає тільки в тому що зараз об’єкт, з якої

дані запечені. Це означає, що є висока полігональність моделі з мільйонами полігонів і ігровою моделлю лише з декількома сотнями. Процес, як правило, проходить швидко і вимагає невеликих коригувань для роботи, до тих пір, поки ці дві моделі приблизно однакового розміру і форми. Після успішного запікання модель повинна мати нормальну карту де були створені для представлення її складних форм моделі з високим рівнем політональності. На рис. 1.8 зображено редактор вузлів текстури стін.

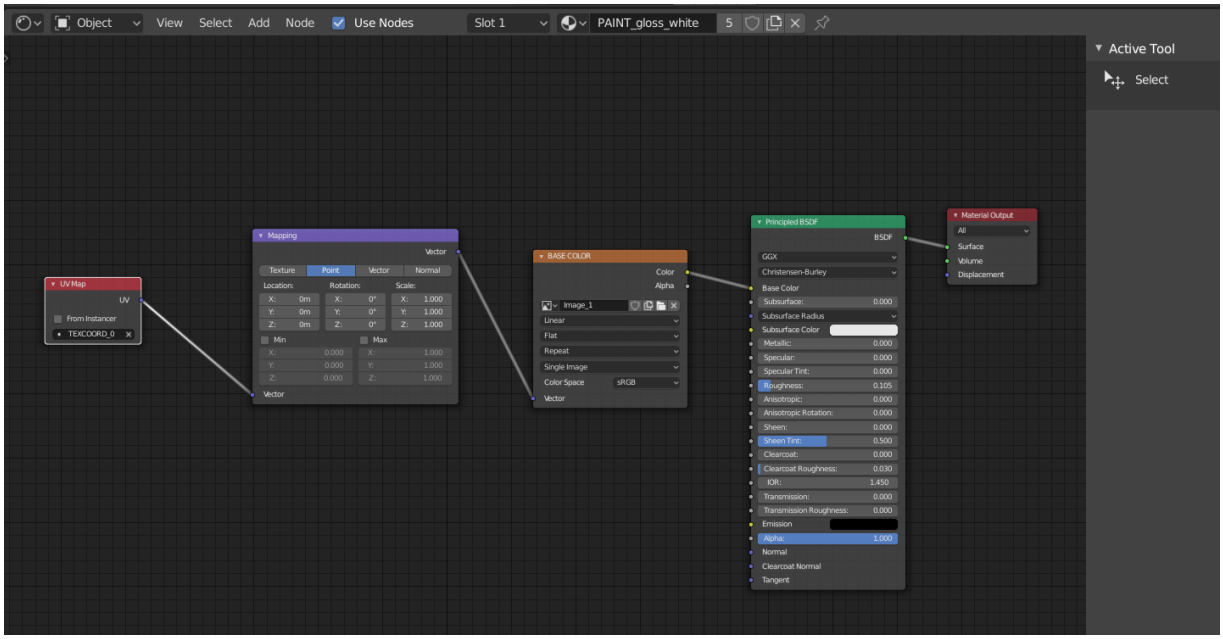

Рисунок 1.8 – Редактор вузлів текстури стін

Після того, як моделі будуть готові, їх можна імпортувати в середовище і використовувати як вони є. Процедура значно відрізняється від фотореалістичного процесу, оскільки в ній немає необхідності рендерінгу і кінцеві результати відображається в режимі реального часу. Також зміни можна легко застосувати на льоту, навіть якщо моделі вже на місці. Це також допомагає при використанні однієї моделі в декількох місцях де матеріали легко перемикаються на інші, а нові варіанти можуть бути створені швидко, не витрачаючи занадто багато пам'яті.

2.4 Остаточні коригування

Проводяться тестові рендери та створюється композиція за допомогою редактора вузлів

Остаточні коригування було зроблено за допомогою редактора вузлів Blender. Вузли – це в основному математичні функції, які додаються і з'єднуються з іншими вузлами, змінюючи, наприклад, кольори.

Інструмент, подібний редактору вузлів, зазвичай використовується для підвищення кінцевої якості виводу, і ті ж функції можна знайти у більшості програм для редагування відео, наприклад, Adobe After Effects.

Поєднуючи у цих вузлах результат може сильно відрізнятися від вихідного результату у візуалізації. Це також економить час пост-виробничої фази, оскільки ефекти зазвичай робляться тільки після завершення фази рендерінгу. Остаточні коригування здійснюються за допомогою редактора вузлів і перегляду всіх матеріалів.

3 ВІЗУАЛІЗАЦІЯ 3D МОДЕЛІ

3.1 Математична постановка задачі

В основі будь-якої тривимірної моделі лежить по-перше структура геометричних сутностей, які взаємопов’язані між собою певним чином. Ця структура сутностей називається топологією моделі, і комбінація геометрії та топології складає граничне представлення (Boundary Representation, B-Rep) тривимірного об’єкту.

В процесі створення тривимірної моделі користувач використовує різноманітні функції системи автоматизованого проектування (САПР) для додання та модифікації елементів граничного представлення. Зазвичай геометричні та топологічні елементи поділяються на декілька класів, в залежності від кількості вимірів в яких вони існують, способу задання та способу визначення топологічних категорій.

Топологічні елементи не мають власної форми та фігури, і вони використовуються тільки для того, щоб задати взаємозв’язок між геометричними сутностями в моделі, створюючи графоподібну структуру, завдяки якій можна відповісти на питання про модель, на які неможливо відповісти лише по геометрії, наприклад, яка частина моделі є внутрішньою, а яка зовнішньою, які грані з’єднані неперервно, які можуть від’єднуватись одна від одної, та інші питання, пов’язані зі структурою об’єкта, а не його формою.

До класів топологічних елементів завжди відносяться вершини, кромки і грані. Вершини вважаються 0-мірними елементами, кромки 1-мірними, та грані 2-мірними, оскільки саме в таких просторах існують їх параметричні функції. Інші типи елементів можуть відрізнятись в різних системах, але зазвичай вони зводяться до того, що існують також компоненти, або оболонки – зв’язані комбінації кромок або граней. Компонент може також містити вершину, але тільки одну, оскільки вершини не можуть бути неперервно пов’язаними між собою. В свою чергу, компоненти групуються в

регіони, і саме регіони визначають, яка частина тривимірного простору належить об’єкту, а яка знаходиться поза ним. Наприклад, куб, який має порожнину всередині, може складатися з двох компонентів, один з яких знаходиться всередині іншої. Обидва компоненти складаються з шести пласких граней, і в такій комбінації вони утворюють три регіони – порожнеча поза кубом, твердотіла частина куба між двома компонентами, та порожнеча всередині меншого компонента. Всі регіони в свою чергу групуються в клас топології «тіло», якщо модель не композитна, або «тіло-частину», якщо модель складається з декількох тіл. В цьому випадку всі частини в результаті групуються в збірку, яка окрім частин може також містити інші збірки, і збірка найвищого рівня і є загальним класом-контейнером всього тривимірного представлення моделі.

Прив’язка геометрії до топології проходить на рівні вершин, кромок та граней. Одна точка може відповідати одній вершині, проте геометрія може параметризуватись топологією різними способами. Наприклад, кромка в вигляді кругової арки може мати цілий геометричний круг як свою базову геометричну сутність, проте використовувати лише ту його частину, яка відповідає певному параметричному інтервалу, наприклад [0, Pi/2]. Те саме стосується поверхонь, які можуть параметризуватися в двох напрямках. Якщо взяти звичайний циліндр, то один з параметричних вимірів його поверхні буде відповідати за висоту циліндру, а інший – за круг в його основі. Як окремий випадок частково використовуваних поверхонь можна також зазначити обрізані поверхні, тобто ті, робоча частина яких обмежується набором циклічно пов’язаних ліній. Такий підхід також означає, що різні кромки або грані одного тіла можуть використовувати одні і ті самі лінії та поверхні.

Після створення граничного представлення користувач отримує повну параметричну модель, і вона може після цього використовуватись для різних цілей – проведення інженерних аналізів, виведення продукту на виробництво через тривимірний друк або CNC машини, тощо. Проте перед цим,

користувачу потрібно переконатись, що модель є правильною, побачивши її повну візуалізацію. Візуалізація тривимірних моделей будується за допомогою чотирьох основних процесів: теселяції, проекції, растеризації та шейдингу.

Теселяція – це процес апроксимації моделі, що складається з різних типів ліній та поверхонь, до стану, коли вона складається лише з пласких полігонів та прямих ліній. Цей крок необхідний, тому що системи для візуалізації традиційно працюють тільки з цими апроксимованими даними, які значно швидше та простіше растеризуються, ніж будь-які інші. В результаті теселяції отримується або повністю полігональне тіло, також відоме як «меш», або набір ламаних ліній. Саме вони надалі будуть використовуватися для візуалізації.

Наступним кроком є проекція тривимірної моделі на екран користувача. На цьому етапі екран користувача приймається за неперервну площину, обмежену чотирма гранями. На основі даних про екран користувача та певних інших параметрів, створюється обрізана піраміда камери та матриця проекції. Особливості цього процесу будуть розглянуті нижче.

Коли була отримана проекція моделі на площину, що відповідає екрану користувача, проходить процес растеризації. Потреба в цьому процесі витікає з того, що тривимірні моделі існують в неперервному векторному просторі, а екран користувача в будь-якому випадку складається зі скінченного набору дискретних елементів-точок, також відомих як пікселі. Виходячи з цього, необхідно визначити, яка саме частина якої геометрії буде видима в місці знаходження кожного пікселя, і визначити його відповідний колір.

Тут мають вплив також інші дані про тривимірний простір, не пов’язані з моделлю напряму, наприклад визначення та розташування абстрактних джерел світла, необхідність прорахування тіней, що можуть кидатись на модель іншими моделями, або самою цією моделлю. Весь цей процес називається шейдингом, і в залежності від архітектури модуля візуалізації,

він може виконуватись або до, а після растеризації. До растеризації шейдинг на моделі виконується повершинно, а після неї – пофрагментно, де кожен фрагмент відповідає одному пікселю на екрані користувача.

Основним елементом абстрактних негеометричних даних тривимірної сцени є поняття камери. Процес візуалізації тривимірного простору на умовно двовимірному екрані користувача потребує вибору певної площини, яка відповідатиме оглядовому вікну екрана (viewport), і на яку буде створюватись проекція решти тривимірного простору. Ця площина може змінювати свій розмір та розміщення в залежності від налаштувань користувача, тому для її визначення використовується абстрактне поняття камери. Камера складається з точки в просторі, яка визначає її положення, трьох векторів, які визначають її орієнтацію (відомі як напрям ока, напрям верху, та напрям лівої сторони, або eye, up, left), а також кута огляду. В більшості випадків в візуалізації тривимірних моделей також визначається відстань від камери до найближчої та найдальшої точки в полі зору, щоб не відображати на екрані моделі, які знаходяться надто далеко від камери або надто близько до неї.

На основі перелічених вище даних створюється обрізана піраміда поля зору, менша основа якої відповідає площині екрану користувача, а весь простір всередині неї буде в результаті спроектований на цю площину. Таким чином, вся геометрія, що не попадає в цю обрізану піраміду, може не оброблятися для візуалізації, значно покращуючи швидкість обробки решти даних. Процес відкидання геометрії відповідно до поля зору називається обрізанням (clipping).

Після визначення геометрії, яку необхідно відобразити на екрані, вона проектується на двовимірну площину за допомогою матриці перспективи. Матриця перспективи відповідає трансформації проекції, тому важливо, щоб вся геометрія була інваріантна відповідно до таких трансформацій. Отримана проекція далі переводиться в однорідний простір, де координати, що відповідають розмірності екрану користувача, нормалізуються для

подальшого використання процесором відеокарти і відображення на фізичному дисплеї.

Існує декілька підходів до організації керування камерою в тривимірному просторі за допомогою миші та клавіатури, і серед них є два типових простих рішення – це трекболове керування та орбітне керування. Обидві ці схеми керування надають можливість переміщати та обертати камеру, а також змінювати її кут огляду для приближення або віддалення проекції. Різниця між ними полягає більшою мірою в методі обертання – трекболове керування обертає камеру відповідно до її поточного напрямку, а орбітне керування обертає камеру навколо однієї визначеної точки в просторі. Це також означає, що при використанні орбітного керування будь-яке обертання також призводить до переміщення камери на умовній сфері для того, щоб зберегти залежність між розміщенням центральної точки обертання та самої камери, що відрізняється від трекболового керування, де камера при обертанні завжди залишається на місці.

Перевага орбітного керування полягає в тому, що воно також завжди зберігає вектор напряму «вверх», і тому камеру неможливо перевернути догори низом або нахилити її вліво або вправо, тобто задати їй крен. Це лише призводить до незручності орієнтації камери, тому використання орбітного керування рекомендується в умовах, коли тривимірна сцена містить одну центральну модель, інспекція якої виконується користувачем. Якщо ж моделей багато, має сенс розглянути інші підходи до керування камерою.

Отже, на основі приведеного вище математичного апарату можна побудувати систему для роботи з тривимірною геометрією та її візуалізації на екрані користувача. В якості моделі для візуалізації використовуватимуться полігональні меші, які найбільш оптимізовані для проекції та растеризації, для шейдингу буде використане стандартизоване визначення фізичного освітлення та матеріалу, а для перегляду сцени буде використане визначення камери з орбітною схемою керування.

3.2 Розробка архітектури програмної системи

Для даного дипломного проекту була обрана архітектура веб-додатку, який може функціонувати як самостійний додаток, або як частина клієнт-серверної архітектури. Розроблюваний код відповідатиме клієнтській частині системи, але для його роботи присутність серверного коду не обов'язкова. Будь-який користувач зможе запустити цей додаток у власному бразуері при наявності коду на локальному накопичувачі даних. В цьому випадку комп'ютер користувача буде виконувати роль сервера.

Візуалізація тривимірної моделі у вікні браузера забезпечується скриптовим кодом, який використовує можливості програмного інтерфейсу WebGL. WebGL – це стандарт програмування тривимірної графіки, розроблений спеціально для веб-бразуерів. Він базується на стандарті OpenGL ES 2.0, що в свою чергу базується на поширеному стандарті графічного інтерфейсу OpenGL. Різниця OpenGL ES 2.0 полягає в тому, що він розроблений спеціально для систем з обмеженими ресурсами, таких як мобільні телефони та планшети. WebGL також може використовуватися на подібних системах, так само як на звичайних комп'ютерах, тому він базується саме на цьому стандарті. Розробкою стандартів WebGL та OpenGL займається орган стандартизації Khronos Group.

WebGL надає можливості для безпосередньої роботи з графічним процесором та графічною пам'яттю пристрою, але для реалізації даного дипломного проекту його повний функціонал використовувати не доцільно. Існує велика кількість бібліотек та фреймворків, заснованих на базі WebGL, і вони надають більш високорівневий програмний інтерфейс для користувача.

Однією з таких бібліотек є Three.js. Three.js це крос-браузерна бібліотека що надає можливості для простого створення та відображення тривимірної графіки в браузері. Її різниця від WebGL полягає в тому, що в WebGL користувач повинен самостійно передавати графічні дані в буфери графічної пам'яті і програмувати інструкції по роботі з ними у вигляді

шейдерів. В Three.js цей процес автоматизований, тому користувач може лише створити геометричний об’єкт та контекст його рендерингу, і цього буде достатньо для простого відображення графіки в браузері. Three.js також надає користувачу можливість змінити стандартну поведінку рендерера, якщо він потребує більш складного функціоналу.

Метою даного дипломного проекту є не тільки візуалізація тривимірної моделі на екрані комп’ютера, а й її відтворення в віртуальній реальності. Цей процес відрізняється від звичайної візуалізації і вимагає використання спеціалізованого програмного інтерфейсу. З 2014 року в веб-браузерах з’явилася можливість використовувати технології віртуальної реальності за допомогою програмного інтерфейсу WebVR. Цей програмний інтерфейс з’явився як розробка компанії Mozilla, проте він досі не стандартизований, і на час виконання цього дипломного проекту триває розробка стандарту OpenXR від Khronos Group, що в майбутньому буде використовуватись для програмування додатків з використанням віртуальної та змішаної реальності.

З огляду на вимоги проекту, для його реалізації було вирішено використовувати фреймворк A-Frame, який поєднує в собі інтерфейси Three.js та WebVR. Це дозволяє використовувати один простий інтерфейс для і для звичайної візуалізації моделі, і для віртуальної реальності. A-Frame підтримує більшість існуючого обладнання для віртуальної реальності, і дозволяє переглядати моделі за допомогою HTC Vive, Oculus Rift, Google Daydream, GearVR, Google Cardboard, Oculus Go та інших пристроїв.

Хоча A-Frame надає простий інтерфейс для програмування додатків, він заснований на потужній архітектурі «сутність-компонент-система» (Entity-component-system, ECS). Ця система застосовує принцип інкапсуляції замість наслідування і визначає кожен об’єкт в системі як сутність. Кожна сутність може володіти одним або більшої кількість компонентів, які визначають поведінку сутності, і існують системи, які виконують визначені операції над сутностями, що володіють певними компонентами.

Прикладом архітектури ECS може бути відображення графічного об’єкту на екрані. Припустимо, що існує декілька сутностей, які володіють компонентом зовнішнього вигляду та компонентом розташування. Система відображення тоді може працювати з цими компонентами щоб визначити, як об’єкт повинен виглядати та де він має з’явитись на екрані. Це дозволяє створювати системи з чіткою та простою відповідальністю, що не буде знати про будь-які інші компоненти що можуть належати об’єкту, і з іншої сторони ізолювати дані об’єктів від систем що не повинні про них знати.

Використання архітектури ECS в даному проекті буде розкрито в наступному розділі.

3.3 Реалізація програмного коду системи

Реалізація коду переглядача тривимірних моделей починається з додавання абстрактної сцени. Сцена в тривимірній графіці виступає в ролі контейнера, що містить в собі як геометричні об’єкти, так і інші абстрактні елементи графіки, такі як камери та джерела світла.

Сцена в A-Frame виступає як одна з сутностей системи ECS, і здебільшого цей фреймворк реєструє сутності, визначені кодом мови JavaScript, як користувацькі HTML-елементи. Таким чином, використання модуля сцени на веб-сторінці може виглядати наступним чином:

```
<a-scene> <!-- Об’єкти сцени --> </a-scene>
```

З подібним синтаксисом можливо визначати параметри сцени як атрибути HTML-елемента, а її складові можливо визначати між відкриваючим та закриваючим тегами.

Така поведінка досягається за допомогою використання такого функціоналу DOM-моделі браузера, як реєстрація нових елементів, з наступним синтаксисом:

```
var constructor = document.registerElement(tag-name,
options);
```

Власне імплементація сцени виконана в вигляді надбудови над сценою бібліотеки Three.js. Сцена A-Frame містить об’єкт сцени Three.js і використовує власні визначення камер та рендерера щоб керувати відображенням цієї сцени на екрані. Композиція сцен досягається створенням нового JavaScript об’єкту, що визначає власні дані наступним чином під час створення:

```
module.exports.AScene = registerElement('a-scene', {
  prototype: Object.create(AEntity.prototype, {
    createdCallback: {
      value: function () {
        this.clock = new THREE.Clock();
        this.isIOS = isIOS;
        this.isMobile = isMobile;
        this.hasWebXR = isWebXRAvailable;
        this.isScene = true;
        this.object3D = new THREE.Scene();
        var self = this;
        this.render = bind(this.render, this);
        this.systems = {};
        this.systemNames = [];
        this.time = this.delta = 0;

        this.behaviors = {tick: [], tock: []};
        this.hasLoaded = false;
        this.isPlaying = false;
        this.originalHTML = this.innerHTML;
      }
},
```

З коду вище видно, що об’єкт сцени A-Frame визначає «object3D» як нову сцену Three.js. Під час виконання програми цей об’єкт віддається рендереру для відображення сцени на екрані наступним чином:

```
    render: {
        value: function (time, frame) {
          var renderer = this.renderer;

          this.frame = frame;
          this.delta = this.clock.getDelta() * 1000;
          this.time = this.clock.elapsedTime * 1000;

          if (this.isPlaying) {
              this.tick(this.time, this.delta);
          }

          renderer.render(this.object3D, this.camera);
        },
        writable: true
    }
```

Першим елементом готової сцени визначається камера, оскільки без наявної камери неможливо визначити, яка частина тривимірного простору має проектуватися на площину екрану.

Реалізація камери також виконується через HTML-елемент, і вона наявно демонструє використання архітектури ECS в A-Frame:

```
    <a-entity id="rig"
                movement-controls
        <a-entity camera
                  position="0 1 20"
                  rotation="-10 180 0"
                  look-controls
                  wasd-controls="fly: true"
                  gamepad-controls></a-entity>
        </a-entity>
```

В коді вище обгортка елементу камери, яка вказує на те, що користувач зможе керувати камерою. Власне сама камера визначена як сутність (a-entity), яка має декілька компонентів:

- компонент камери, що вказує на те, що ця сутність може слугувати точкою проекції тривимірного простору на екран;
- компоненти розташування та орієнтації, що визначають початкову позицію сутності в просторі;
- компонент керування орієнтацією камери за допомогою миші;
- компонент керування розташуванням камери за допомогою клавіатури;
- компонент керування орієнтацією та розташуванням камери за допомогою ігрового контролера (геймпада).

Незважаючи на те, що в одній сутності поєднуються компоненти, що працюють з різними типами маніпуляторів, вони не конфліктують одне з одним і дозволяють легко компонувати функціонал різних модулів. При такому визначенні, камеру можна пересувати за допомогою клавіш W, A, S та D, розвертати її за допомогою миші, а також виконувати ці самі дії за допомогою правого та лівого аналогових стіків геймпада.

Використання ігрового контролера для маніпуляції камерою було додано з огляду на те, що натискати на екран мобільного пристрою в режимі віртуальної реальності дуже складно, і здебільшого до них неможливо під’єднати мишу та клавіатуру одночасно, тому інтегрований пристрій маніпуляції що не вимагає взаємодії з екраном пристрою є оптимальним вирішенням цієї проблеми.

Параметр «fly» компонента «wasd-controls» вказує на те, що пересування камери по сцені буде відбуватись у напрямку, куди дивиться камера, а не лише уздовж базової площини простору.

Після визначення камери необхідно скористатись менеджером ресурсів A-Frame для того щоб визначити ресурси сцени, в тому числі тривимірні моделі.

A-Frame має власну систему менеджменту ресурсів, що дозволяє завчасно завантажити всі ресурси що будуть використовуватись в додатку.

Це дозволяє покращити швидкодію додатку і зменшити час на завантаження ресурсів під час його виконання.

Додатки, що використовують тривимірний контент, традиційно завчасно завантажують всі ресурси для того, щоб користувач не бачив недозавантажені моделі та текстури під час рендерингу.

Такими ресурсами можуть бути не тільки моделі і текстури, а й звичайні зображення, відео та аудіо, а також матеріали.

Визначення ресурсу може бути зроблено наступним чином:

```
<a-assets>
    <a-asset-item id="modelGltf" src="model.glb">
    </a-asset-item>
</a-assets>
```

Створений таким чином ресурс можна використовувати повторно в різних елементах сцени.

Ця система працює таким чином, що всі ресурси в списку ініціалізуються перед тим, як ініціалізується їх контейнер (a-assets), тому перед тим як його можна буде використовувати в браузері всі ресурси повинні бути повністю завантажені.

Якщо ресурс поданий у вигляді файлу з тривимірною моделлю, то A-Frame використовує стандартні завантажувачі файлів з Three.js для таких форматів як Collada, OBJ, STL, GLTF, та інших.

A-Frame також має можливість завантажувати не тільки локальні файли, а й файли на інших доменах. Для цього джерело ресурсу має ввімкнути заголовки CORS, інакше система безпеки браузерів не дозволить завантажити цей ресурс на сторінку додатку.

Після створення ресурсів, до сцени додаються додаткові елементи, що впливають на відображення моделі. Одним з таких елементів є елемент a-sky, що заповнює «порожнечу» тривимірного простору або певним кольором, або

зображенням текстури. Зазвичай цією текстурою виступає звичайне зображення неба.

Створити a-sky можна таким чином:

```
<a-sky color="#6EBAA7"></a-sky>
```

Іншим важливим елементом сцени є освітлення. Освітлення тривимірної сцени виконується за допомогою невидимих елементів, які слугують джерелом променів, що перетинаються з геометрією моделей, і в залежності від цього перетину прораховується ступінь освітленості поверхні. Зазвичай в тривимірній графіці використовується декілька типів джерел світла:

- AmbientLight освітлює всю сцену, не використовуючи променів. У цього джерела немає фізичного аналогу, і він використовується лише для штучного додавання освітленості.
- HemisphereLight має форму напівкулі та освітлює сцену, найближче симулюючи небо.
- DirectionalLight має один певний напрям, в який направляються промені світла, найближче симулюючи ліхтар.
- PointLight, на відміну від попередього типу, розсіює світло в усі напрямки, симулюючи звичайну лампочку.

В A-Frame джерела світла також створюються як користувацькі HTML-елементи:

```
<a-light type="ambient" color="#999999"></a-light>
<a-light type="point" intensity="0.25"
    position="4.5 15 4">
</a-light>
```

В даному випадку джерелу світла ambient задається нейтральний колір, а джерелу світла point задається позиція та інтенсивність освітлення.

Наостанок, до сцени A-Frame необхідно додати саму модель, яка базується на ресурсі, доданому раніше:

```
<a-gltf-model id="model" src="#modelGltf"
    scale="10 10 10" rotation="0 0 0" position="0 0 0">
</a-gltf-model>
```

В цьому елементі, що завантажує моделі в форматі GLTF, використовується джерело (src) з ресурсу, що був створений вище, а також конкретному екземпляру цього ресурсу задається масштаб, орієнтація та позиція в тривимірному просторі. Ресурс в цьому випадку можна вважати кресленням об’єкту, а екземпляри – його фізичними відтвореннями.

Елементи коду вище становлять основу всього необхідного для імплементації переглядача тривимірних моделей, і незважаючи на простоту визначень сутностей, такий підхід має великий потенціал в конфігурації та змішуванні різних сутностей та компонентів.

3.4 Результати тестування

Тестування веб-додатку вимагає перевірки роботи всього функціоналу на декількох браузерах, а також наявність задовільної швидкодії. Для цього були розроблені наступні тестові випадки:

- тестування всього функціоналу повинно проводитись на браузерах Edge, Firefox та Chrome;
- перевірка правильності відображення тривимірної моделі;
- перевірка на відсутність графічних артефактів (неочікуваних результатів візуалізації) на сцені;
- перевірка роботи системи керування камерою за допомогою клавіатури, миші та геймпаду;
- перевірка правильності освітлення моделі;

- перевірка правильності роботи візуалізації в віртуальній реальності на декількох пристроях, в тому числі мобільних.
- перевірка швидкодії. Рендеринг повинен відмальовувати принаймні 30 кадрів на секунду.

Рисунок 1.9 – Результати тестування

В результаті тестування було виявлено, що перелічений функціонал працює на всіх браузерах, в тому числі мобільних, і ніяких збоїв виявлено не було.

Також було виявлено, що правильність відображення моделі залежить від її масштабу. Занадто малі моделі з великою кількістю елементів можуть втрачати точність візуалізації, або втрачати дані топологічної та геометричної зв’язаності полігонів. Це пов’язано з проблемою точності обчислень з числами з плаваючою точкою, і це відома проблема багатьох систем з тривимірною графікою. Цю проблему можна уникнути експортувавши модель з достатнім масштабом координатної системи.

Правильність роботи камери на мобільних пристроях залежить від використовуваної версії A-Frame. На версіях старіших від 0.8.0 камера може бути занадто сенситивною і рухатись занадто швидко для комфортного використання системи.

3.5 Впровадження системи

Розроблений додаток може використовуватися як самостійний додаток на комп'ютері користувача, якщо на ньому встановлений веб-браузер. Для цього необхідно скопіювати файли проекту на локальний накопичувач даних і відкрити за допомогою браузера файл index.html. Відображувана модель при цьому може міститися або на цьому самому накопичувачеві, або на певному сервері в інтернеті, в залежності від конфігурації додатку.

Альтернативним підходом до впровадження цього додатку може бути його розгортання на будь-якому сервері. Оскільки всі файли додатку є файлами HTML та JavaScript, вони формують самостійну клієнтську систему, тому можуть бути додані до будь-якого серверу, який може надавати клієнту статичні файли.

Прикладами таких серверів можуть бути Apache, Express, Glassfish, Bobcat та інші. Імплементація серверного коду в даному випадку не грає ролі, і сама відображувана модель може зберігатися як на цьому самому сервері, так і деінде, якщо прописана правильна адреса для ресурсу моделі.

Нижче наведений приклад реалізації серверного коду за допомогою сервера Express.js с даним додатком:

```
const express = require('express')
const app = express()
const port = 5000
var path = require('path');

app.use(express.static(__dirname + '/public'));

app.get('/', (req, res) => {
    res.sendFile(path.join(__dirname + '/index.html'));
});

app.listen(port, () => console.log(`${port}`))
```

Цей сервер визначає папку зі статичними ресурсами (public) та при запиті до кореневої сторінки (наприклад, https://exampleserver.com/) надсилає клієнту файл index.html, який самостійно завантажить інші елементи додатку, включаючи JavaScript-код та модель.

Простішою альтернативою для публікації додатку може бути публікація на сервісі glitch.com, який дозволяє розробникам публікувати невеликі веб-проекти з обмеженою кількістю ресурсів, а також копіювати існуючи проекти створюючи власні аналоги існуючого коду. Основна ідея цього сервісу – це поширення експериментів та концепцій, які інші люди можуть надалі використовувати та поліпшувати у власних проектах. Все, що необхідно зробити, щоб опублікувати проект на glitch.com – це скопіювати файли в проект та запустити файл index.html.

Таким чином, розроблений додаток може легко використовуватися як локально, так і в вигляді частини певного веб-сайту, опублікованого на віддаленому сервері.

ВИСНОВКИ

У результаті роботи, був виконаний аналіз предметної галузі 3D моделювання. Був проведений аналіз ринку програмного забезпечення, детально проаналізовані основні графічні программи для створення 3D зображень, указані їх недоліки і переваги.

В результаті аналізу сучасного стану на ринку 3D моделювання для виконання бакалавскої роботи було обрану графічне програмне забезпечення Blender, що є більш раціональним для вирішення поставленної задачі. Графічне програмне забезпечення Blender займає провідне місце серед аналогічних графічних редакторів, тому дало можливість створити якістний, сучастний 3D продукт.

Було проведено моделювання, обрана программа та середовище розробки веб-додатку. З розглянутих середовищ розробки, була обрана архітектура веб-додатку який забезпечується скриптовим кодом, який використовує можливості програмного інтерфейсу WebGL.

Розроблено інтерактивну візуалізацію для Одеського Державного Екологічного Університету, що надає можливість заочно розглянути його архітектуру у 3D просторі.

Одержаний веб-додаток може бути розміщений на сайті ОДЕКУ.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. GPL license. URL: http://download.blender.org/release/GPL-license.txt (дата звернення 30.03.2019).
2. About Renderfarm. service. Renderfarm.site documentation. URL: http://www.renderfarm.fi/page/about-renderfarmfi (дата звернення 1.04.2019).
3. Blender developer interview URL: http://cgenie.com/articles/130.html
4. Community survey made by CGenie. URL: http://cgenie.com/component/content/article/47-articles/104-cg-community-surveyupgrades-09.html (дата звернення 02.04.2019).
5. History of Blender. Blender.org site documentation. URL: http://www.blender.org/blenderorg/blender-foundation/history/ (дата звернення 15.03.2019).
6. Yang, J. 2007. Construction Innovation: Smart and Sustainable Built Environment. Emerald Group Publishing Limited. (дата звернення 05.05.2019).
7. Джеймс Кроністер. «BlenderBasics4-е издание». URL: http://www.cdschools.org/site/Default.aspx?PageID=455 (дата звернення 10.04.2019).
8. Андрей Прахов «Самоучитель Blender2.6». URL: http://www.ozon.ru/context/detail/id/19025028/?gclid=cjgkeajwtzucbrd77aiiq_v4xnasjabkag8j12sut5ntr-k-ig-gvq1mkhel68jfwjw_x_r3tvafd_bwe (дата звернення 11.04.2019).
9. Blender Developer wiki URL: https://wiki.blender.org/wiki/Main_Page (дата звернення 20.04.2019).