НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ

«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
імені ІГОРЯ СІКОРСЬКОГО»

Факультет електроніки
(повна назва інституту/факультету)

Кафедра акустичних та мультимедійних електронних систем
(повна назва кафедри)

«До захисту допущено»
Завідувач кафедри
С.А. Найда
(ініціали, прізвище)
"01" червня 2021 р.

Дипломна робота

на здобуття ступеня бакалавра

зі спеціальності 171 Електроніка
(код і назва)

на тему: «Особливості застосування 3D-графіки для створення мультимедійного контенту»

Виконав: студент IV курсу, групи ДВ-72
(шифр групи)

Захарченко Ілля Павлович
(прізвище, ім'я, по батькові) (підпис)

Керівник

асистент, к.т.н. Філіпова Наталія Юріївна
(посада, науковий ступінь, вчене звання, прізвище та ініціали) (підпис)

Консультант
(назва розділу) (посада, вчене звання,науковий ступінь,прізвище, ініціали) (підпис)

Рецензент доцент каф. ЕПС, к.т.н., доц. Клен К. С
(посада,науковий ступінь,вчене звання,науковий ступінь,прізвище,ініціали) (підпис)

Засвідчую, що у цьому дипломному проекті немає запозичень з праць інших авторів без відповідних посилань.

Студент
(підпис)

Київ – 2021 року

Національний технічний університет України

«Київський політехнічний інститут імені Ігоря Сікорського»

Факультет Електроніки

Кафедра акустичних та мультимедійних електронних систем

Рівень вищої освіти – перший (бакалаврський)

Спеціальність 171 Електроніка

ЗАТВЕРДЖУЮ
Завідувач кафедри

С.А. Найда
(ініціали, прізвище)

« 24 » травня 2021 р.

ЗАВДАННЯ

на дипломну роботу студенту

Захарченко Ілля Павлович
(прізвище, ім'я, по батькові)

1 Тема роботи: «Особливості застосування 3D-графіки для створення мультимедійного контенту»

керівник роботи Філіпова Наталія Юріївна, к.т.н.
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)

затверджені наказом по університету від «24»травня 2021 р. №1316-с

2 Термін подання студентом роботи 04 червня 2021 р.

3 Вихідні дані до роботи:

4 Зміст роботи: 1) Розглянуто теоретичні засади технологій 3D графіки; 2) Проаналізовано програмні інструменти для реалізації моделі; 3) Розглянуто теоретичні засади анімації, а саме інверсійної кінематики; 4) Створено анімовану модель.

5 Перелік ілюстративного матеріалу: комплект презентації за матеріалами проведеного дослідження.

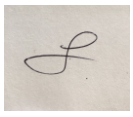
6. Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв

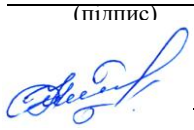
7. Дата видачі завдання 11 березня 2021 р.

Календарний план

№ з/п	Назва етапів виконання дипломної роботи	Термін виконання етапів роботи	Примітка
1	Написання першого розділу	24.03.2021	виконано
2	Написання другого розділу	20.04.2021	виконано
3	Написання третього розділу	10.05.2021	виконано
4	Підготовка матеріалів до друку та оформлення пояснювальної записки	30.05.2021	виконано
5	Підготовка та оформлення презентації для доповіді	1.06.2021	виконано

Студент  І.П.Захарченко

(підпис) (ініціали, прізвище)

Керівник роботи  Н.Ю. Філіпова

(підпис) (ініціали, прізвище)

УДК 004.92

РЕФЕРАТ

Дипломна робота: 72 с., 53 рис, 20 джерел,1додаток.
КОМП'ЮТЕРНА ГРАФІКА, ТРАСУВАННЯ ПРОМІНІВ, ВІЗУАЛІЗАЦІЯ, РОБОТНА КІНЕМАТИКА, АНІМАЦІЯ.

Об’єкт дослідження – програмні інструменти та технології для створення анімованої моделі.

Метою роботи є аналіз особливостей застосування 3D-графіки для створення мультимедійного контенту. Крім цього, необхідно визначити основні програмні підходи, які дозволяють реалізувати модель.

У результаті виконання дипломної роботи була створена модель робота, анімовано її за допомогою інверсійної кінематики

Галузь застосування: кіновиробництво, машинобудівництво, програми зі створення візуальних ефектів для кінофільмів.

THE ABSTRACT

Thesis of bachelor’s: 72 p., 53 fig., 20 refer, 1 ad.

COMPUTER GRAPHICS, RAY TRACING, VISUALIZATION, ROBOT KINEMATICS, ANIMATION.

The object of research is software tools and technologies for creating an animated model.

The aim of the work is to analyze the features of the use of 3D graphics to create multimedia content. In addition, it is necessary to identify the main software approaches that allow you to implement the model.

As a result of the thesis, a robot model was created, animated by its inversion kinematics

Field of application: film production, mechanical engineering, programs for creating visual effects for movies.

ЗМІСТ

ВСТУП

В наш час важливість виготовлення комп'ютерної графіки складно переоцінити. Світ не стоїть на місці, швидкі темпи розвитку технічного прогресу зробили комп'ютерну графіку затребуваною у багатьох напрямках промислово-побутової сфери. 3D графіка є невід'ємним супутником архітекторів, дизайнерів, діячів культури, рекламних фахівців, фахівців в області ігрової індустрії і важкого машинобудування. Можливості 3D графіки в сучасному світі практично безмежні.

Досить частим явищем є створення рекламної кампанії продукту ще до його появи на світ. В такому випадку єдиним виходом є застосування комп'ютерної графіки. при правильному підході до процесу створення 3D моделі і анімації, людині дуже складно визначити, що перед ним зображення не реального товару, а його комп'ютерної моделі. Технології розвиваються дуже швидко, тому більше немає потреби використовувати складні декорації або реквізит, який неможливо дістати – комп'ютер запросто вирішить цю проблему.

Базовим поняттям в 3D графіці є 3D модель. Якісно створенна модель є основою будь-якого проекту, адже якщо вона зроблена неякісно, то ні текстури, ні анімація, ні візуалізація – не виправлять ситуацію суттєво.

Ще одним поняттям, розглянутим в даній роботі, є ріггінг. Він дозволяє спростити маніпуляцію великою кількістю об'єктів, що значно полегшує роботу аніматору.

Проаналізувавши ситуацію на ринку комп'ютерної графіки, можна зробити висновок, що опис процесу створення 3d моделі з ріггінгом є дуже актуальною темою для дипломної роботи. Результатом практичної частини роботи є 3d модель з ріггінгом для анімації.

1. ТРИВИМІРНА ГРАФІКА

1.1 Введення в 3D графіку

1.1.1 Етапи створення 3D моделі

Традиційне малювання здійснюється тільки в площинах, при цьому видно тільки одна сторона зображеного предмета. У 3D зображенні використовується вісь глибини, вісь Z. За допомогою цієї додаткової площини виходить уявити всі сторони предметів.

Перша перевага даного методу полягає в тому, що при наявності моделі 3D художник розміщує її в кадрі і анімує, а фінального зображення отримується за допомогою спеціальної програми - візуалізатор (render).

Друга перевага в тому, що одна тривимірна модель є багаторазова і її можно використати в інших проектах, так як нею можна легко маніпулювати і міняти зовнішній вигляд.

Третя перевага полягає в можливості створення сильно деталізованих моделей. На загальному плані це не завжди можливо побачити, але якщо наблизити камеру і візуализатор розрахує, що видно в кадрі, а що - ні.

Найпоширеніший спосіб моделювання – це полігональне моделювання. Цей метод полягає в тому, що поверхня моделі створюється двовимірними геометричними примітивами - для комп'ютерних ігор трикутники, для інших цілей - чотирикутники і інші багатокутники. Ці багатокутники, з яких складається модель, називають полігонами. В найчастіше, при створенні 3D моделі, використовуються чотирикутники, із-за того що при переносі моделі в ігровій движок дуже легко отримати трикутники з чотирикутників, а також модель з чотирикутників при необхідності згладжування виходить без артефактів.

З кількістю полігонів в моделі зростає якість диталізації. Але, у велика кількість полігонів має свої мінуси наприклад зниження швидкодії. Для великої кількості необхідна велика кількість точок, на яких вони побудовані, це призводить до суттєвого збільшення кількості даних, яку обробляє

процесор. Тому задля створення якісної моделі необхідно тримати баланс між продуктивністю і деталізацією моделі. Тому вкористовуються такі терміни, як high poly (високополігона модель) і low poly (низькополігона модель). Для комп'ютерних ігор застосовуються нізкополігональних моделі, це пов’язано з візуалізацією яку вони виконують в реальному часі.

Після моделювання отримується лише математична модель, з інформацією про геометричній формі об'єкта. Для надання моделі потрібного кольору, використовуються текстури.

Текстура – це двовимірне зображення, яке накладається на тривимірну модель. Використовуються процедурні текстури (згенеровані алгоритмом) і створенні в графічному редакторі. Текстурою задається тільки колір і малюнок моделі, а здатність відзеркалювати, переломлювати світло, мати рельєф і прозорість задаються в графічному редакторі у властивостях матеріалу. Тобто, матеріал – це математична модель, якою описуються параметри надані поверхні.

Перед тим, як накладанням текстури на модель, потрібно створити її UV розгортку - тобто уявити як поверхня моделі буде проектуватися на площину. Літерами U і V позначають осі координат площини розгортки, оскільки літери X, Y і Z використовуються для позначення просторових координат.

Створення тривимірної моделі можно розділити на такаі стадії:

- Пошук референсів або креслень з яких буде створена модель.
- Моделювання об'єкта на основі рефференса або креслення.
- UV розгортка.
- Створення текстур.
- Налаштування матеріалу його фізичних властивостей

Після проходження цих етапів модель готова для візуалізації або можна працювати з нею роботу:

- Налаштування рігу моделі.
- Анімація.

У 3D графіці після виготовлення моделі для її візуалізації потрібно її помістити в сцену додати камеру і освітлення, після цього можна отримати фінальний рендер. Для прорахування рендеру використовується фізична модель поширення променя світла з урахуванням відображення, розсіювання, відблиски, заломлення тощо. При традиційному малюванні художник сам створює світлові ефекти, а в 3D графіці візуалізатор сам відмальовує підсумкове зображення, а автор створює сцену з урахувуючи геометрію, матеріали освітлення, властивості камери.

1.1.2 Аналіз сучасних методів 3D моделювання

Системи тривимірного моделювання працюють в трьох площинах координат таким чином, якщо змінити щось одному виді це зміниться на інших видах. Деякі з 3D систем автоматично аналізують фізичні характеристики, таких як моменти інерції, вага, і т.д.

Види тривимірного моделювання:

- Каркасне моделювання;
- Твердотільне моделювання;
- Поверхневе моделювання;

Особливістю каркасної моделі є те, що при її описі використовуються геометричні об'єкти першого порядку – лінії і ребра. Каркасні моделі застосовуються для завдання об'єктів, що становлять поліедри тобто замкнуті багатогранники довільної форми, обмежені плоскими гранями, або об'єкти, одержувані переміщенням утворює, яка фіксується в деяких положеннях. 3D - модель в цьому випадку містить список координат вершин багатогранника із зазначенням зв'язків між ними тобто ребер. [19]

Недоліки каркасної моделі:

- Необхідність подати всі ребера для того, щоб зобразити модель;
- Нездатність розпізновати криволінійні грані;
- Помилки при обчисленнях фізичних характеристик оделі;
- Нездатність створення тонових зображень через відсутність інформації про

грані.

- Відсутність інформації те як об'єкти розтошовані один відносно іншого;

Однак не можна сказати, що цей спосіб моделювання зовсім безпідставний. Каркасна модель витрачає менше ресурсів комп'ютера і тому краще підходить для виконання простих завдань. Цей метод часто використовують не скільки при моделюванні, а скільки як один з методів візуалізації при відображенні готових моделей.

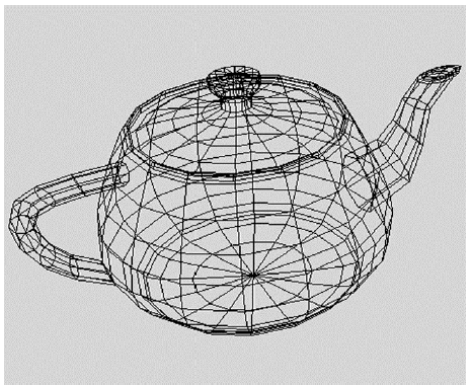


Рисунок 1.1 – Каркасна модель чайника.[19]

Другий метод – поверхневе моделювання, цей метод визначається не тільки точками і лініями, а й поверхнями. В цьому методі об'єкти обмежеються поверхнями, за допомогою яких їх відокремлюють від навколишнього середовища. Поверхня об'єкта обмежується контурами, які є результатом пересічення двох поверхонь. Вершину об'єкту можно задати через перетин трьох поверхонь.[19]

Переваги поверхневого моделювання над каркасним:

- Здатність визначати складні криволінійні грані;
- Можливість отримувати тонові зображення;
- Здатність розпізнавати особливі побудови на поверхнях, наприклад, отвори;
- Можливість отримання якісного зображення.

Поверхневе моделювання базується на двох математичних положеннях:

Будь-яка поверхня може бути апроксимована многогранником, грані якого складаються з найпростіших плоских багатокутників;

В моделі можливо користуватися поверхнями другого порядку і поверхнями, форму яких визначено різними способами апроксимації та

інтерполяції.

Всі об'єкти, створені за допомогою поверхневого моделювання, мають внутрішні і зовнішні сторони.

Типи поверхонь:

- Базові поверхні – це поверхні, які отримуються якщо одна плоска крива рухається відносно іншої кривої;
- Поверхні обертання – це поверхні, які отримуються якщо плоска грань обертається навколо деякої осі;
- Поверхні перетинів і сполучень - поверхні, які отримуються при сполучені або перетині;
- Скульптурні поверхні – поверхні, які не описиються математичними рівняннями і які отримуються лише за допомогою точок у просторі що поєднуються сплайнами;
- Аналітичні поверхні – поверхні, які можно описати задопомогою математичного рівняння.

Рисунок 1.2 – Поверхневе моделювання.[19]

Третій метод – твердотільний. Твердотільна модель визначена в тривимірному обсягу, який займає тіло яке вона визначає. Твердотільне моделювання можно назвати – найдосконалішим і достовірнішім методом для відтворення реального об'єкта. [19]

Переваги цього методу:

- Внутрішня і зовнішня області об'єкта розмежованні;
- Приховані лінії автоматично видаляються;

- Тривимірні розрізи об'єкта будуються автоматично, це важливо для аналізування складних моделей;
- Метод використовує аналіз який автоматично отримує зображення з конкретними характеристиками;
- Можливість отриманти тонові зображення;

Способи створення твердотільних моделей бувають двох класів:

- Метод конструктивного уявлення;
- Метод граничного уявлення.

При використанні методу конструктивного уявлення твердотільна модель створюється з базових елементів (примітивів, які визначено розмірами, формою, орієнтацією і точкою прив'язки).

Для створення твердотільних моделей використовуються булеві операції:

- Об'єднання;
- Різниця;
- Перетин.

У даного методу є свої плюси і мінуси. Метод конструктивного уявлення дозволяє розраховувати фізичні властивості об'єкта з високою точністю. Недолік цього методу в тому що він потребує відповідний досвід роботи який відмінний від звичних способів моделювання.

Другий метод – це метод граничного уявлення. Цей метод описує межі тіла і задає межі, якими описується об'єкт. Це лише один метод, який дозволяє створювати точне уявлення про геометричне тверде тіло. При використанні цього методу необхідно задавати контури або межі об'єкта і різні види об'єкта, а також вказувати лінії що повязують види для встановлення взаємної відповідності.[19]

Кожен з методів має свої переваги і недоліки. Метод конструктивного представлення зручніший при первісній побудові моделі; також, цей метод буде займати менше місця в базі даних. Нажаль, метод конструктивного представлення, не надто добрий для побудови складних форм. Також, хоча в конструктивному методі дані займають набагато менше місця, кількість

обчислень для відтворення моделі виявляється більше.

Але, граничний метод зберігає точний опис обсягу моделі, що займає багато пам'яті, але вимагає малу кількість обчислень для відтворення моделі. Велика перевага систем, які базуються на граничному уявленні - це простота з якою можно перетворити модель в каркасну і назад.

Рисунок 1.3 – Твердотільне моделювання.[19]

Поєднання граничного і конструктивного уявлень утворюють собою гібридну систему. За допомогою гібридного моделювання поверхнева, каркасна і твердотільні геометрії поєднуються, а також зявляється можливість використовувати параметричне моделювання.

1.2 Технології відображеня 3D графіки

1.2.1 Розвиток 3D графіки

Головними факторами прогресу графічних технологій є ігри і кінематограф. Коли, технології не дозволяли робити тривимірні ігри, були вигадані способи імітації 3D. Перші спроби візуалізувати тривимірний простір можна побачити в іграх subLOGIC Flight Simulator 1 (1979) і Battlezone (1980).

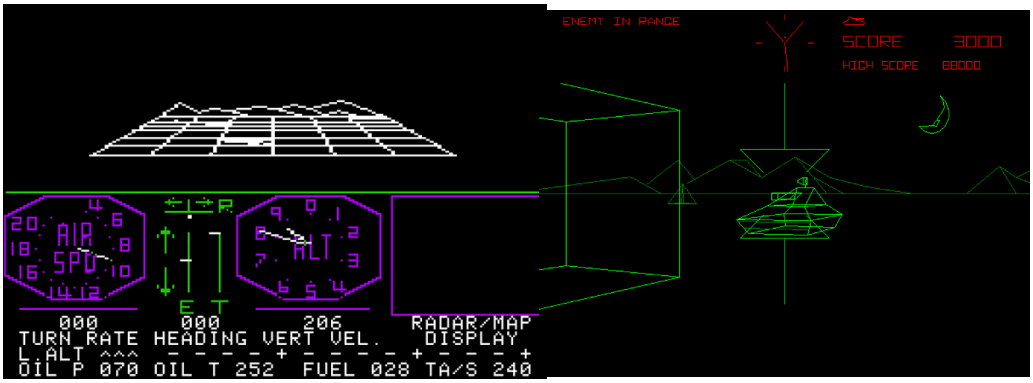


Рисунок 1.4 – (а) subLOGIC Flight Simulator 1 (1979) , (б)Battlezone (1980).[20]

Ці ігри використовували каркасний рендеринг (wireframe rendering). Об'єкти в просторі представляються у вигляді точок. Деякі пари точок з'єднані лініями. Для відображення такої сцени потрібно спроектувати точки, які знаходяться в зоні видимості, на екран, а після просто домалювати лінії між ними.

Перша більш реалістична імітація 3D простору була реалізована в грі Wolfenstein 3D від idSoftware в 1992 році. Вони вперше реалізували в грі метод візуалізації, який називається ray casting.

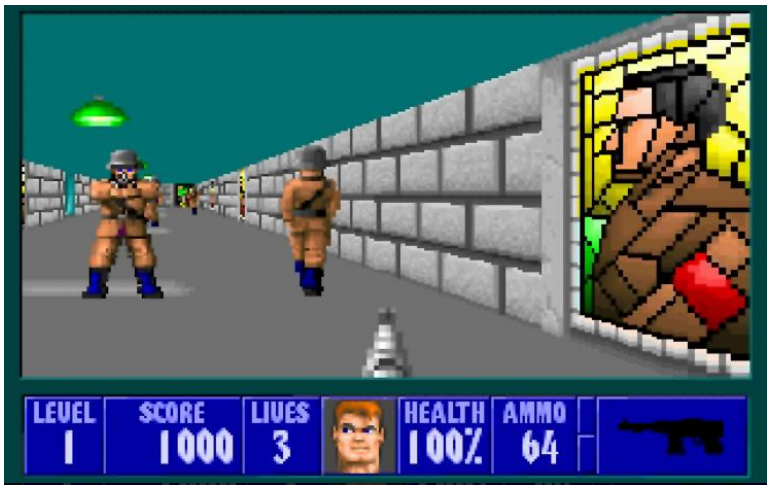


Рисунок 1.5–Wolfenstein 3D від idSoftware 1992 рік. [20]

1.2.2 Кидання променів (ray casting)

У методі ray casting сцена представляється у вигляді зображення на якому розмічені стіни. У цій сцені визначається положення камери і її кут огляду, а після з камери пускаються промені і для кожного з них шукається точка зіткнення. Кількість променів дорівнює кількості пікселів монітору по ширині.

Далі для кожного променя малюється вертикальний сегмент стіни, розмір якого залежить від того наскільки далеко від камери знаходиться точка зіткнення. Після відмалювання усіх вертикальних сегментів, виходить зображення стін з імітацією лінійної перспективи.

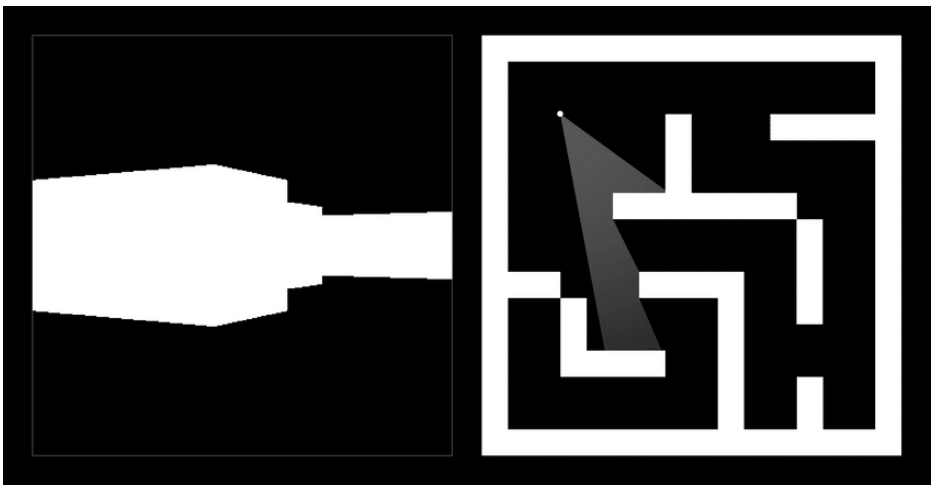


Рисунок 1.6 – Ray casting.[20]

Додатково можна пофарбувати стіни в певний колір або ж накласти на них текстуру. Для цього на карті стін, стіни помічаються певними кольорами кожен колір пов'язуєманий з певною текстурою. Методика відтворення при цьому не змінюється. При малюванні вертикальних сегментів стін буде братися відповідний сегмент з текстури.

Цей метод досить обмежений. Тривимірними є тільки стіни, а решта об'єктів реалізуються у вигляді спрайтів, які весь час спрямовані до камери.

1.2.3 Z – буферизація

Задля оптимізації, полігони які перекриваються іншими полігонами не беруть участі в відображенні У цьому випадку потрібно зрозуміти, чи варто малювати нові пікселі поверх старих або ж пропустити їх. Для цього використовується z-буфер. Який використовує додатковий масив, в якому кожному пікселю в растрі зображення присвоюється координата Z. Координата Z відповідає відстані на якій знаходиться точки об'єкта від площини на яку вона проектується вісь Z є перпендикулярна відносно площини проектування.

При відображенні, глибина пікселів порівнюється між собою і пріоритет віддається тим пикселям що ближче до екрану. На рис 7. зображено як працює z-буфер. Найближчий до камери рожевий трикутник

має значення 0,1, більш віддалені обє'кти мають більші значення і відповідно відмальовуються останніми.

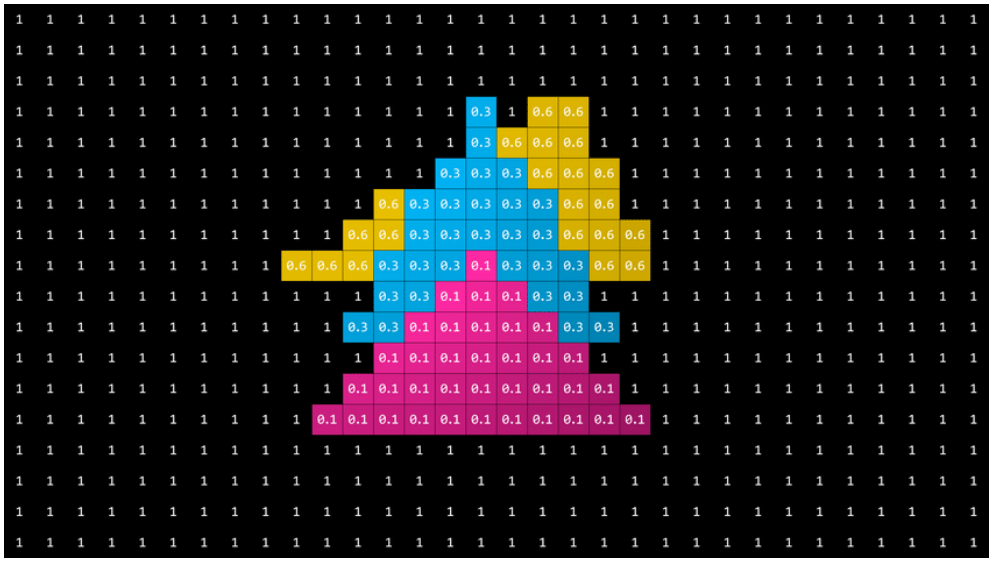


Рисунок 1.7 – Z – буферизація.[20]

У Z-буфері в його класичному вигляді розрядна сітка буфера недостатньо точна на близьких відстанях. Для вирішення цієї проблеми застосовується w-буфер, в якому застосовується не віддаленість, а зворотна їй величина (w = 1 / z). Що краще застосовувати - z-буфер або w-буфер - залежить від програми.

На сучасних відеоадаптерах робота з z-буфером забирає чималу частину пропускної здатності ОЗУ відеоадаптера. Для боротьби з цим застосовують стиснення без втрат: стиснення / відновлення забирає менше ресурсів, ніж звернення до пам'яті.

На початку кадру відбувається заповнення буфера деяким числом (наприклад, числом 1,0). Це також забирає деяку частку машинного часу, тому часто роблять так: перший кадр буферизация налаштовується так, щоб глибина ближніх об'єктів була 0,0, а дальніх – 0,5. Другий кадр – від 1,0 до 0,5. Це знижує точність на 1 біт, але дозволяє позбутися від очищення буфера.[5]

Хоча Z-буфер призначений саме для того, щоб обійтися без сортування видимих граней, швидкість роботи Z-буфера серйозно залежить від

сортування об'єктів. Тому движок повинен хоча б приблизно впорядкувати об'єкти від далеких до ближніх.

Якщо два об'єкти мають близьку Z-координату, іноді, в залежності від точки огляду, показується то один, то інший, то обидва смугастим візерунком. Це називається Z-конфлікт (англ. Z fighting). Найчастіше конфлікти притаманні спецефектів (декаль), накладається на основну текстуру, наприклад, дірок від куль.

Вирішуються Z-конфлікти зрушенням одного об'єкта щодо іншого на величину, що перевищує похибка Z-буфера.

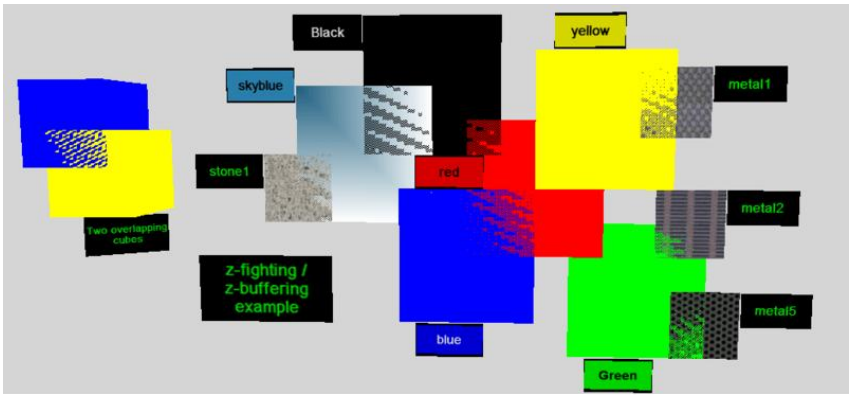


Рисунок 1.8 – Z-конфлікти.[20]

Недоліки використання методу Z-буфера.

- Z-буфер потребує пам'ять обсяг якої відповідає точності з якою читаються координати Z і розміару растра зображення. Зазвичай на один піксель Z-буфера використовують 16, 24 або 32 бітів.
- Ініціалізація Z-буфера зменшує швидкодію анімації. Але, часто використовують такий прийом – Z-буфер форматується один раз для декількох кадрів. Це стає можливим, якщо розділяти діапазон значень який приймає Z навпіл.
- Зайві операції. Для невидимих полігонів виконуються зайві операції. Невидимі полігони повинні бути відсіченні до циклу виведення. Щоб прискорити виведення використовують ієрархічний Z-буфер, який використовує 2 буфера з різною роздільною здатністю.
- Помилки при відображенні напівпрозорих об'єктів.[6]

1.2.4 Растеризація

Починаючи з 1994 року стали з'являтися ігри які використовували метод растеризації. Цей спосіб рендеринга і наразі використовується в тривимірних іграх. А першою грою яка повноцінно впровадила растеризування в свій движок можна вважати Descent, 1994 року. Варто сказати, що ніяких прискорювачів 3D-графіки на той момент ще не існувало. Так як не було загальноприйнятого методу рендеринга 3D графіки, то і прискорювати було нічого. І саме растеризация сприяла появі відеокарт з 3D прискорювачами.

Рисунок 1.9 – Descent, 1994.[20]

Растеризація – це процес перетворення вершинного (або алгебраїчного) опису фігури в піксельне уявлення. Растеризация часто називається перетворенням сканування, оскільки багато алгоритмів растеризації застосовують інкрементальні методи, скануючи фігуру по рядках (лініях) растра, експлуатуючи властивість когерентності.

Інкрементальні методи обчислюють нові значення швидко на основі значень, отриманих на попередньому кроці, замість того, щоб обчислювати нові значення безпосередньо, що часто виявляється досить повільним процесом. Когерентність в просторі або за часом - це термін, що означає, що близькі об'єкти (тобто рядки пікселів в нашому випадку) мають якісні характеристики подібні характеристикам поточного об'єкта. [5]

Перш ніж перейти до безпосередньо розглянути можливості перекладу математичного опису об'єкта. Необхідно, розглянути поняття зв'язності.

Зв'язності - можливість з'єднання двох пікселів растрової лінією, тобто послідовним набором пікселів. Необхідно зрозуміти, коли пікселі (x1, y1) і (x2, y2) вважаються сусідніми. Для цього використовують два поняття зв'язності:

1. Чотирьохзв’язність: пікселі можно вважати сусідніми, якщо або їх x-координати, або їх y - координати відрізняються на одиницю:

$|x_1 - x_2| + |y_1 - y_2| \leq 1;$

2. Восьмизв'язність: пікселі можно вважати сусідніми, якщо їх x- координати і y-координати відрізняються не більше ніж на одиницю:

$|x_1 - x_2| \leq 1, |y_1 - y_2| \leq 1;$

На Рисуноку 1.10 зображені чотирьохзв’язна і восьмизв'язна лінії.[5]

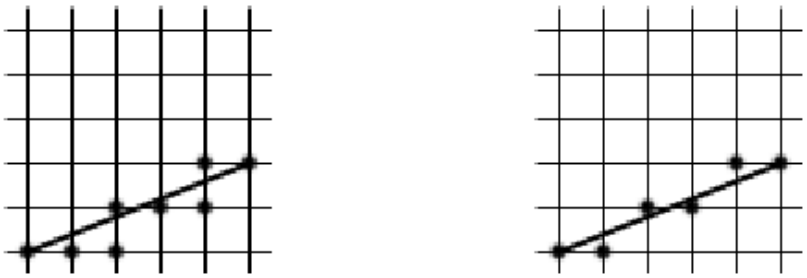


Рисунок 1.10 – Класифікація растрових алгоритмів

Задача побудови растрового зображення відрізка в з'єднанні точок $A(x_a, y_a), B(x_b, y_b)$. Для простоти можнл вважати, що $0 \leq y_b - y_a \leq x_b - x_a$Тоді відрізок описується рівнянням:[5]

$$y = y_a + \frac{y_b - y_a}{x_b - x_a}(x - x_a), x \text{ є } [x_a, x_b] \text{ або } y = kx + b \qquad (1)$$

Наведено найпростіші покрокові алгоритми для побудови відрізка вони мають ряд недоліків:

1. Операції виконуються над числами з плаваючою точкою, але краще працювати з целочисленою арифметикою;

2. Для кожного кроку необхідно виконати операцію округлення, що знижує швидкодію.[6]

В наступному алгоритмі Брезенхейма ці недоліки усунуті. Як і в попередньому випадку, вважається, що тангенс кута нахилу відрізка приймає

значення в діапазоні від 0 до 1. Розглянуто і-й крок алгоритму (Рисунок 1.11). На цьому етапі піксель P_{i-1}вже визначений як найближчий до реального відрізку. [5]

Необхідно визначити, який з пікселів (T_i або S_i) буде встановленонаступним.

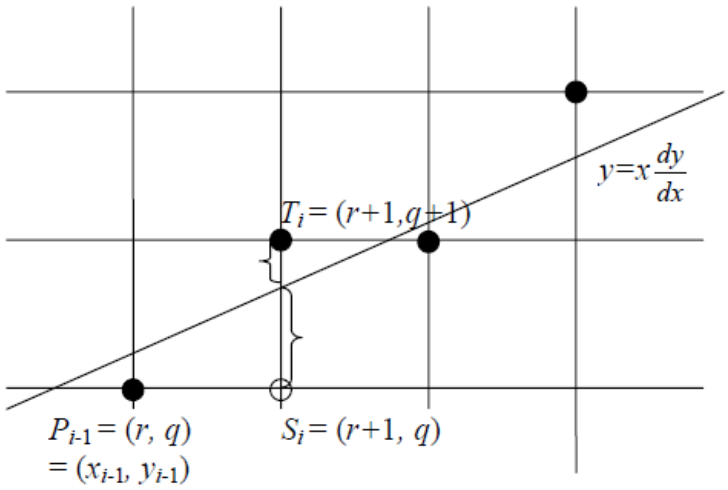


Рисунок 1.11 – і-й крок алгоритму Брезенхейма

Алгоритм використовує керуючу змінну d_i, яка пропорційна різниці між S і T на кожному кроці. Якщо S <T, то S_i ближче до відрізка, в іншому випадку береться T_i. Якщо відрізок що зображується проходить з точки $(x_1,\ y_1)$ в точку $(x_2,\ y_2)$ виходячи з початкових умов, точка $(x_1,\ y_1)$ближче до початку координат. Тоді обидва кінці відрізка переносяться за допомогою перетворення $T(-x_1,-y_1)$ так щоб перший кінець відрізка збігся з початком координат. Початковою точкою відрізка стала точка (0, 0), кінцевою точкою - (d_x,d_y), де $d_x = x_2 - x_1,\ d_y = y_2 - y_1$ (Рисунок 1.12).[5]

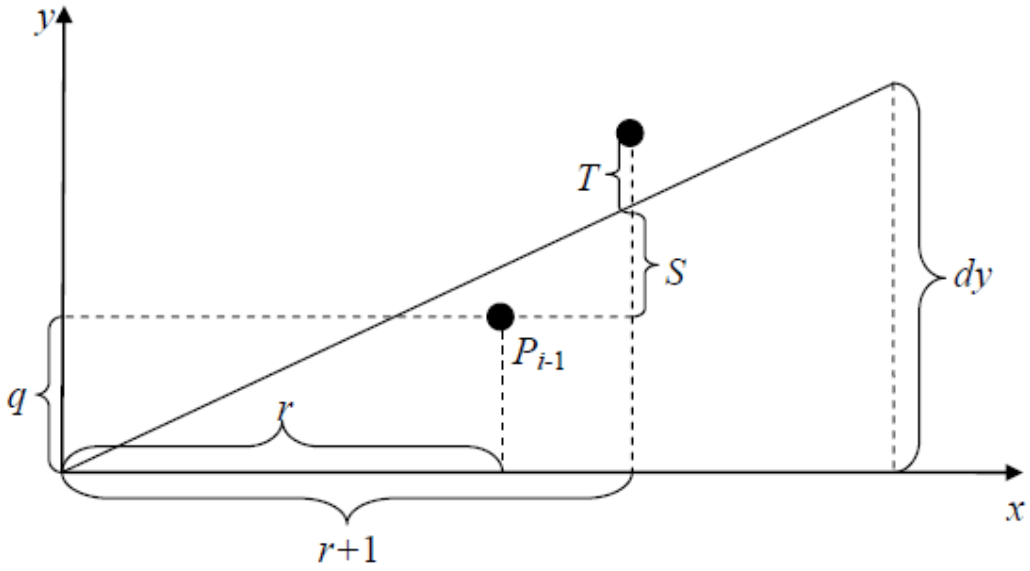


Рисунок 1.12 – Вид відрізка після переносу на початку координат

Ітеративна формула обчислення керуючого коефіцієнта d_{i+1} за попереднім значенням d_i. [5]

$$d_i + 1 = d_i + 2\,d_y - 2\,d_x\,(y_i - y_{i-1}). \quad (2)$$

За допомогою керуючого коефіцієнта вибирається наступний піксель - S_i або T_i

Якщо $d_i \geq 0$, тоді вибирається T_i і $y_i = y_i - 1 + 1, d_i + 1 = d_i + 2\,(d_y - d_x)$

Якщо $d_i < 0$, тоді вибирається S_i і $y_i = y_i - 1$ і $d_i + 1 = d_i + 2\,d_y$

Початкові значення d_1 з урахуванням того, що $(x_0, y_0) = (0, 0)$

$$d_i = 2\,d_y - d_x \quad (3)$$

Перевагою алгоритму є те, що для роботи алгоритму потрібні мінімальні арифметичні можливості: додавання, віднімання і зрушення вліво для множення на 2.[5]

Якщо $d_y > d_x$, то необхідно буде використовувати цей же алгоритм, але крок за кроком збільшуючи у і на кожному кроці обчислювати х.

Всі об'єкти складаються з трикутників, вони ж полігони. Трикутники

складаються з мінімальної кількості вершин, якими можна описати площину. Відповідно вони не можуть бути опуклими або увігнутими і тому є мінімальною структурною одиницею.

Щоб намалювати об'єкт, нам потрібно намалювати кожен полігон з якого він складається. Для цього всі вершини проектуються на площину екрану. Потім для кожного полігону закрашуються пікселі між його вершинами. Весь процес зображений на картинках нижче.

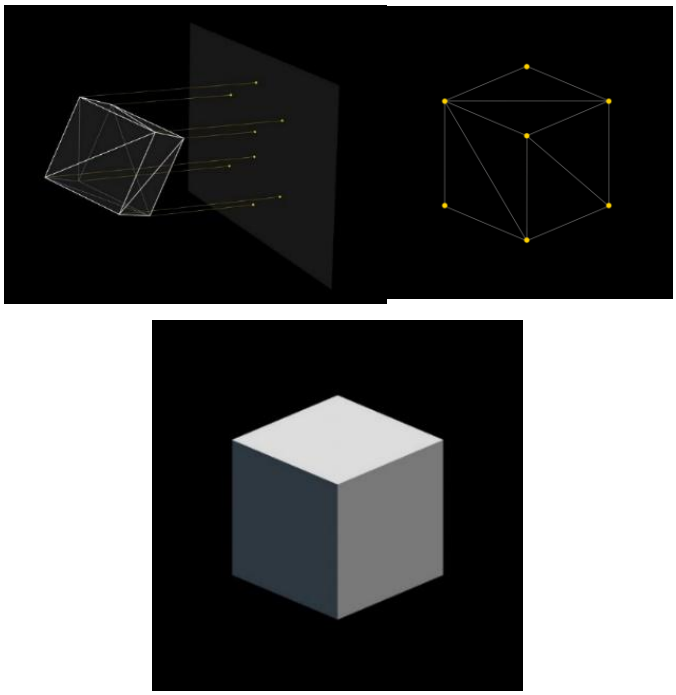


Рисунок 1.13 – Процес відмальовки об’єкта.[20]

Проекція точок на екран називається ортогональною проекцією. І вона не зовсім підходить для візуалізації 3D графіки, так як в ній відсутня перспектива. Перспектива - це коли віддалені об'єкти здаються менше, а паралельні лінії сходяться в одній точці. І щоб досягти такого ефекту нам потрібно використовувати перспективну проекцію.

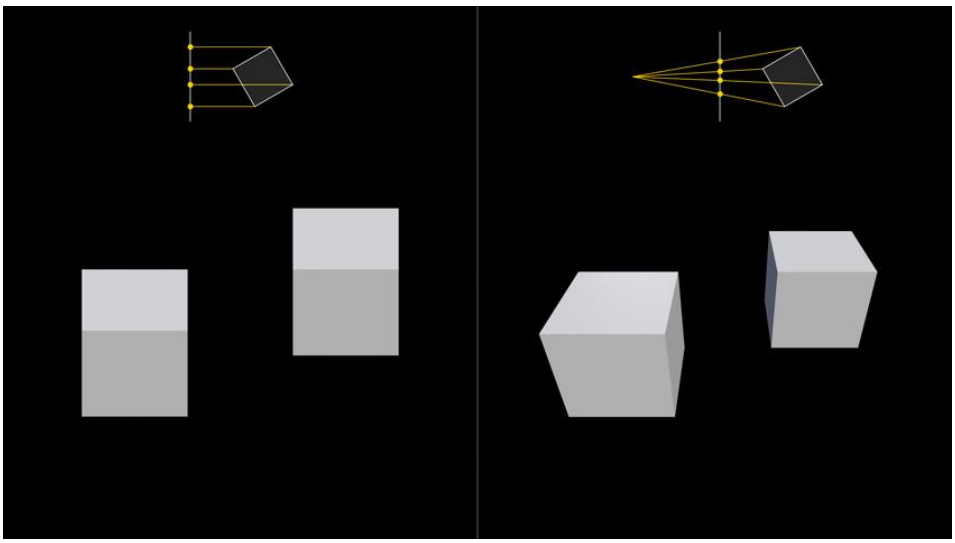


Рисунок 1.14 – (а) ортогональна, (б) перспективна проекції.[20]

З зображення вище можна побачити. В перспективній проекції з кожної вершини проводяться лінії до точки за екраном, назвемо її точкою огляду. Місця, де ці лінії перетинають екран і є проектувати точками. Від положення точки огляду безпосередньо залежить область видимості камери. Технічно така проекція реалізується дуже нескладно, так як це проста геометрія.

У комп'ютерній графіці всі геометричні трансформації і проекції виконуються не за допомогою формул, а шляхом множення вершин на матриці. Числа в матриці визначають якесь перетворення: переміщення, поворот і т. д. Можна перемножити між собою 3 матриці і тоді результуюча матриця буде містити в собі відразу всі 3 перетворення. Таким чином можна помножити кожну вершину на одну матрицю, а не на три, що сильно економить обчислювальні ресурси. Власне цим і займаються відеокарти - множенням матриць.

Перша відеокарта з прискорювачами 3D графіки була випущена в 1996 році компанією 3dfx Interactive, а називалася вона Voodoo Graphics. І в ній не було прискорювача для двовимірної графіки, а значить доводилося ставити дві відеокарти: для 2D і 3D графіки.

Перейдемо до недоліків методу растеризації. Один з них - це складність створення реалістичного освітлення. Для затінення об'єктів для

кожного полігону обчислюється кут під яким на нього подають сонячні промені, тобто кут між нормаллю полігону і сонячним промінням. Він визначає освітленість полігону і відповідно колір пікселів. Але при такому підході можна бачити чіткі межі полігонів. Щоб них позбутися, проводиться інтерполяція кутів між нормалями. Залежно від властивостей матеріалу можна реалізувати відблиски, зміну відтінку і т. д. Більшість ігор сьогодні використовують PBR (physically based rendering) - фізично коректний рендеринг. Який описує, як повинні матеріали реагувати на світло.

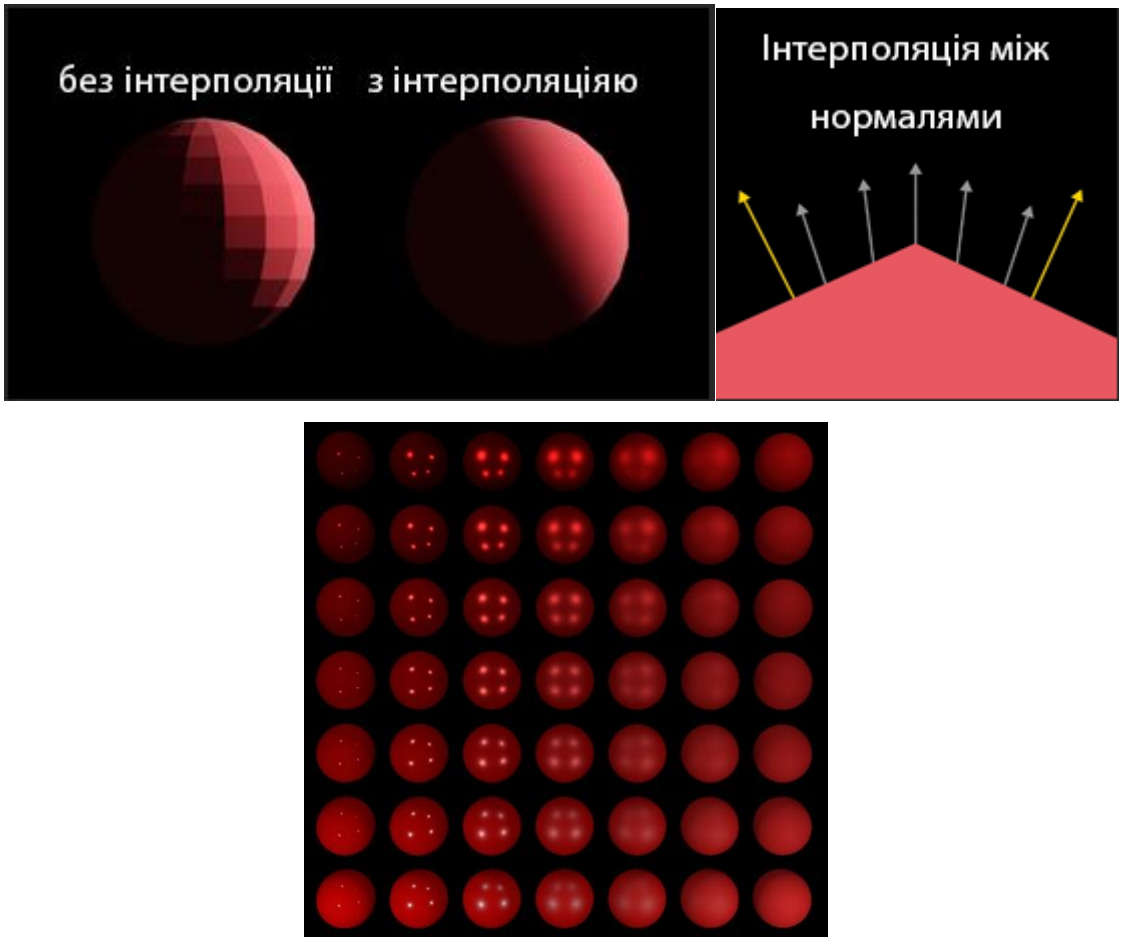


Рисунок 1.15 – (а) і (б) інтерполяція, (в) реакція матеріалів на світло, згідно PBR.[20]

Але вся фізична коректність руйнується, коли справа доходить до тіней. Єдине що можливо зробити в растеризации - це імітувати тіні. Самий банальний спосіб відображення тіней полягає в рендерингу сцени з позиції джерела світла. Після чого ми можемо отримати дані про те які частини сцени знаходяться в тіні і знизити освітленість відповідних пікселів.

Але цей метод немає достатньої точності і іноді тіні зміщюються щодо об'єктів. Цей класичний недолік отримав забавну назву «Пітер Пеннінг» (Peter Panning). Але тінь абсолютно не реалістична. У житті краї тіні розмиваються в міру віддалення від об'єктів, а тут тінь виходить абсолютно жорсткою. Її можна хіба що розмити по всьому периметру.

Насправді м'які тіні розробники вже навчилися робити. Але алгоритми рендеринга таких тіней досить складні і вимогливі до ресурсів. Цікаво що проблема настільки серйозна, що деякі відеокарти реалізують апаратне прискорення для певних методів рендеринга тіней.

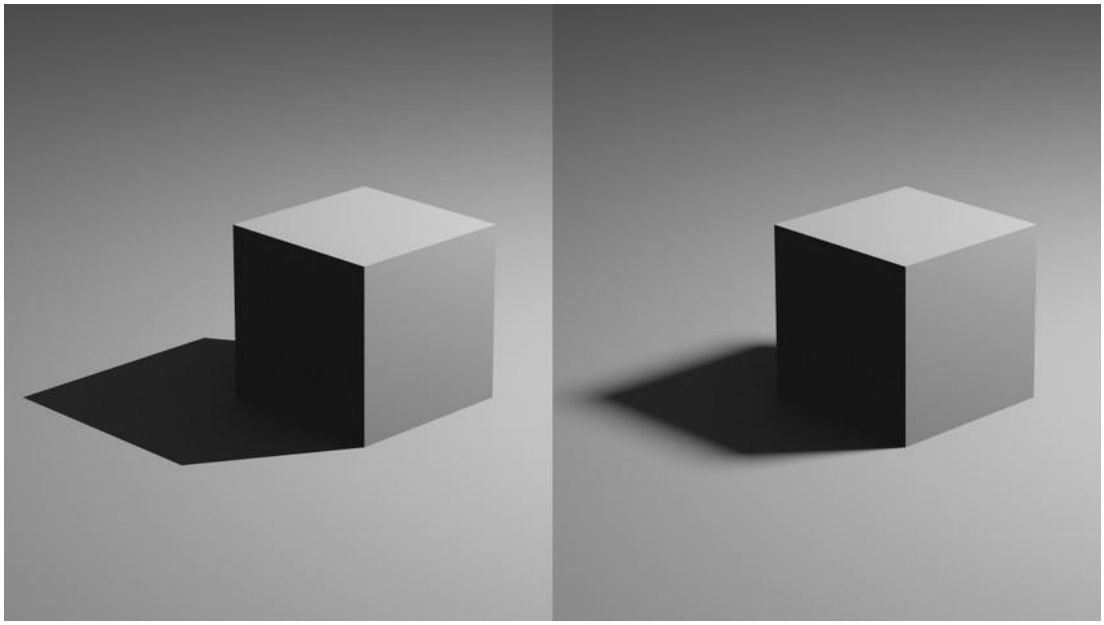


Рисунок 1.16 - Рендеринг тіней.[20]

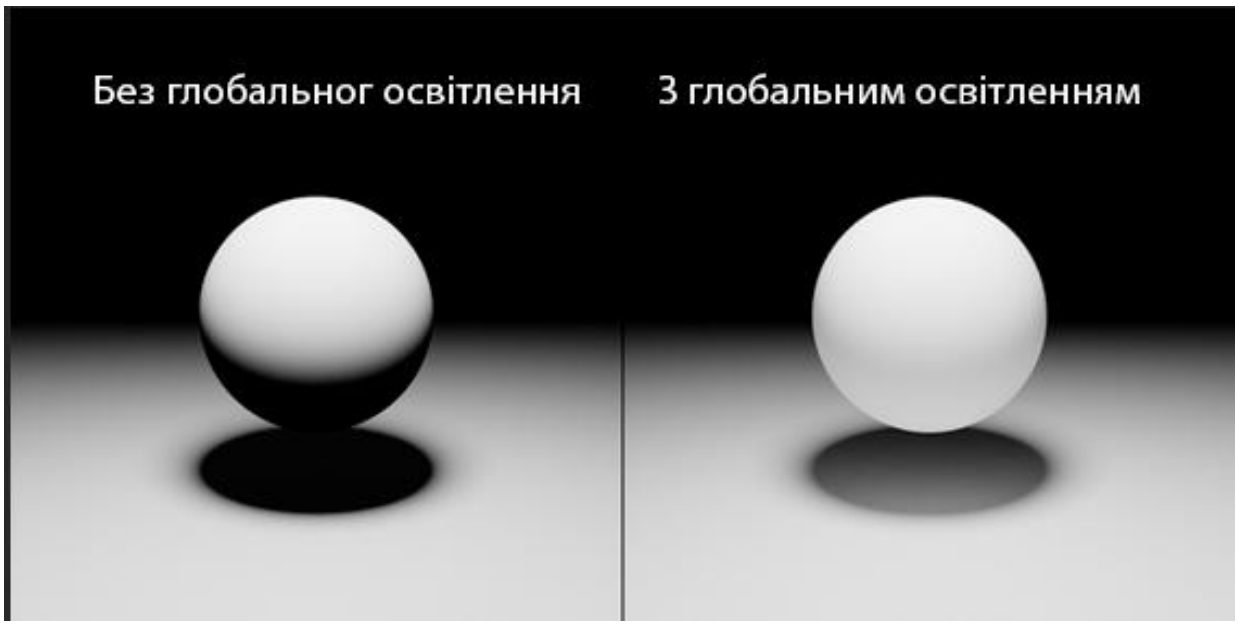


Рисунок 1.17– Глобальне освітлення.[20]

Сфера справа здається занадто світлою, але саме такою вона і повинна бути, її висвітлює площина під нею.

Також освітлення в растеризації страждає через неможливість реалізувати глобальне освітлення (global illumination). Справа в тому що промені світла постійно відзеркалюються і висвітлюють місця куди світло не потрапляє на пряму. Тому тіні ніколи не бувають абсолютно чорними. На зображенні нижче можна побачити наскільки величезною може бути різниця в освітленні, якщо не враховувати глобальне освітлення.

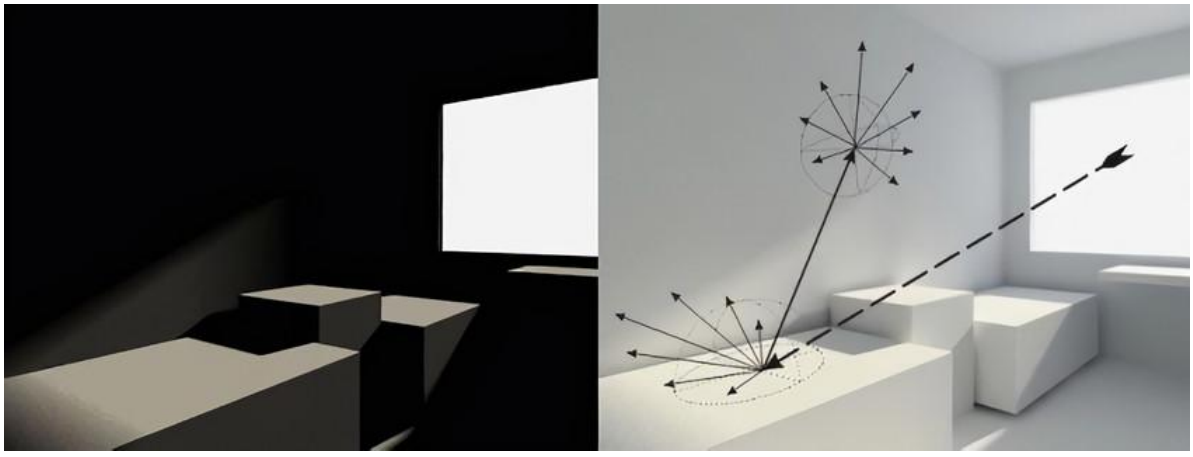


Рисунок 1.18 - Глобальне освітлення.[20]

Неможливо створити і достовірні відображення. Є один хак який допомагає частково вирішити цю проблему - це створення кубічних карт. Кубічна карта (cube map) – це зображення яке зберігає в собі простір навколо

об'єкта. З їх допомогою можна прорахувати, що саме повинно відображатися в певній точці. Але вони не підходять для динамічних сцен, де об'єкти або джерела світла можуть рухатися, тому що об'єкти будуть відображати тільки те, що збережено в кубічної карті.

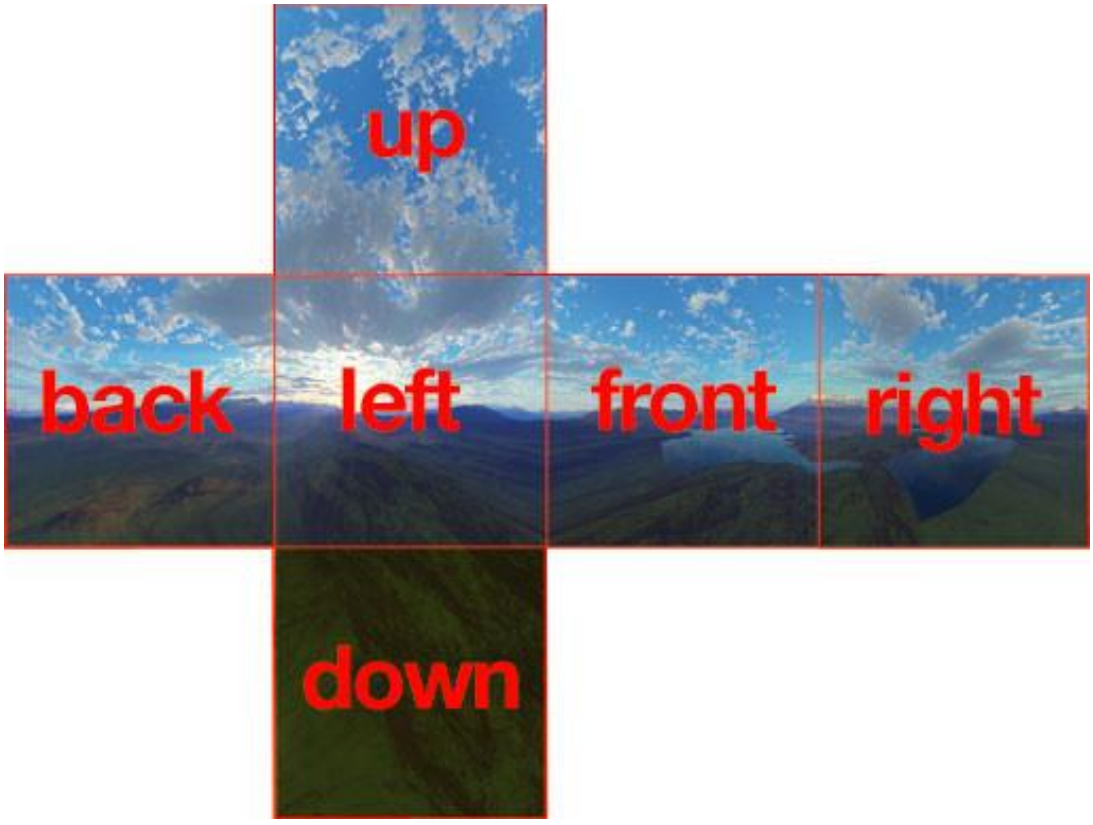


Рисунок 1.19 - Кубічна карта.[20]

Класична проблема при використанні кубічних карт – персонаж не відзеркалюється на скляних дверях.

Останнє на що варто звернути в методі растеризации - це проблема сортування. Виникає вона при рендерингу напівпрозорих об'єктів. У сцені знаходиться дві напівпрозорі панелі. При цьому зелений куб за першої панеллю не відображається. Щоб зрозуміти чому це сталося не обхідно вспам’ятати про z-буфер. Якщо на місці відтворення полігону вже присутні пікселі іншого полігону, який ближче до камери, то відмальовка нових пікселів пропускається. Тобто, на сцені нижче, спочатку намальовані напівпрозора панель, а потім куб. Так як панель ближче до камери, при відображенні куба пікселі які знаходяться за панеллю були пропущені.

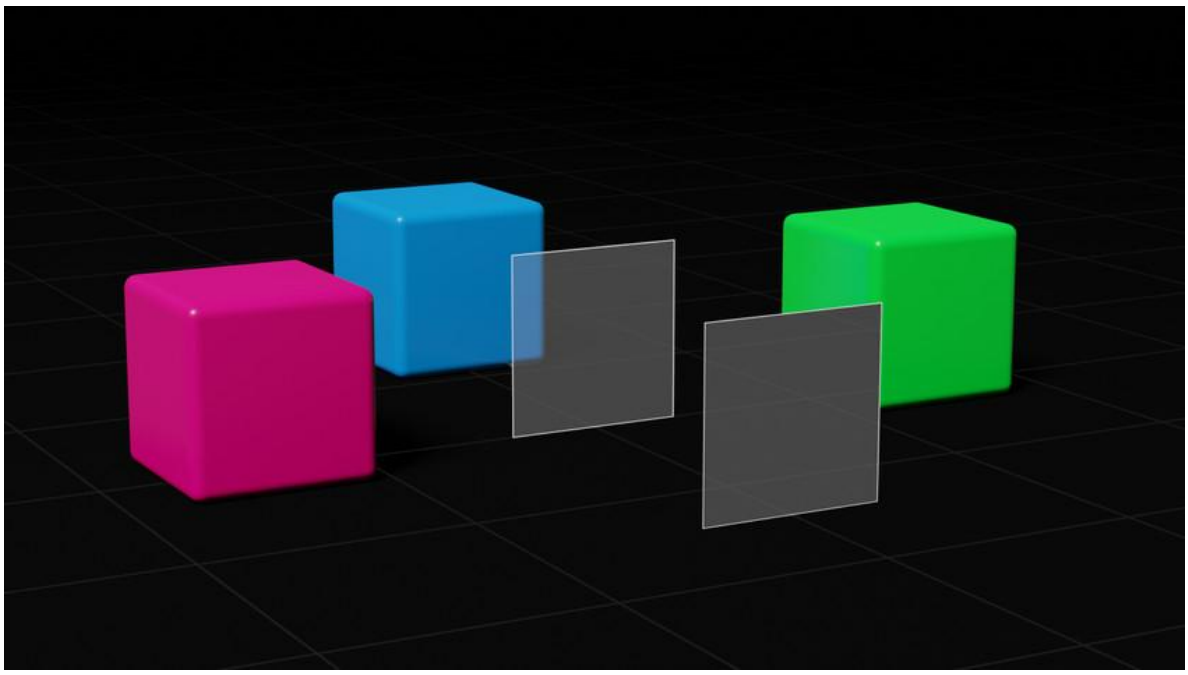


Рисунок 1.20 – Некоректна робота напівпрозорих панелей.[20]

Щоб уникнути цієї проблеми, об'єкти малюються впорядку від дальнього до ближнього. Таким чином напівпрозорі пікселі будуть налагоджуватися на попередні. Але можливі ситуації коли об'єкти в яких напівпрозорі полігони перетинаються з непрозорими і тоді сортуванням об'єктів не підходить. Таких ситуацій намагаються уникати. Але в при необхідності реалізується метод order-independent transparency (порядку-незалежної прозорості). В цьому методі кожноий піксель зберігається в точці, не залежно від того яку глибину він має. Далі ці пікселі упорядковуються між собою по глибині і малюються в необхідному порядку. Тобто сортуються не об'єкти, а пікселі. Такий метод вимагає більше пам'яті, менш продуктивний і складніший в реалізації, тому його складно вважати панацеєю.

Що ж, растеризация має безліч обмежень, проте до цих пір використовується як основний метод візуалізації в іграх. Все завдяки тому, що цей метод досить продуктивний, при відтворенні в реальному часі.

1.2.4 Трасування променів (Ray tracing)

Трасування променів (Ray Tracing) наразі можно вважати

найпотужнішим і найбільш універсальним методом створення фотореалістичних зображень. Існує багато методів якими реалізується алгоритми трасування для відображення найскладних тривимірних сцен. Універсальність методів трасування обумовлена тим, що вони основані на простих поняттях, які базуються на людському досвіді сприйняття навколишнього світу. Принцип трасування променів лежить в імітації сонячних променів. Коли промінь світла потрапляє на поверхню, він може відбитися в будь-якому напрямку. Це тому що навіть гладкі об'єкти мають мікроскопічні нерівності. Якщо ж об'єкт дійсно гладкий, то ми отримуємо дзеркальний матеріал. В цьому випадку всі промені відбиваються приблизно в одному напрямку.

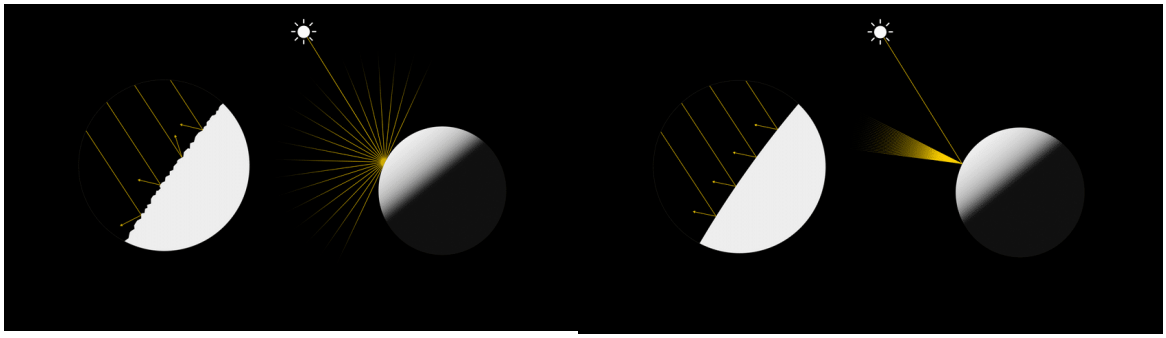


Рисунок 1.21 – Відбиття.[20]

Метод зворотного трасування променів дозволяє значно скоротити перебір світлових променів. Метод розроблений в 80-х роках, основними вважаються роботи Уітта й Кея. Відповідно до цього методу відстеження променів здійснюється не від джерела світла, а в зворотньому напряму - від точки спостереження. Так враховуються тільки ті промені, які вносять вклад у формування зображення.

В першу чергу шукається точка зіткнення з об'єктом, потім будується промінь до джерела світла. Як і в растеризации, кут між нормаллю полігону і падаючим світлом визначає освітленість. Якщо промінь на своєму шляху до джерела світла перетинає будь-який об'єкт значить точка знаходиться в тіні і освітленість нульова.[6]

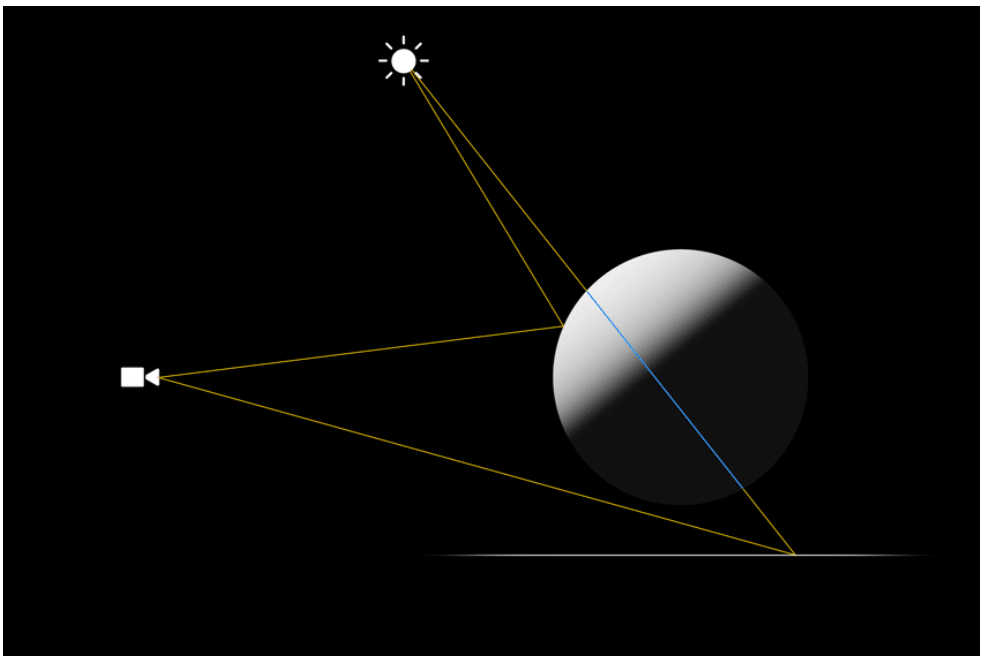


Рисунок 1.22 – Перетин об’єкта.[20]

Наразі такий підхід нічим не краще растеризації. І навіть гірше, адже потрібно шукати зіткнення з об'єктами, а це набагато ресурсовитратностішн ніж просто проектувати полігони на екран. Тому зазвичай використовується більш складна версія трасування шляхів, яка називається path tracing, або ж трасування шляхів. Вона дозволяє симулювати багатократне віддзеркалення сонячних променів і тим самим реалізувати ефект глобального освітлення.

У цій варіації алгоритму з точки зіткнення, крім променя до джерела світла, впадає ще кілька додаткових променів у випадковому напрямку. Для кожного такого променя так само визначається зіткнення і освітленість в цій точці. Після чого ми можемо визначити вплив кожного такого променя на освітленість нашої початкової точки (пікселя). Щоб результат був більш реалістичним ми можемо рекурсивно повторювати процес кидання додаткових променів для кожного додаткового променя.[6]

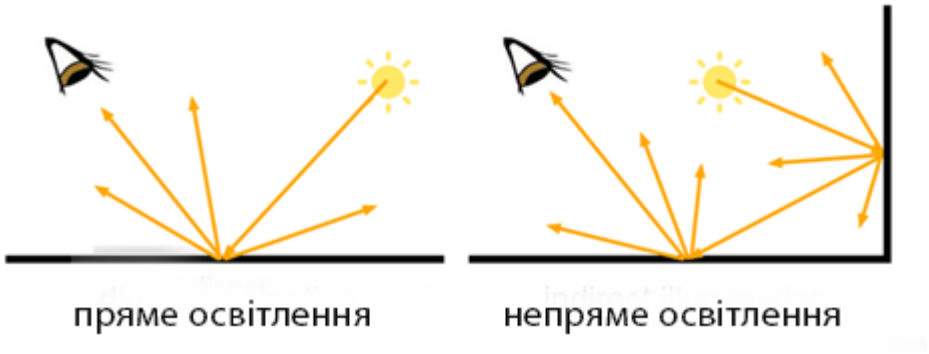


Рисунок 1.23 – Відбиття при прямому та непрямому освітленні.[20]

Також можуть бути різні типи променів для різних матеріалів. Наприклад, заломлючі промені для скла або відзеркалючі промені для дзеркальних матеріалів. Все це залежить від реалізації алгоритму. Різні движки рендеру реалізують трасування променів по різному, але принцип один, ми кидаємо промені в різних напрямках щоб оцінити вплив навколишніх об'єктів на освітленість в конкретній точці.

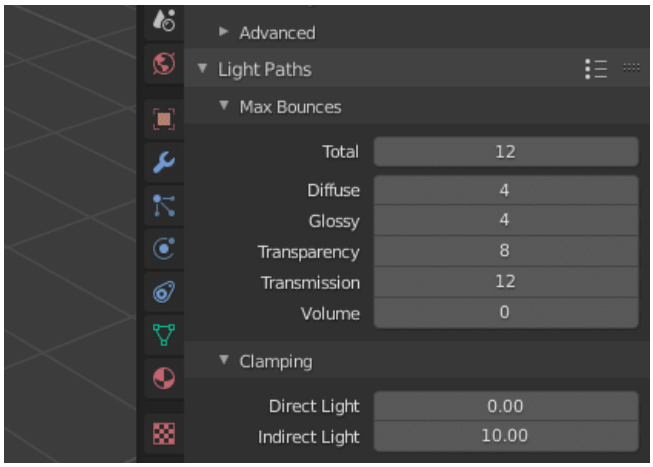


Рисунок 1.24 – Налаштування кількості різних променів в програмі Blender

Кількість додаткових променів обмежена, тому оцінка впливу навколишніх об'єктів на початкову точку не надто точна. Через це в сусідніх точках освітленість може відрізнятися навіть якщо вона повинна бути однаковою. Із-за цього в зображеннях створених з використанням трасування променів присутній шум. Для позбуття шуму необхідна збільшена кількість променів, але це значно збільшує час рендерингу. Простіше використати денойзери (denoisers). Єфективніші денойзери використовують штучний інтелект. Приклади таких денойзорів, Intel Open

Image Denoise або NVIDIA OptiX. Вони прибрають шум за кілька секунд.

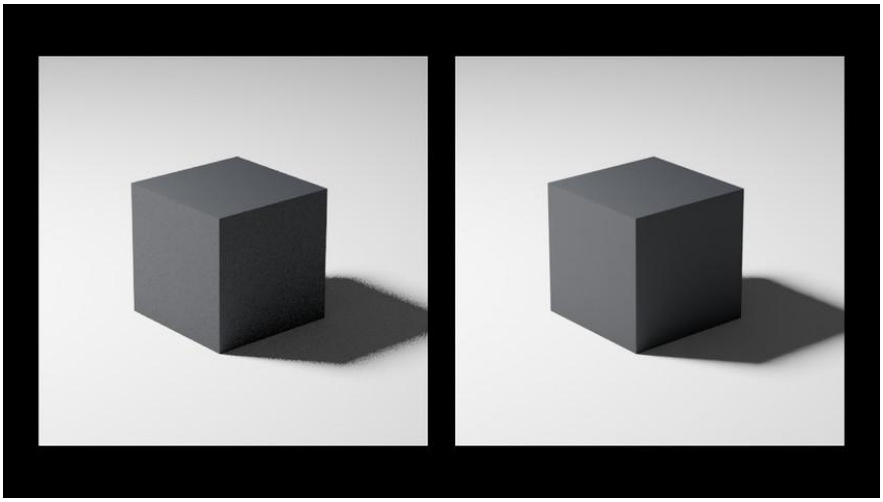


Рисунок 1.25 – Зліва - оригінальне зображення, праворуч - із застосуванням NVIDIA OptiX.[20]

Якщо об'єкт дзеркальний, то необхідно побудувати вторинний промінь падіння, попередній, первинний, трасуємий промінь важається промінем відображення.

Для побудови ідеального дзеркала можно простежити лише чергову точку перетину вторинного променя з деяким об'єктом. Ідеальне дзеркало має ідеально рівну відполіровану поверхню, тому одному відбитому променю відповідає тільки один падаючий промінь. Дзеркало може бути затемненим, тобто поглинати частина світлової енергії, але все одно виконується правило: один промінь падає - один відбивається. Можна розглядати також "недосконале дзеркало". Це буде означати, що поверхня нерівна. Напрямку відбитого променя будуть відповідати кілька падаючих променів (або навпаки, один падаючий промінь породжує кілька відображених променів), які утворюють деякий конус, можливо, несиметричний, з віссю уздовж лінії падаючого променя ідеального дзеркала. Конус відповідає деяким законом розподілу інтенсивностей, найпростіший з яких описується моделлю Фонга - косинус кута, зведений в деяку ступінь. Недосконале дзеркало різко ускладнює трасування - потрібно простежити не один, а безліч падаючих променів, враховувати внесок випромінювання від інших видимих з даної точки об'єктів.

Для прозорого об'єкта необхідно побудувати новий промінь, який при залоленні давав попередній трасуємий промінь. Також можна використати оборотність, яка є справедливлою і для заломлення.

Коли у об'єкта присутні властивості дифузного заломлення і віддзеркалення, то як і для неідеального дзеркала, виникає необхідність трасувати промені, які надходять з усіх наявних об'єктів. Для дифузного віддзеркалення інтенсивність відбитого світла пропорційна косинусу кута між вектором променя від джерела світла і нормаллю. Джерелом світла може бути будь-який об'єкт видимий з даної точки, який може передавати світлову енергію.

Коли з'ясовується, що промінь зворотного трасування не перетинає об'єкт, а направлений у вільний простір, то на трасування для нього закінчується.

За практичної реалізації методу зворотного трасування вводяться обмеження. Вони необхідні, щоб можна було в вирішити задачу створення зображення, а деякі обмеження значно підвищити швидкість трасування.

Приклади обмежень.

- Серед усіх типів об'єктів виділяються деякі, які назвемо джерелами світла. Джерела світла можуть тільки випромінювати світло, але не можуть його відображати або заломлювати (будемо розглядати тільки точкові джерела світла).
- Властивості поверхонь, що відбивають описують сумою дифузної і дзеркальної компонентів.
- Дзеркальність описується двома складовими. Reflection, що враховує відбиття від об'єктів, що не є джерелами світла. Створюється тільки один дзеркально відбитий промінь для подальшого трасування. Specular, означає світлові відблиски від джерела світла. Промені направляються на всі джерела світла і далі визначаються кути, утворені цими променями і дзеркально відбитим променем зворотного трасування.

- При дифузному відображенні враховують тільки промені від джерел світла. Ті промені що відображають від дзеркальних поверхонь ігноруються. Коли промінь закривається іншим об'єктом, це означає, що дана точка знаходиться в тіні.
- Для завершення трасування можно внести граничне значення освітленості, яке не робить внесок в результуючий колір, або обмежує кількість ітерацій.

За моделлю Уиттеда колір деякої точки об'єкта можна визначити сумарною інтенсивністю:[6]

$$I(\lambda) = K_a I_a(\lambda) C(\lambda) + K_d I_d(\lambda) C(\lambda) + K_s I_s(\lambda) + K_r I_r(\lambda) + K_t I_t(\lambda) \qquad (4)$$

де λ - довжина хвилі,

C (λ) - заданий вихідний колір точки об'єкта,

K_a, K_d, K_s, K_r, K_t- коефіцієнти, що враховують властивості конкретного об'єкта через параметри фонового підсвічування, дифузного розсіювання, дзеркальності, відображення і прозорості,

I_a- інтенсивність фонового підсвічування,

I_d- інтенсивність, що враховується для дифузного розсіювання,

I_s- інтенсивність, що враховується для дзеркальності,

I_r- інтенсивність випромінювання, що приходить по відбитому променю,

I_t- інтенсивність випромінювання, що приходить по заломлення променя.

Інтенсивність фонового підсвічування для деякого об'єкта зазвичай константа. Формули для інших інтенсивностей. Для дифузного віддзеркалення:[6]

$$I_d = \sum_i I_i(\lambda) \cos\theta_i \qquad (5)$$

де $I_i(\lambda)$- інтенсивність випромінювання i-го джерела світла, θ_i - кут між нормаллю до поверхні об'єкта і напрямком на i-vi джерело світла.

Для дзеркальності:

$$I_d = \sum_i I_i\,(\lambda) \cos \alpha_i{}^p \qquad (6)$$

де p – це показник ступеня (відповідно до моделі Фонга) від одиниці до кількох сотень, α_i-кут між відбитим променем і напрямком на г'-те джерело світла.[6]

Інтенсивності випромінювань проходячих по відбитому променю (I_r), а так само по заломлення променя (I_t), множать на коефіцієнт, що враховує ослаблення інтенсивності в залежності від відстані, пройденої променем. такий коефіцієнт записуется у вигляді $e^{-\beta d}$де d - пройдену відстань, β - параметр ослаблення, враховує властивості середовища поширення проміня.

Для первинного променя необхідно, щоб був заданий напрямок, який би відповідав обраній проекції. При центральній проекції первинні промені будуть розходяться із загальної точки, паралельна проекція використовує паралельні первинні промені. Луч задається координатами початкової і кінцевої точок відрізка, або координатою початкової точки і напрямком, або ще як-небудь. Завдання первинного променя однозначно визначає проекцію зображуваної сцени. При зворотного трасуванні променів будь-які перетворення координат взагалі не обов'язкові. Проекція виходить автоматично - в тому числі, що не тільки плоска, а й, наприклад, циліндрична або сферична. Це один із проявів універсальності методу трасування.

В ході трасування променів необхідно визначати точки перетину прямої лінії променя з об'єктами. Спосіб визначення точки перетину залежить від того, кокой це об'єкт, і яким чином він представлений в певній графічній системі. так, наприклад, для об'єктів, представлених у вигляді багатогранників і полігональних сіток, можна використовувати відомі методи визначення точки перетину прямої і площині, розглянуті в аналітичній геометрії. Однак, якщо ставиться завдання визначення перетину променя з гранню, то необхідно ще, щоб знайдена точка перетину лежала всередині контуру межі.

Відомо кілька способів перевірки довільної точки на приналежність

полігону. Розглянемо два різновиди, по суті, одного і того ж методу (Рисунок 1.26).[6]

Перший спосіб. Знаходяться всі точки перетину контуру горизонталлю, яка відповідає координаті Y заданої точки. Точки перетину упорядковано відповідно до зростанню значень координат X. Пари точок перетину утворюють відрізки. Якщо точка, яка перевіряється, відноситься до одного з відрізків (для цього порівнюються координати X заданої точки та кінців відрізків), то вона - внутрішня.

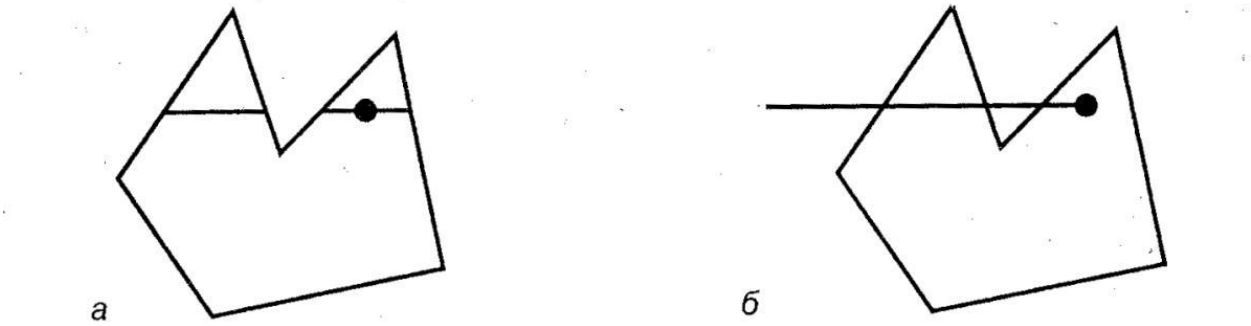


Рисунок 1.26 – Точка – внутрішня, якщо: а – точка належить перетинаючому відрізку, б – кількість перетинів непарна.

Другий спосіб. Знаходиться точка, що належить горизонталі випробуваної точки, також необхідно, щоб вона знаходилась за контуром полігону. Знайдені зовнішня і випробувана точки є кінцями горизонтального відрізка. Точки перетину визначаються з контуром полігону. Коли кількість перетинів є непарна, зрозуміло, що випробувана точка внутрішня. Коли промінь перетинає кількох об'єктів, обирають найближчу точку за напрямком поточного променя.

Плюси цього методу:

- Метод є універсальний, його можно застосувати для створення зображення складних просторових схем. Цей метод втілює багато з законів геометричної оптики. Легко реалізуються різноманітні проекції.
- Спрощені варіанти даного методу все одно дозволяють отримати реалістичні зображення. Якщо використовувати тільки первинні промені (з точки проектування), то це надає видалення невидимих

точок.

- Перетворення координат лінійні, це дозволяє працювати з текстурами досить просто.
- Для одного пікселя растрового зображення можна трасувати трохи близько розташованих променів, а потім усереднювати їх колір для усунення сходового (ступеневої) ефекту (антиалиасинг).
- Оскільки розрахунок окремої точки зображення виконується незалежно від інших точок, то це може бути ефективно використано при реалізації даного методу в паралельних обчислювальних системах, в яких промені можуть трассіроваться одночасно.

Недоліки

- Проблеми з моделюванням дифузного віддзеркалення і заломлення
- Для кожної точки зображення необхідно виконувати багато обчислювальних операцій. Трасування променів належить до числа найбільш повільних алгоритмів синтезу зображень.[6]

1.2.7 Маршірування променів (ray marching)

Це дуже простий метод візуалізації.

1. Для пікселя на екрані визначається відстань до найближчого об'єкта.

2. Ця відстань визначає радіус на який ми можемо пустити наш промінь.

3. Для кінцевої точки променя пункти 1 і 2 повторюються до тих пір поки наступний радіус не стане досить маленьким, що буде означати зіткнення з об'єктом. Або ж радіус може почати стабільно збільшуватися, якщо промінь пройшов повз всіх об'єктів.

Виконавши цей процес для всіх пікселів отримаються зображення сцени. Для пікселів які відображають зіткнення з об'єктом можна навіть реалізувати найпростіше затінення спираючись на кут під яким падає світло. Але навіщо взагалі потрібен цей метод, коли є інші? Як можна помітити основна дія яке в ньому відбувається - це пошук відстані до об'єкта. Зазвичай щоб знайти мінімальну відстань, нам потрібно знайти відстань до кожного полігону з якого складається об'єкт і вибрати менше. Але це вимагає багато

ресурсів, тому ray marching використовується для рендеринга об'єктів для яких існує функція відстані (signed distance function).

Функція відстані дозволяє нам обчислити відстань до об'єкта використовуючи інформацію про його положення в просторі. І нам не потрібно вважати відстань до кожного полігону. Насправді полігони в ray marching навіть не використовуються. Якщо нам потрібно відобразити сферу, то ми просто вказуємо її положення і радіус, і цього буде достатньо щоб її зобразити. Функція відстані існує для більшості простих фігур, але також для складних фрактальних структур.

Рисунок 1.27 – Фрактал під назвою «Оболонка Мандельброта».[20]

Ще функції відстані можна модифікувати, щоб виконувати булеві операції над об'єктами або плавно зливати об'єкти в один. Це все робиться без будь-яких додаткових обчислень.

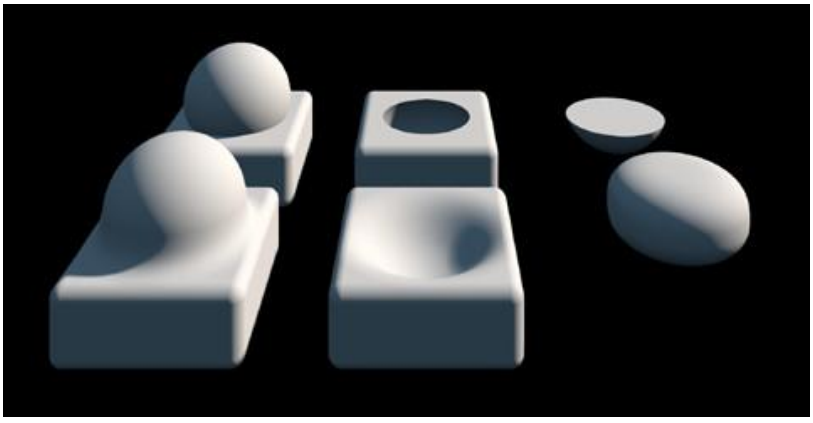


Рисунок 1.28 – Булеві операції.[20]

Найчастіше цей метод використовується для рендеринга воксельної

геометрії. Так як вокселі - це куби, а відстань до куба розрахувати дуже просто, то можна відображати сцени які складаються з мільйонів вокселей. Відразу спадає на думку гра Teardown. Так, вона повністю рендерится з використанням методу ray marching. Коли вся сцена складається з кубів, то реалізувати хорошу графіку не складно. Але якби в грі були інші фігури, то це б все ускладнило.

Рисунок 1.29 – Teardown.[20]

Ця гра використовує трасування променів для деяких ефектів, тобто движок рендеринга там насправді гібридний. Ray marching часто використовують в зв'язці з іншими методами рендеринга, в тому числі з растеризуванням.

Цей метод використовується для створення об'ємних ефектів таких як туман, вогонь, дим, або сутінкових променей. Ці ефекти складаються з дуже великої кількості кубів, а це означає, що за допомогою ray marching можливо відтворювати ефекти в реальному часі, нажаль тільки без прораховуваення фізики. Тому що дим напівпрозорий, необхідно шукати не тільки перше зіткнення з кубом, а й подальшими для оцінки густини диму. Це можливо виконувати в реальному часі так як це продуктивний алгоритм.

Це важливі ефекти для досягнення реалістичної картинки. Так як

планета заповнена газами, а вони не прозорі і впливають на промені світла. Саме завдяки об'ємним ефектам графіка в сучасних іграх виглядає дуже реалістично без використання трасування променів.

Рисунок 1.30 – Red Dead Redemption 2.[20]

1.3 Blender

Blender – безкоштовний, професійний пакет для створення 3d графіки, який включає в себе засоби моделювання, анімації, рендеринга, постобробки і монтажу відео зі звуком. Він найпопулярніший з безкоштовних редакторів, тому що програма постійно і швидко вдосконалюється. Великою перевагою пакета Blender є його невеликий розмір у порівнянні з іншими популярними 3d редакторами.[3]

Програма має велику кількість функцій:

- Підтримка геометричних примітивів, полігональних моделей, систему швидкого моделювання в режимі SubSurf, криві Безьє, скульптурне моделювання, векторні шрифти та багато іншого;
- Універсальні вбудовані механізми рендеринга і інтеграція з різними зовнішніми рендерами;
- Професійні інструменти для анімації, наприклад, інверсна кінематика, скелетна анімація, сіткова деформація, ключові кадри, система волосся на

основі частинок, динаміка м'яких і твердих тіл і т.д.
- Базові функції нелінійного монтажу відео;

Інтерфейс користувача має такі особливості:
- Два режиму редагування: «об'єктний режим» і «режим редагування». «Об'єктний режим» в основному служить для маніпуляцій з індивідуальними об'єктами, «режим редагування» для маніпуляцій з фактичними даними об'єкта.
- Широке використання гарячих клавіш, велику кількість команд, що виконуються з клавіатури.
- Особливості в управлінні робочим простором. У програмі є кілька екранів, кожен з яких ділиться на секції і підсекції. Кожен окремий елемент інтерфейсу контролюється тими ж інструментами, що і маніпуляції в 3d просторі. Користувач сам налаштовує інтерфейс під свої потреби і певні завдання.[17]

Пакет Blender має багато додаткових особливостей:
- Всі файли з розширенням .blend є сумісними з усіма версіями Blender;
- Blender створює резервні копії проектів протягом всієї роботи в програмі, це дозволяє не втратити дані під час непередбачених обставин;
- Всі сцени, матеріали, текстури, звуки, ефекти і т.д. можуть бути збережені в один файл з розширенням .blend;
- Також в .blend файл можна зберегти налаштування робочого середовища, що дозволяє користуватися налаштованим під власні потреби інтерфейсом навіть за іншим комп'ютером.[17]

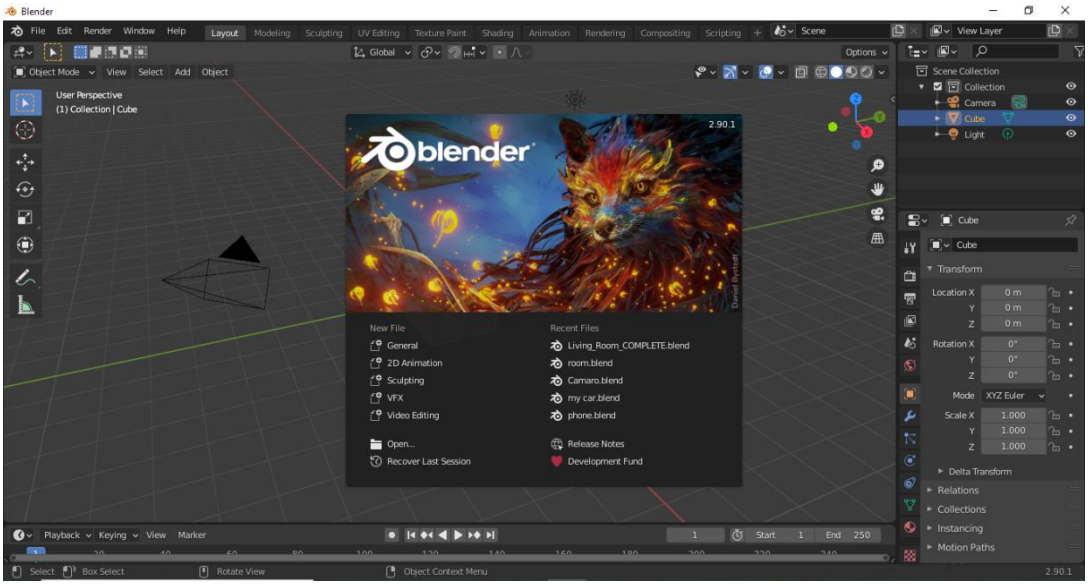


Рисунок 1.31 – Інтерфейс програми Blender

Висновки до розділу 1

В першій частині досліджено види моделювань. При каркасному моделюванні модель, повністю описується в термінах ліній і точок. Полігональний метод передбачає, що об'єкти обмежені поверхнями, які відокремлюють їх від навколишнього середовища. При твердотільному моделюванні модель, визначається в термінах того тривимірного обсягу, який займає визначене нею тіло. Також розглянуто історію прогресу графіки в іграх та технології за допомогою яких він відбувся. Від каркасного рендерингу до сучасних фотореалістичних проектів з використанням трасування промінів. Розглянуто нову технологію воксельної графіки. При використанні воксельної графіки усі об’єкти складаються з велечезної кількості кубів. Наприкінці описано программу для створення графіки , яку буде використанно в данній роботі.

2. АНІМАЦІЯ

2.1 Ріггінг

У 3D анімації ріггінг – це процес підготовки до анімації, що включає створення і розміщення всередині тривимірної моделі рігу (від англ. Rig - оснащення), віртуального «скелета» - набору «кісток» або «суглобів» (bones, joints), встановлення ієрархічної залежності між ними і значень можливих трансформацій для кожної з цих кісток.[7]

Оскільки між ними встановлюється ієрархічна залежність, то зміщення в просторі кожної кістки, що знаходиться в залежності від іншої, буде являти собою сукупність її власних трансформацій і трансформацій, яких зазнає «материнська» кістка. Якісне налаштування залежностей дозволяє аніматорам значно економити зусилля, вказуючи, наприклад, траєкторії зміщення тільки для невеликої кількості окремих кісток, які потягнуть за собою інші, що знаходяться в ієрархічному підпорядкуванні.

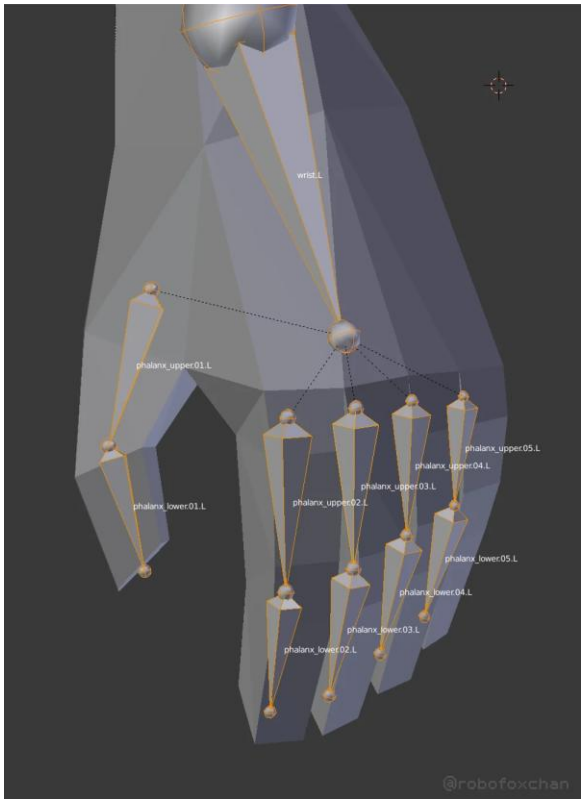


Рисунок 2.1 – Модель руки з рігом.[9]

2.2 Кістки

У Blender «кістка» називається «арматура» відноситься до типу об'єкта, який використовується для деформації сітки. Хоча арматура є об'єктом в Blender, це не об'єкт-сітка. Його форму не можна редагувати, крім збільшення або зменшення. Його можна обертати і переміщувати. У нього є центр, як і у будь-якого іншого об'єкта, який можна зміщувати, але при використання багато кісткової арматури краще ставити центр у вершині нижньої піраміди в центрі сфери. При включеному інструменті маніпуляції інструмент також розташовується по центру.[9]

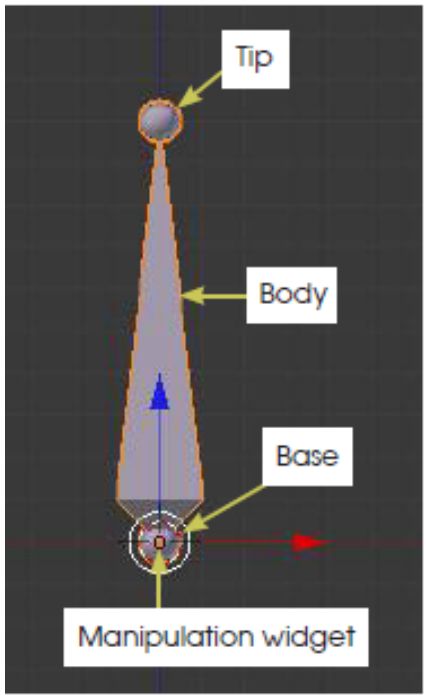


Рисунок 2.2 – Кістка.[9]

За замовчуванням тип відображаємої кісткової арматури восьмигранний, він так називається тому, що він виглядає як дві чотиригранні піраміди, з'єднані в підставі зі сферами на вершинах. З метою демонстрації їх можна називати основа, корпус і наконечник.

Тип відображення арматури за замовчуванням - восьмигранний, але є чотири альтернативних типу відображення: палиця, b-кістка, конверт і дріт.

Варіант відображення провід виглядає так само, як палиця. Який тип відображення використовувати залежить від того, що потрібно зробити з арматурою.[9]

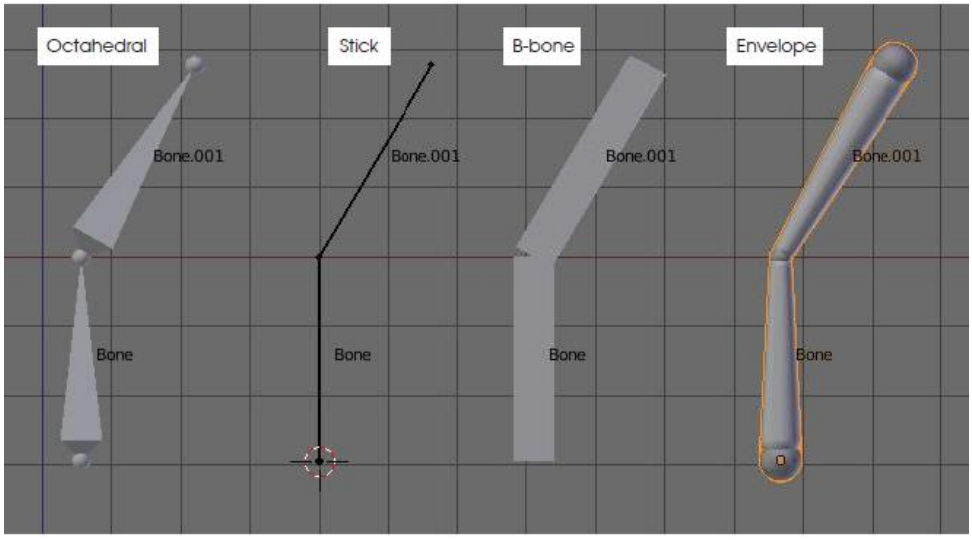


Рисунок 2.3 – Типи відображення арматури.[9]

2.3 Огляд інверсної кінематики

Інверсна кінематика (англ. Inverse kinematics, IK) – процес визначення параметрів пов'язаних гнучких об'єктів для досягнення необхідної позиції, орієнтації і розташування цих об'єктів. Інверсна кінематика активно використовується в робототехніці, тривимірній комп'ютерній анімації і в розробці комп'ютерних ігор. Вона використовується в основному в тих ситуаціях, коли необхідне точне позиціювання гнучких зчленувань одного об'єкта щодо інших об'єктів навколишнього середовища. ІК – це метод, який застосовується до шарнірного тіла. [15]

Шарнірне тіло може являти собою більшість живих тіл, наприклад людей і тварин зі скелетами. Простіше кажучи, зчленовані тіло – це дерево пов'язаних ланцюгів. Пов'язані ланцюги складаються з шарнірів і ланок, де ланка являє собою жорсткий циліндр. [16]

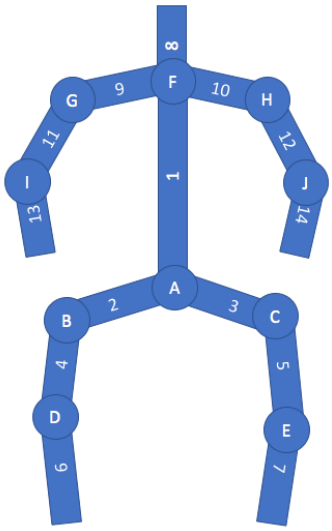


Рисунок 2.4 – Шарнірне тіло людини.[16]

Буває два типи шарнірів. Перший тип шарніра – поворотний. Він підключений до ланки, яка обертається навколо нього. На (Рисунок 36) показано Поворотний шарнір.

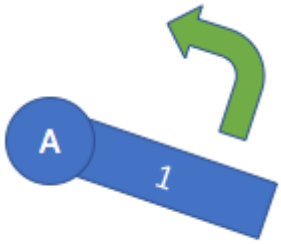


Рисунок 2.5 – Поворотний шарнір.[16]

Поворотний шарнір дуже схожий на центральний штифт годин – з часовою стрілкою в якості ланки.

Призматичний шарнір – це з'єднання, в якому поєднане ланка переміщується від шарніра для подовження / укорочення ланки.

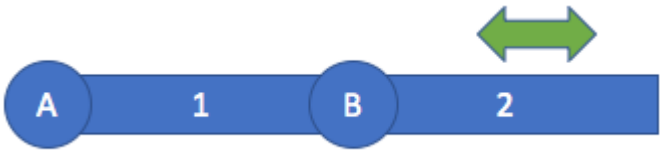


Рисунок 2.6 – Призматичний шарнір.[16]

Шарнірнозчленованою тіло має кореневий суглоб. Кореневий суглоб – це основа конструкції. Кореневий суглоб у гуманоїда зазвичай знаходиться в

центрі стегон. Шарнірно-зчленований корпус утворений деревом суглобів і ланок, починаючи з кореневого суглоба. Нова ланка і зчленування схожі на нову гілку на дереві. Внутрішня ланка / зчленування - це ланка / зчленування, яка знаходиться ближче до кореневого суглоба в деревовидної ієрархії зчленованого тіла, оскільки вона відноситься до даного зчленування / ланки. Зовнішнє ланка / зчленування - це ланка / зчленування, яка знаходиться далі від кореневого зчленування в деревовидної ієрархії зчленованого тіла. На (Рисунок 2.7) з'єднання A є кореневим з'єднанням, ланка 1 – внутрішнєю ланкою з'єднання B, ланка 2 – зовнішня ланка з'єднання B.[16]

Кінцевий ефектор – це положення в самому крайньому положенні самого зовнішньої ланки. Це вільний кінець ланцюга чергующихся зчленувань і ланок. Кінцевий ефектор – це не суглоб. Кінцевий ефектор – це просто стан на кінці шарнірного тіла. Шарнірно-зчленоване тіло може мати кілька кінцевих ефекторів, точно так же, як двійкове дерево може мати кілька листів.

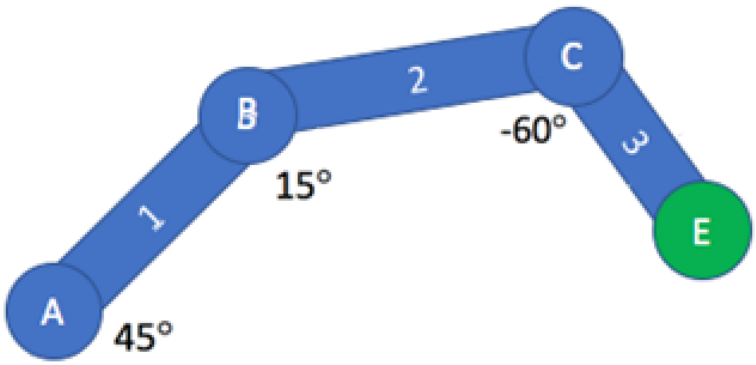


Рисунок 2.7 – Кінцевий ефектор показаний зеленим колом.

Шарнірне з'єднання – це поворот / переміщення з’єднання, яке переміщує пов'язана ланка. Наприклад, на (Рисунок 2.7) [16] суглоби A, B, C, відповідно мають вигини 45 °, 15 °,60 °. Позу можно назвати набором суглобових зчленувань, які позиціонують зчленоване тіло. Поза являє собою векторне значення, а не скалярне.

Для зрозуміння інверсної кінематики, потрібно з'ясувати, що являє

собою пряма кінематика. Сутність прямої задачі кінематики у знаходженні абсолютного положення кожного з суглобів та кінцевої точки (*x*, *y*). Координати будь-якого і-го суглоба можно знайти за формулами:[18]

$$x_i = x_{i-1} + L_i * \cos\left(\sum_{j=0}^{i-1} \theta_j\right), y_i = y_{i-1} + L_i * \cos\left(\sum_{j=0}^{i-1} \theta_j\right) \quad (1)$$

За умови $i = n$, звідци отримано розв'язання прямої задачі.

У процесі руху може бути необхідність отримати кути повороту суглобів за їх координатами. Кут між відрізками *AB* і *BC* можна отримати за формулами:[18]

$$\theta = arccos(L_{AB}^2 + L_{BC}^2 - L_{AC}^2)/(2 * L_{AB}^2 * L_{BC}^2) \quad (2)$$

деL_{AB}, L_{BC}, L_{AC}– довжини відповідних відрізків, визначених за формулою:[18]

$$L_{AB} = \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2} \quad (3)$$

Задача інверсної кінематики полягає у віднаходженні такої конфігурації ланцюга, при якій end-effector буде знаходитись у заданій точці-цілі.

Розв'язанням задачі для ланцюга загального вигляду з n ланками та *n* суглобами є множина кутів суглобів$\theta_i, i = \overline{0, n-1}$. У випадку $n > 2$ може бути необмежена кількість рішень, тому виникає необхідність застосувати такі методи, які забезпечать найбільш природний і поступовий перехід від одного положення до іншого для плавної анімації, та які є найефективнішими і потребують найменше обчислювальних потужностей для застосування методів у режимі реального часу.[18]

Деякі методи можуть не дати розв'язання у випадку, коли точка – ціль є недосяжною, тобто довжина всього ланцюга *L* менша за відстань від кореня до точки цілі (*x*, *y*).

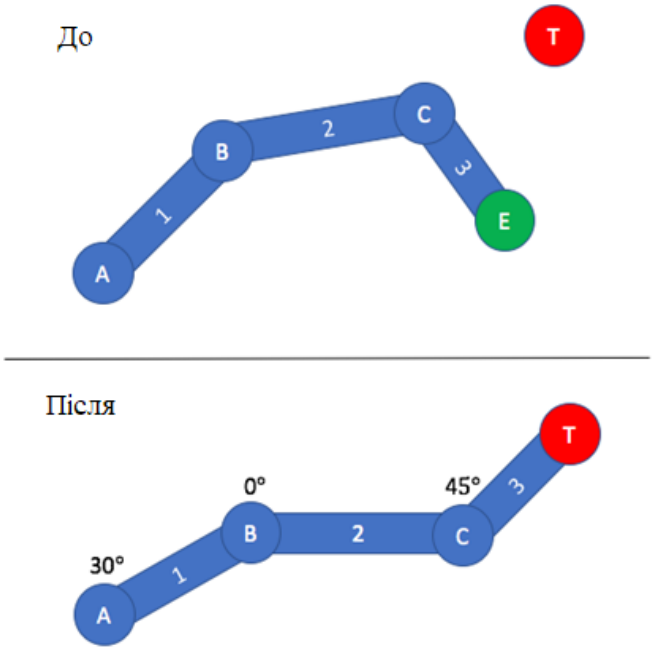


Рисунок 2.8 – Цільова позиція представлена червоним колом. Цільова позиція визначається як вхід, а результуюча поза, необхідна для досягнення кінцевим ефекторів цільової позиції, є виходом.

Інверсійна кінематика – це алгоритм зворотній прямії кінематиці. Алгоритм зворотної кінематики приймає цільову позицію в якості вхідних даних і обчислює на її основі позу, необхідну для досягнення кінцевим ефектором цільової позиції - поза є виходом.

Зворотня кінематика виконує обчислення для знаходження пози. На (Рисунок 2.8) [16] у сцені «До» зображено зчленоване тіло з відомою позою. Цей алгоритм визначає необхідну позицію, яку необхідно досягнути кінцевому ефектору.

Наприклад, коли людина хоче щось взяти рукою, її мозок робить обчислення, щоб знайти необхідну позицію руки і тіла людини. Основна дія це рух рукою, але для виконання цього руху необхідно задіяти суглобів з декількома зчленуваннями, за для того щоб дістатися рукою до необхідного об'єкту. Так працюють і технологічні додатки – задля досягнення бажаної мети, програма виконує обчислення інверсної кінематики для розташування кінцівок. Дуже часто інверсна кінематика необхідна для робототехніки.

Наприклад, оператору робота необхідно перемістити об'єкт за допомогою маніпулятора, але він не хоче окремо керувати кожним зчленуванням маніпулятора.

Часто інверсна кінематика використовується в додатках комп'ютерної графіки та анімації. Аніматору для керування моделлю людиноподібного персонажа, анімація якого генерує комп'ютер, але анімування індивідуальних зчленуваннь, витрачає багато часу. Для цього моделюються віртуальні зчленування і це дозволяє аніматору рухати руки, ноги і тіло ляльки, а комп'ютер автоматично генерує необхідні позиції кінцівок для.

Інверсна кінематика часто використовується в комп'ютерних іграх для створення анімації гуманоїдних персонажів. В основному інверсна кінематика використовується для створення анімації ніг моделей людиноподібної істоти або людини. Наприклад, досить просто створити анімацію пересування (ходьба, біг) людини або наземного тваринного, якщо він рухається по рівній площині. Однак якщо ландшафт нерівний, то створення точної анімації ходьби є фактично неможливим завданням. Анімація ніг не буде відповідати рельєфу поверхні, що проявиться в таких ефекти, як прослизання ніг по поверхні і неточному позиціонуванні ніг щодо її (ступня буде «тонути» в поверхні або «не діставати» до неї). Саме для якісного та ефективного вирішення цих проблем використовується інверсна кінематика.

Висновки до розділу 2

В другої частині дипломної роботи описанні засади анімації 3D моделей та їх підготовки до анімації. Перше що необхідне для анімації це ріг , тобто скелет який складається з кісток. Кістки розташованні в скелетті за ієрархічною структурою. Описано інверсійну кінематику, яка найбільш підходить для антропоморфних моделей та роботів. Матеріалів досліджених в перших двух , цілком достатньо для створення власної анімованої моделі.

3. ПРОГРАМНА РЕАЛІЗАЦІЯ

3.1 Створення 3D моделі засобами Blender

Пальці захвату створенні з двох кубів та циліндру, за допомогою операцій extrude, scale та move. Потім були використані модифікатори bevel та subdivision surface. Модифікатор bevel використано для створення фаски на гранях елементу, грані до яких буде примінено модифікатор у вьюпорті позначаються синім. Subdivision surface створює додаткову геометрію, яка необхідна для кращого відбиття світла.

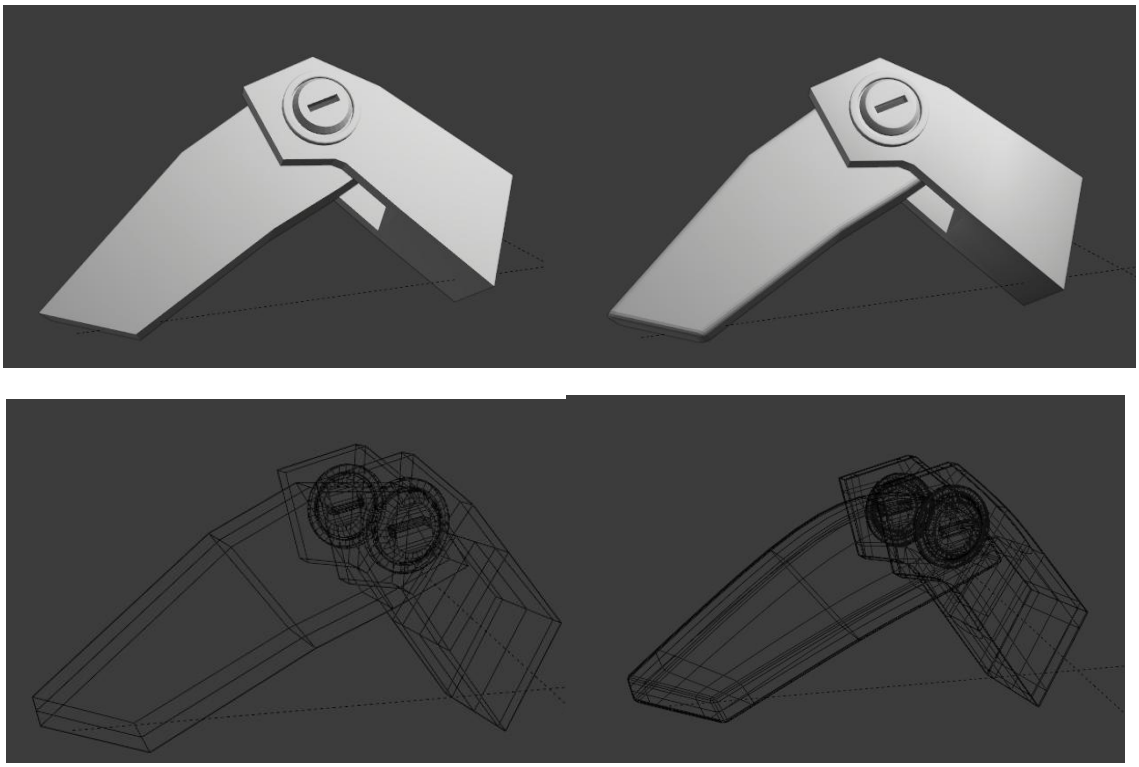


Рисунок 3.1 – Пальці захвату до та після модифікаторів та їх сітка

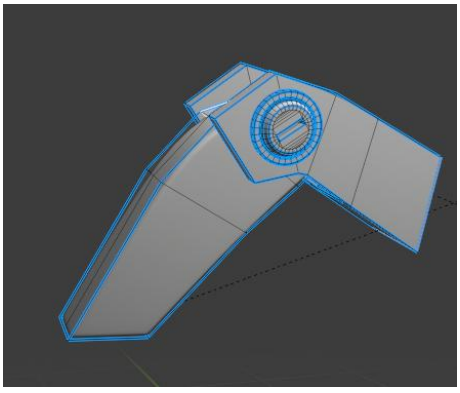


Рисунок 3.2 – Позначення модифікатора bevel

Для створення захисного кожуха було використане коло якому надано необхідну форму.

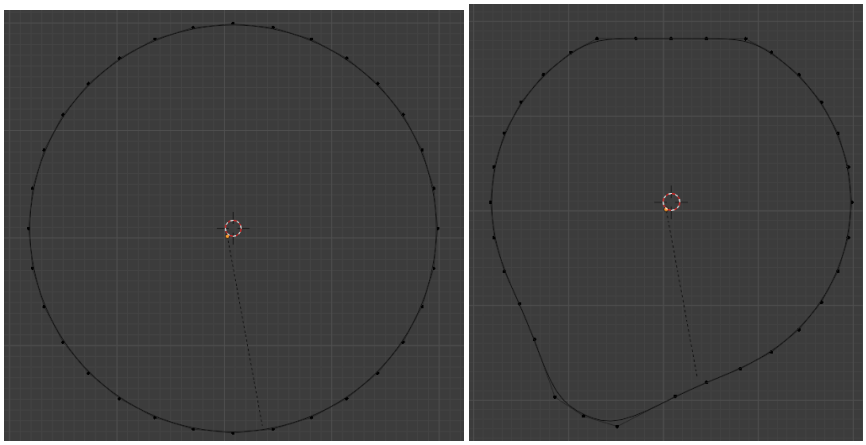


Рисунок 3.3 – Коло до і після зміни форми

Далі отриману форму було екструдовано і звуженно в необхідних місцях.

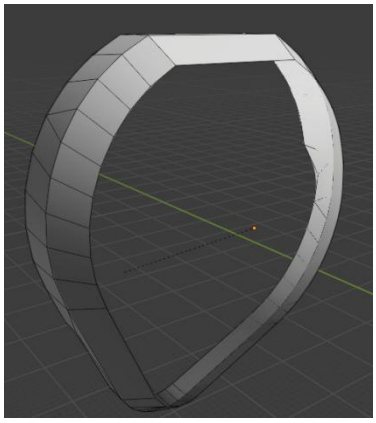
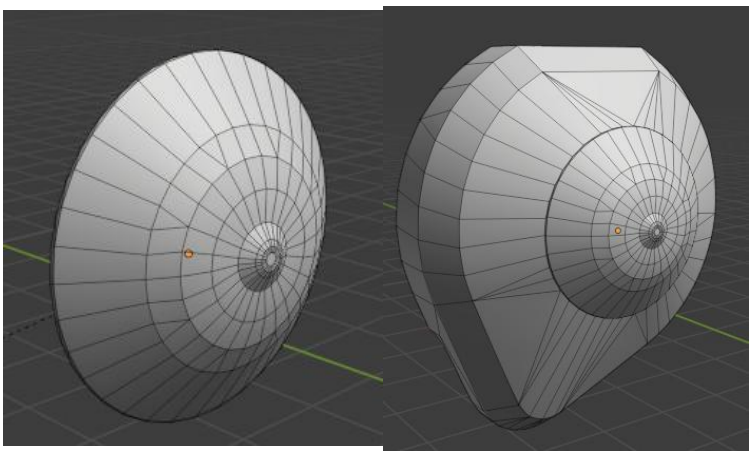


Рисунок 3.4 – Деталь після пр

Далі на основі нового кола було створенно другу частини деталі, вона була об’єднана задопомогою операції brige edge loops. На прикінці до моделі також були додані модифікатори bevel та subdivision surface.



(а) (б)

Рисунок 3.5 – (а) друга частина моделі, (б) фінальний вигляд після об'єднання.

Створення основи також було виконано на основі базового елемента куба. Отвори було зроблено за допомогою булевих оперпцій. Було створено циліндр та надано йому необхідної форми. Потім геометрію цього циліндру було віднято від основної частини за допомогою модифікатора Boolean.

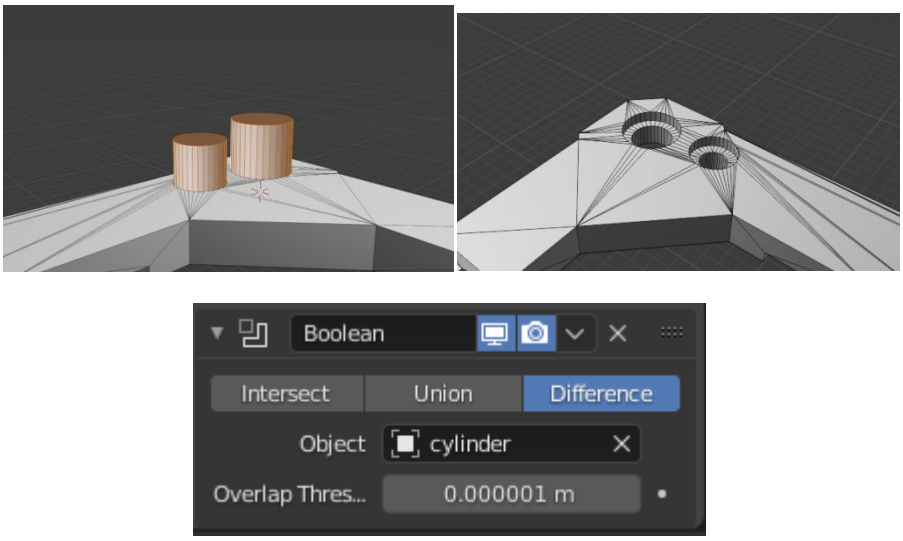


Рисунок 3.6– Створення отворів за допомогою модифікатора Boolean.

Після створення отворів основу було дзеркально відображенно по осі Y. Цю операцію виконано за допомогою модифікаторп Mirror.

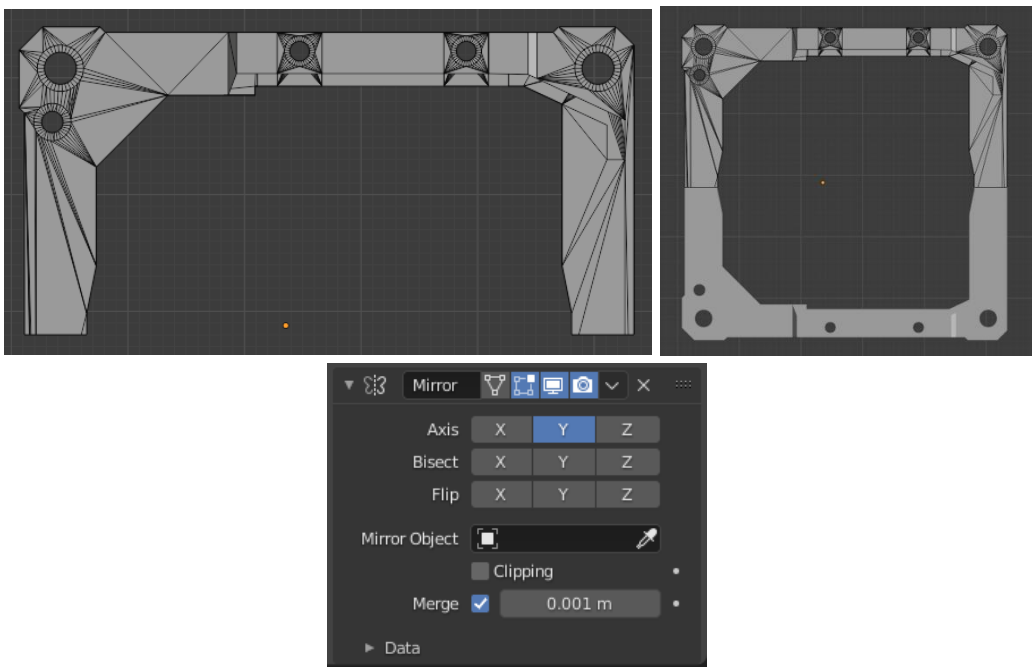


Рисунок 3.7 – Віддзеркалення за допомогою модифікатора Mirror.

Усі інші деталі було виконано за допомогою використання та комбінування наведених вище методів.

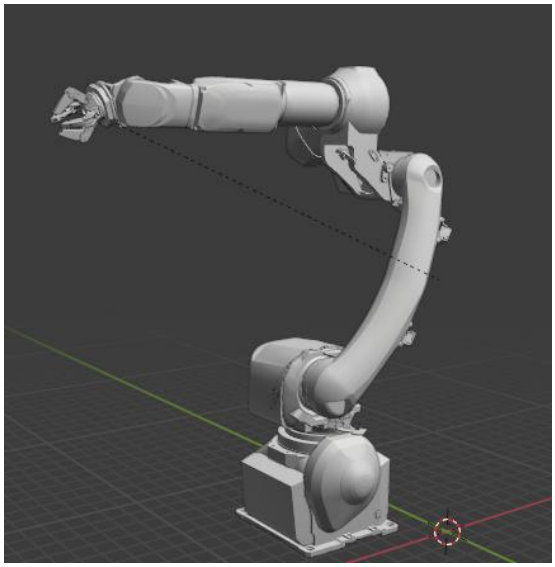


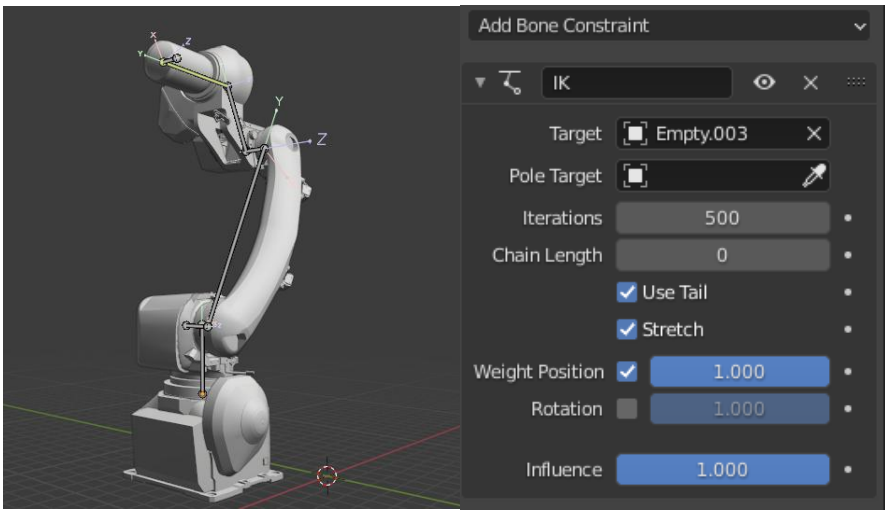
Рисунок 3.8 – Фінальний вигляд моделі.

3.2 Створення ріга

Ріггінг створеної моделі складається з трьох частин:

- Ріг основної частини
- Ріг обертальної частини руки
- Ріг захвату

Було створенно систему кісток, які було прив’язано до рухомих частин моделі. На кінці останьої кістки створенно об'єкт Empty.003, який видно у вьюпорті, але не видно при рендерінгу. Відносно цього об’єкту програма буде прораховувати інверсну кінематику.



(а) (б)

Рисунок 3.9 – (а) скелет основної частини, (б) додання інверсної кінематики.

У всіх кісток заблоковані осі, окрім потрібної, кістки мають свою локальну систему координат, яка не завжди співпадає з глобальною. Також були обмеженні мінімальний та максимальні кути обертання, синя дуга на Рисунок49.

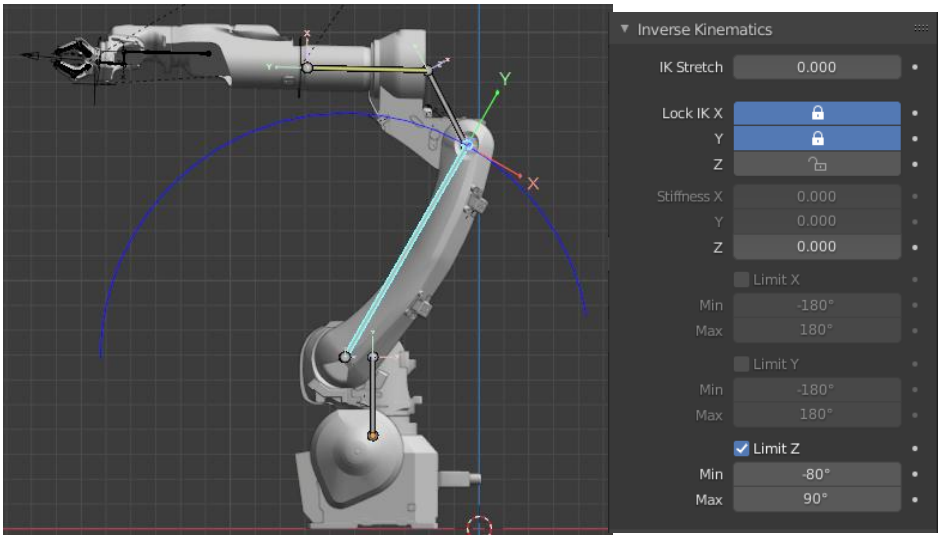
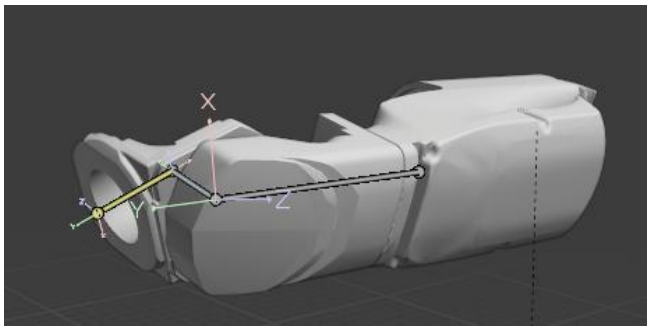
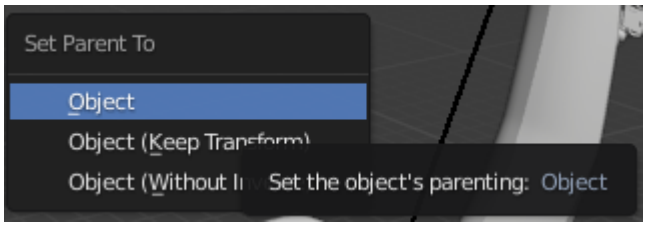


Рисунок 3.10 – Блокування і ліміт осей

Ріггінг обертальної частини виконано аналогічно рігу основної. Об’єкт Empty.002 прив’язано до об’єкту Empty.003 попередньої частини, тому при русі Empty.003 рухається також і обертальна частина



(а)



(б)

Рисунок 3.11 – (а) скелет обертальної частини,(б) прив’язка об’єктів.

Ріггінг всіх чотирьох «пальців» захвату зроблений за одним шаблоном. Кожна фаланга пальцю прив’язана до своєї кістки. Об’єкти Empty кожного з пальців прив’язано до стрілки, яка керує відкриттям та закриттям всього захвату. Цю частину прив’язано до обертальної частини.

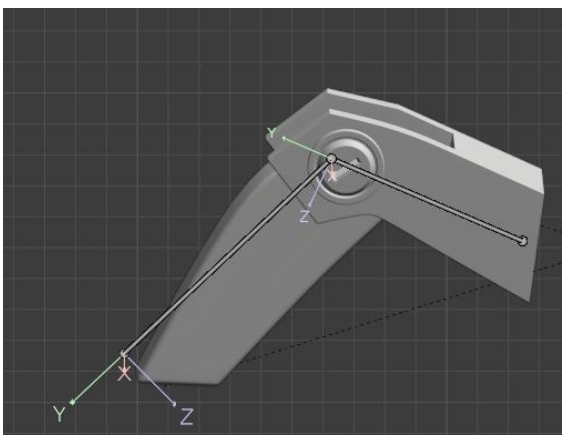


Рисунок 3.11 – Ріг одного з «пальців»

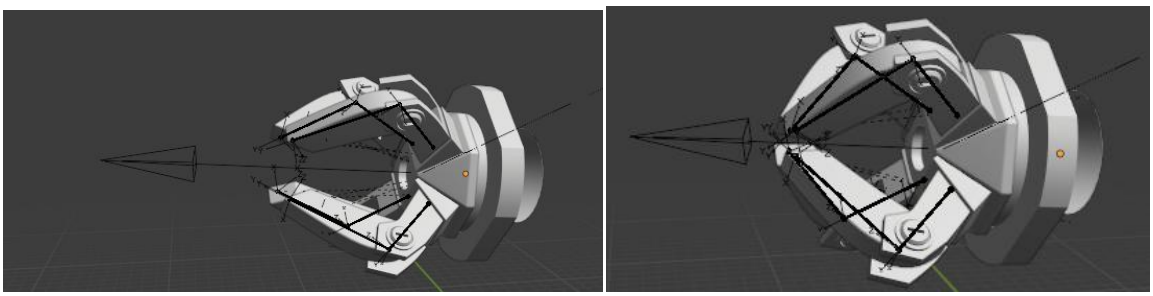


Рисунок 3.12 – Відкрите, та закрите положення захвату.

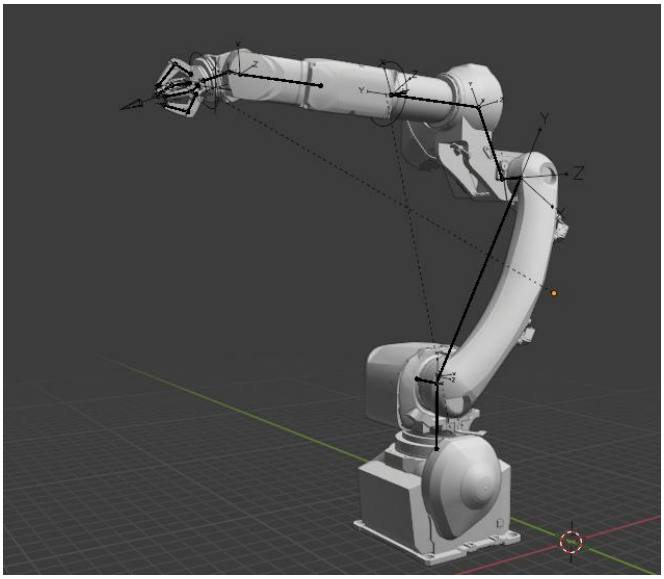


Рисунок 3.13 – Фінальний вигляд рігу.

Рисунок 3.14 – Фінальний рендер.

Висновки до розділу 3

За допомогою програмного забезпечення Blender створено модель, зроблено ріггінг та анімовано на основі інверсійної кінематики.

ВИСНОВКИ

Комп'ютерна графіка - незамінний інструмент для багатьох сфер життя суспільства: реклама, кіно, мультиплікація, ігри. Тривимірна графіка еволюціонувала від специфічного інструменту вузького кола людей до одного з найпопулярніших і затребуваних інструментів в медіаіндустрії. Тривимірна графіка має величезну значимість для мультимедіа, дозволяє втілити в життя найсміливіші мрії і проекти. 3D графіка дозволяє створити реалістичне зображення з усіма тінями, відблисками, відтінками

В ході виконання дипломної роботи надано детальний опис прийомів і методів роботи в середовищі тривимірного моделювання Blender.

Створена тривимірна модель робота-руки з ріггінгом для анімації. Прототипом даної моделі став зварювальний робот компанії Fanuc. Отриманий результат може використовуватися в сфері анімації в кіно, мультиплікації або в ігровій індустрії.

Досліджено види моделювань. При каркасному моделюванні модель, повністю описується в термінах ліній і точок. Полігональний метод передбачає, що об'єкти обмежені поверхнями, які відокремлюють їх від навколишнього середовища. При твердотільному моделюванні модель, визначається в термінах того тривимірного обсягу, який займає визначене нею тіло. А також їх поєднання.

Досліджено технології для відтворення зображень, а саме – трасування промінів та растеризації алгоритми на яких вони базуються та особливості їх використання, проблеми при їх використанні – їх вирішення чи оптимізація. Алгоритм z-буферизації, що дозволяє фільтрувати порядок відтворення об’єктів.

Проаналізовано професійний пакет для створення 3d графіки Blender, який включає в себе засоби моделювання, анімації, рендеринга, постобробки і монтажу відео зі звуком. Це один з найпопулярніших безкоштовних редакторів, тому що програма постійно і швидко вдосконалюється.

Досліджено інструменти для створення анімації. Найважливішим є

алгоритм інверсної кінематики, який дозволяє полегшити процес створення анімації. Це один із найкращих алгоритмів для анімації людино подібних створінь і робототехніки тому часто використовується для інженерної візуалізації.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Wang L.T., Chen C.C. A Combined Optimization Method for Solving the Inverse Kinematics Problem of Mechanical Manipulators. *IEEE Transactions On Robotics and Automation*. V.7. 1991. P. 489 – 498.
2. David J. Eck. Introduction to Computer Graphics. Geneva, NY, 2018.432 с.
3. Blender Developers Blog [Електронний ресурс] – Режим доступу: https://code.blender.org/ Назва з екрана.
4. Computer graphics [Електронний ресурс] – Режим доступу: https://www.newworldencyclopedia.org/entry/Computer_graphics Назва з екрана.
5. Демин А.Ю. Основы компьютерной графики: Учебное пособие– Томск: Изд-во Томского политехнического университета, 2011. 191 с.
6. Конспект лекций по дисциплине «Компьютерная графика» для студентов специальности 230102 «Автоматизированные системы обработки информации и управления» – Москва 2009. 249 с.
7. Death to the Armatures: Constraint-Based Rigging in Blender. Kuhn Industries 2013, 361с.
8. *Modeling and Animation Using Blender: Blender 2.80: The Rise of Eevee* Ezra Thess Mendoza Guevarra Laguna, Philippines, 2020. 302 с.
9. The Complete Guide to Blender Graphics Computer Modeling and Animation John M. Blain Publisher : A K Peters/CRC Press; 1st edition 2012, 390 с.
10. Blender for Animation and Film-Based Production 1st Editionby Michelangelo Manrique Publisher : A K Peters/CRC Press; 1st edition 2014, 284 с.
11. Backward Ray Tracing James Arvo Apollo Computer, Inc Chelmsford 1986, 8 с.
12. Методические рекомендации к лабораторным работамдля студентов специальности 1-53 01 02 «Автоматизированныесистемы обработки

информации» и направлениям подготовки09.03.01 «Информатика и вычислительная техника»,09.03.04 «Программная инженерия» дневной и заочной форм обучения Могилев 2018, 38 с.

13. Рендеринг.URL: https://uk.wikipedia.org/wiki/Рендеринг (дата звернення: 14.03.2021).
14. Козырев Ю. Г. Промышленные роботы: основные типы и технические характеристики – М.: КНОРУС. 2017, 560 с.
15. Інверсійна кінематика URL: https://ru.wikipedia.org/wiki/Инверсная_кинематика (дата звернення: 24.04.2021).
16. Overview of Jacobian IK URL: https://medium.com/unity3danimation/overview-of-jacobian-ik-a33939639ab2 (дата звернення: 24.04.2021).
17. Blender URL: https://wikizero.com/uk/Blender (дата звернення: 25.03.2021).
18. Кинематика: прямая и обратная задачи. URL: http://robocraft.ru/blog/mechanics/ 756.html (дата звернення: 04.04.2021).
19. Types of 3D Modeling: Which Is Best for Your Needs?Picture of Leon Ortizby Leon Ortiz URL: https://all3dp.com/2/types-of-3d-modeling/ (дата звернення: 04.03.2021).
20. КАК НА САМОМ ДЕЛЕ РАБОТАЕТ 3D ГРАФИКА URL: https://media-xyz.com/ru/articles/1270-kak-na-samom-dele-rabotaet-3d-grafika?tg_rhash=67d1bbd3e3de58 (дата звернення: 01.03.2021).

ДОДАТОК А

SUMMARY

SUMMARY

Nowadays, the importance of computer graphics is difficult to overestimate. The world is not standing still, Rapid pace of technological progress Make computer graphics in demand in many areas of industry. 3D graphics is an integral companion of architects, designers, cultural figures, advertising professionals, experts in the gaming industry and difficult engineering. The possibilities of 3D graphics in today's world are almost limitless.

Quite often the phenomenon is the creation of an advertising campaign for the product before it was born. In this case, the only way out is! Application of computer graphics. with the right approach to the process of creating a 3D model and animation, it is very difficult for a person to determine that in front of him the image is not a real product, but his computer model. Technology is evolving very fast, so no more consuming use COMPLEX scenery or props that are impossible to get - the computer will easily solve this problem.

The basic concept in 3D graphics is a 3D model. Qualitatively Creating a model is the basis of any project, because if it is made of poor quality, then no texture, no animation, no visualization - does not correct the situation significantly. Another concept considered in this paper is rigging. It simplifies the manipulation of a large number of objects, which greatly facilitates the work of the animator.

After modeling, only a mathematical model is obtained, with information about the geometric shape of the object. Textures are used to give the model the right color.

A texture is a two-dimensional image that is superimposed on a three-dimensional model. Procedural textures (generated by the algorithm) and creation in the graphic editor are used. The texture sets only the color and pattern of the model, and the ability to reflect, refract light, have relief and transparency are set in the graphics editor in the properties of the material. That is, the material is a mathematical model that describes the parameters given to the surface.

Before applying the texture to the model, you need to create a UV scan - that is, imagine how the surface of the model will be projected on the plane. The letters

U and V denote the coordinate axes of the scan plane, because the letters X, Y and Z are used to denote the spatial coordinates.

Creating a three-dimensional model can be divided into the following stages:

- Search for references or drawings from which the model will be created.
- Modeling of the object on the basis of a reference or drawing.
- UV scan.
- Create textures.
- Adjustment of the material to its physical properties

After passing these stages, the model is ready for visualization or you can work with it:

- Adjust the horn of the model.
- Animation.

In 3D graphics, after making the model to visualize it, you need to place it in the stage, add a camera and lighting, after which you can get the final render. To calculate the render, a physical model of light beam propagation is used, taking into account reflection, scattering, glare, refraction, and so on. In traditional painting, the artist himself creates light effects, and in 3D graphics, the visualizer himself paints the final image, and the author creates a scene with geometry, lighting materials, camera properties.

The peculiarity of the frame model is that its description uses geometric objects of the first order - lines and edges.

Disadvantages of the frame model:

- The need to submit all the edges in order to depict the model;
- Inability to recognize curvilinear faces;
- Errors in calculating the physical characteristics of the suit;
- Inability to create tonal images due to lack of face information.
- Lack of information on how objects are arranged relative to each other;

However, it cannot be said that this method of modeling is completely unfounded. The frame model consumes less computer resources and is therefore

better suited for simple tasks. This method is often used not so much in modeling, but as one of the methods of visualization in the display of finished models.

The second method is surface modeling, this method is determined not only by points and lines, but also by surfaces. In this method, objects are limited to the surfaces by which they are separated from the environment. The surface of the object is limited by contours that are the result of the intersection of two surfaces.

Advantages of surface modeling over frame:

- Ability to identify complex curvilinear faces;

- Ability to receive tone images;

- Ability to recognize special constructions on surfaces, for example, openings;

- Ability to obtain a quality image.

The third method is solid. The solid model is defined in the three-dimensional volume occupied by the body it defines. Solid state modeling can be called - the most perfect and reliable method for reproducing a real object. Advantages of this method:

- Internal and external areas of the object demarcation;

- Hidden lines are automatically deleted;

- Three-dimensional sections of the object are built automatically, this is important for complex models;

- The method uses analysis that automatically obtains images with specific characteristics;

- Ability to receive tone images;

There are two classes of methods for creating solid-state models:

- Method of constructive representation;

- The method of boundary representation.

When using the method of constructive representation, the solid model is created from the basic elements (primitives, which are determined by size, shape, orientation and anchor point).

Boolean operations are used to create solid-state models:

- Association;
- The difference;
- Intersection.

This method has its pros and cons. The method of constructive representation allows you to calculate the physical properties of the object with high accuracy.

The second method is the method of boundary representation. This method describes the boundaries of the body and sets the boundaries that describe the object. This is just one method that allows you to create an accurate idea of a geometric solid. When using this method, you must specify the contours or boundaries of the object and the different types of objects, as well as specify the lines connecting the types to establish mutual correspondence.

Each of the methods has its advantages and disadvantages. The method of constructive representation is more convenient at initial construction of model; also, this method will take up less space in the database. Unfortunately, the method of constructive representation is not very good for constructing complex forms. Also, although in the constructive method the data take up much less space, the number of calculations to reproduce the model is greater.

Before applying the texture to the model, you need to create a UV scan - that is, imagine how the surface of the model will be projected on the plane. The letters U and V denote the coordinate axes of the scan plane, because the letters X, Y and Z are used to denote the spatial coordinates.

The main factors in the progress of graphics technology are games and cinema. When technology did not allow 3D games to be made, ways to simulate 3D were invented. The first games used wireframe rendering. Objects in space are represented as points. Some pairs of points are connected by lines. To display such a scene, you need to project the points that are in line of sight on the screen, and then just draw the lines between them.

The first more realistic simulation of 3D space was implemented in the game Wolfenstein 3D from idSoftware in 1992. They first implemented a visualization method in the game called ray casting.

In the ray casting method, the scene is represented as an image on which the walls are marked. In this scene, the position of the camera and its viewing angle are determined, and then the rays are let out of the camera and for each of them a point of contact is sought. The number of rays is equal to the number of pixels of the monitor in width.

Next, for each beam, a vertical segment of the wall is drawn, the size of which depends on how far from the camera is the point of contact. After painting all the vertical segments, you get an image of the walls with an imitation of a linear perspective.

Additionally, you can paint the walls in a certain color or apply a texture to them. To do this on the map of the walls, the walls are marked with certain colors, each color is associated with a certain texture. The playback technique does not change.

. When drawing vertical segments of walls the corresponding segment from a texture will be taken.

This method is quite limited. Only the walls are three-dimensional, and the rest of the objects are realized in the form of sprites, which are constantly directed to the camera.

The principle of rasterization is quite simple. All objects consist of triangles, they are polygons. Why triangles? They consist of a minimum number of vertices that can describe a plane. Accordingly, they cannot be convex or concave and are therefore a minimal structural unit.

To draw an object, we need to draw each polygon of which it consists. To do this, all vertices are projected on the plane of the screen. Then for each polygon pixels between its vertices are painted over. The whole process is depicted in the pictures. Rasterization is often called scan transformation, because many

rasterization algorithms use incremental methods, scanning the figure along the lines (lines) of the raster, exploiting the property of coherence.

Incremental methods calculate new values quickly based on the values obtained in the previous step, instead of calculating new values directly, which is often a rather slow process. Coherence in space or time is a term that means that close objects (ie, pixel strings in our case) have qualitative characteristics similar to those of the current object.

For optimization, polygons that overlap with other polygons do not participate in the display. In this case, you need to understand whether to draw new pixels on top of the old ones or skip them. The z-buffer is used for this purpose. Using an additional array, in which each pixel in the raster image is assigned a coordinate Z. The coordinate Z corresponds to the distance at which the point of the object from the plane on which it is projected Z axis is perpendicular to the plane of design.

When displayed, the pixel depth is compared with each other and priority is given to those pixels that are closer to the screen. The figure shows how the z-buffer works. The pink triangle closest to the camera has a value of 0.1, more distant objects have larger values and are drawn last.

If two objects have a close Z-coordinate, sometimes, depending on the point of view, one, then the other, then both are shown in a striped pattern. This is called Z-conflict. Conflicts are often characterized by special effects (decals), superimposed on the main texture, such as bullet holes.

Z-conflicts are resolved by shifting one object relative to another by an amount greater than the Z-buffer error.

One of the problems is the difficulty of creating realistic lighting. To shade the objects for each landfill, the angle at which the sun's rays are applied to it is calculated, ie the angle between the normal of the landfill and the sun's rays. It determines the brightness of the polygon and, accordingly, the color of the pixels. But with this approach, you can see clear boundaries of landfills. To get rid of them, the angles between the normals are interpolated.

Another problem is the reflection of shadows. The only thing that can be done in rasterization - is to simulate shadows. The most common way to display shadows is to render the scene from the position of the light source. Then we can get data on which parts of the scene are in the shadows and reduce the brightness of the corresponding pixels.

But this method is not accurate enough and sometimes the shadows are shifted relative to the objects. This classic flaw has been amusingly called "Peter Panning". But the shadow is not realistic at all. In life, the edges of the shadows blur as you move away from objects, and here the shadow is absolutely stiff. It can only be blurred around the perimeter.

In fact, soft shadows developers have already learned to do. But the algorithms for rendering such shadows are quite complex and resource intensive.

Also, rasterization lighting suffers from the inability to implement global illumination. The fact is that the rays of light are constantly reflected and illuminate places where light does not fall directly. Therefore, the shadows are never completely black. In the image below you can see how huge the difference in lighting can be, if you do not take into account global lighting.

Unable to create valid mappings. There is one hack that helps to partially solve this problem - it is the creation of cubic maps. A cube map is an image that stores space around an object. With their help you can calculate what should be displayed at a certain point. But they are not suitable for dynamic scenes where objects or light sources can move because the objects will only reflect what is stored in the cubic map.

Ray Tracing can now be considered the most powerful and versatile method of creating photorealistic images. There are many methods used to implement tracing algorithms to display the most complex three-dimensional scenes. The versatility of tracing methods is due to the fact that they are based on simple ones.

concepts based on human experience of perception of the world around us. The principle of ray tracing lies in the imitation of sunlight. When a beam of light

hits the surface, it can reflect in any direction. This is because even smooth objects have microscopic irregularities. If the object is really smooth, then we get a mirror material. In this case, all the rays are reflected in approximately one direction.

The number of additional rays is limited, so the assessment of the impact of surrounding objects on the starting point is not very accurate. Because of this, the illumination in neighboring points may differ even if it must be the same. Due to this, there is noise in the images created using ray tracing. An increased number of rays is required to eliminate noise, but this significantly increases the rendering time. It is easier to use denoisers.

This is a very simple method of visualization.

1. For a pixel on the screen, the distance to the nearest object is determined.

2. This distance determines the radius at which we can let our beam.

3. For the end point of the beam, points 1 and 2 are repeated until the next radius becomes small enough, which will mean a collision with the object. Alternatively, the radius may begin to increase steadily if the beam has passed all objects.

Performing this process will obtain scene images for all pixels. For pixels that reflect a collision with an object, you can even implement the simplest shading based on the angle at which the light falls. But why do you need this method at all, when there are others? As you can see, the main action that takes place in it - is to find the distance to the object. Usually, to find the minimum distance, we need to find the distance to each polygon that makes up the object and choose a smaller one. But this requires a lot of resources, so ray marching is used to render objects for which there is a signed distance function.

The distance function allows us to calculate the distance to an object using information about its position in space. And we do not need to consider the distance to each landfill. In fact, polygons in ray marching are not even used. If we need to display a sphere, we simply specify its position and radius, and that will be enough to represent it.