МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ХАРКІВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ ІМЕНІ В.Н.КАРАЗІНА
НАВЧАЛЬНО-НАУКОВИЙ ІНСТИТУТ КОМП'ЮТЕРНОЇ ФІЗИКИ ТА ЕНЕРГЕТИКИ
Кафедра комп'ютерної фізики

# ПОЯСНЮВАЛЬНА ЗАПИСКА

до дипломної роботи
за освітньо-кваліфікаційним рівнем магістр

"Точкові імпостори для візуалізації в режимі реального часу"

Виконав: студент 2 курсу, групи НФ-64
Чан Куанг Олена
спеціальність:
105 Прикладна фізика та наноматеріали
освітня програма: Прикладна фізика
енергетичних систем
Керівник:
к.ф-м.н. Лісіна О.Ю
Рецензент: Подоляка О.О.

Навчально-науковий інститут комп'ютерної фізики та енергетики
Кафедра комп'ютерної фізики
Рівень вищої освіти (освітньо-кваліфікаційний рівень) магістр
Спеціальність 105 Прикладна фізика та наноматеріали

**ЗАТВЕРДЖУЮ**

**Завідувач кафедри**

_____ Костянтин НЄМЧЕНКО
підпис

"____"_____2020 року

# З А В Д А Н Н Я
## НА ДИПЛОМНУ РОБОТУ (ПРОЕКТ)

Чан Куанг Олена

1.Тема роботи «Точкові імпостори для візуалізації в режимі реального часу» керівник роботи: Лісіна Ольга Юлієвна, кандидат фізико-математичних наук.

затверджені наказом по університету від "10" квітня 2020 року №0210-05/513

2. Строк подання студентом роботи 12.05.2020

3. Перелік питань, які потрібно розробити:

1) Провести аналіз методів аналізу точкових 3D-моделей;

2) Ознайомитися із методом інтегрування Монте-Карло;

3) Вивчити алгоритм спрощення метричної геометрії для клітинок перегляду

4) Розробити програмне забезпечення для заміни складної точкової моделі імпостором.

5) Провести тестування розробленого програмного забезпечення.

4. План роботи

| № з/п | Назви етапів роботи |
|---|---|
| 1 | Вибір теми, узгодження, а також затвердження її на кафедрі |
| 2 | Робота над літературою та іншими джерелами |
| 3 | Затвердження теми на кафедрі |
| 4 | Написання вступу до роботи та першого розділу |
| 5 | Написання другого розділу |
| 6 | Написання третього розділу |
| 7 | Завершення написання дипломної роботи (висновки та література) |
| 8 | Подача науковому керівнику |
| 9 | Відгук, рецензія, підготовка до захисту |
| 10 | Попередній захист |
| 11 | Захист |

5.Дата видачі завдання

Студент _____ Чан Куанг Олена
підпис

Керівник роботи _____ Лісіна Ольга Юліївна
підпис

# РЕФЕРАТ

Чан Куанг О. Точкові імпостори для візуалізації в режимі реального часу.

Дипломна робота представлена на 41 сторінках; складається з вступу, 3 розділів та висновків; містить 12 рисунків, 1 таблиця, список використаної літератури з 35 найменувань.

Мета даної роботи – створити результуюче уявлення яке є компактним, швидко рендерится з використанням існуючого графічного обладнання, створює високоякісне зображення з невеликим накладенням, не має відсутніх або зникаючих об'єктів при переміщенні точки огляду і правильно представляє залежні від виду зміни в клітинки перегляду.

Об’єкт дослідження – вибіркове уявлення простору об'єктів, фотореалістичний рендеринг.

Предмет дослідження – демонстрація методу застосування у міській візуалізації, модифікація точних методів, з метою зменшення часу рендеринга на один кадр, і відображення їх на архітектурі.

Ми представляємо нову структуру даних для кодування зовнішнього вигляду геометричної моделі, що видно з області перегляду (клітинок перегляду). Це представлення може використовуватися в інтерактивних програмах для візуалізації в режимі реального часу для заміни складної моделі імпостором, підтримуючи високу якість візуалізації, скорочуючи час візуалізації. По-перше, швидкість вибірки контролюється, щоб забезпечити достатню щільність для всіх можливих умов перегляду із зазначеної клітинки перегляду. По-друге, правильне, антиалійне представлення плеоптичної функції обчислюється за допомогою інтегрування методом Монте-Карло. КЛЮЧОВІ СЛОВА: РЕНДЕРИНГ, КЛІТИНОК ПЕРЕГЛЯДУ, МІСЬКА ВІЗУАЛІЗАЦІЯ, МОНТЕ-КАРЛО.

# РЕФЕРАТ

Чан Куанг Е. Точечные импосторы для визуализации в режиме реального времени.

Дипломная работа представлена на 41 страницах; состоит из введения, 3 глав и выводов; содержит 12 рисунков, 1 таблицы, список использованной литературы из 35 наименований.

Цель данной работы - создать результирующее представление которое является компактным, быстро рендерится с использованием существующего графического оборудования, создает высококачественное изображение с небольшим наложением, не имеет отсутствующих или исчезающих объектов при перемещении точки обзора и правильно представляет зависящие от вида изменения в ячейке вида.

Объект исследования - выборочное представление пространства объектов, фотореалистичный рендеринг.

Предмет исследования - Демонстрация метода применения в городской визуализации, модификация точных методов, с целью уменьшения времени рендеринга на один кадр, и отображения их на архитектуре.

Мы представляем новую структуру данных для кодирования внешнего вида геометрической модели, видимой из области просмотра (ячейка просмотра).Это представление может использоваться в интерактивных программах для визуализации в режиме реального времени для замены сложной модели импостором, поддерживая высокое качество визуализации, сокращая время визуализации. Во-первых, частота дискретизации регулируется для обеспечения достаточной плотности во всех возможных условиях просмотра из указанной ячейки просмотра. Во-вторых, правильное сглаженное представление плеоптической

функции вычисляется с использованием интегрирования методом Монте-Карло. Поэтому наша система обеспечивает высокое качество рендеринга, используя простое представление с ограниченной сложностью.

КЛЮЧЕВЫЕ СЛОВА: РЕНДЕРИНГ, ЯЧЕЙКА ВИДА, ГОРОДСКАЯ ВИЗУАЛИЗАЦИЯ, МОНТЕ-КАРЛО.

## ABSTRACT

Chan Kuang O. Point-Based Impostors for Real-Time Visualization.

The diploma project is presented on 41 pages; consists of introduction, 3 chapters and conclusions; contains 12 figures, 1 table, and a list of used literature of 35 items.

The purpose of this work is to create a resulting view that is compact, quickly rendered using existing graphics equipment, creates a high-quality image with a small overlay, has no missing or disappearing objects when moving the viewpoint, and correctly represents the type-dependent changes in the view cell.

The object of study a selective representation of the space of objects, photorealistic rendering.

The subject of the study demonstration of the method of application in urban visualization, modification of the exact methods, in order to reduce the rendering time by one frame, and display them on the architecture.

We present a new data structure for encoding the appearance of a geometric model as seen from a viewing region (view cell). This representation can be used in interactive or real-time visualization applications to replace a complex model by an impostor, maintaining high quality rendering while cutting down rendering time. First, the sampling rate is controlled to provide sufficient density across all possible viewing conditions from the specified view cell. Second, a correct, antialiased representation of the plenoptic function is computed using Monte Carlo integration. Our system therefore achieves high quality rendering using a simple representation with bounded complexity.



KEYWORDS: RENDERING, VIEW CELL, URBAN VISUALIZATION, MONTE CARLO

**ЗМІСТ**

ВСТУП……...9

1. ОГЛЯД ІСНУЮЧИХ АЛГОРИТМІВ……11

1.1 Алгоритм та деталі реалізації……14

2. ТОЧКОВІ ІМПОСТОРИ……...19

2.1 Складність появи……..……..19

2.2 Геометрична вибірка……...20

2.3 Функція пленоптичного зображення……...…….21

3. РЕЗУЛЬТАТИ ТЕСТУВАННЯ……..27

3.1 Отримання геометричних зразків…….…….27

3.2 Інтегрування методом Монте-Карло…..……29

3.3 Стиснення і рендеринг…….……...29

ВИСНОВКИ ……....34

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ ……....35

ДОДАТОК…...……...……39

# ВСТУП

Серед сучасних методів на основі зображень - імпостори - потужна техніка, яка широко застосовується протягом багатьох років у графічному співтоваристві. Імпостори - це двовимірна текстура зображення, яка відображається на прямокутній карті чи рекламному щиті, створюючи ілюзію деталізованої геометрії. Імпостори можуть бути створені на льоту або попередньо обчислені та збережені в пам'яті. В обох випадках импостори точні лише для конкретного напрямку перегляду. Оскільки погляд відхиляється від цієї ідеальної умови, імпростори втрачають точність. Ці же імпростори повторно використовується, поки його візуальна помилка не перевищує заданий довільний поріг і не буде замінена більш точним імпростором.

Метод справжніх імпосторів використовує найсучасніше графічне обладнання для досягнення точного геометричного зображення, яке оновлюється кожен кадр для будь-якого довільного кута огляду. Справжні імпостори вибудовують попередні методи відображення зміщення на пікселі. Це також розширює мету додавання візуально точних підповерхових функцій на будь-яку довільну поверхню шляхом генерації цілих 3D-об’єктів на рекламному щиті. Метод справжніх імпосторів відрізняється від попередніх методів картографування переміщення на піксель, оскільки він не обмежує перегляд, але він забезпечує оптимізацію продуктивності та справжнє множинне заломлення. Як і традиційні імпостори, підхід справжніх імпосторів обертає квадрат навколо його центру, щоб він завжди стикався з камерою. На відміну від традиційної техніки імпосторів, яка відображає

статичну текстуру, справжня техніка імпостора використовує піксельний шейдер для викиду оглядового променя через квадроцикл у просторі текстури-координати для перетину 3D-моделі та обчислення кольору в точці перетину. Координатно-координатний простір визначається рамкою з центром квадрата як початком.

Ми почнемо з визначення наших цілей в цьому процесі. По-перше, ми маємо намір створити уявлення, яке забезпечує високоякісне візуалізоване зображення для всіх видів з певною клітинкою перегляду . Це означає, що ми намагаємося уникати таких артефактів, як діри, накладення псевдонімів або пропущених даних, і що нам потрібна висока точність відтворення візуальних змін зовнішнього вигляду. По-друге, ми наполягаємо на тому, щоб наше уявлення було настільки компактним, наскільки це можливо, і в той же час піддається апаратному прискоренню на графічному обладнанні споживчого рівня.

Мета даної роботи – створити результуюче уявлення яке є компактним, швидко рендерится з використанням існуючого графічного обладнання, створює високоякісне зображення з невеликим накладенням, не має відсутніх або зникаючих об'єктів при переміщенні точки огляду і правильно представляє залежні від виду зміни в клітинці перегляду.

Об’єкт дослідження – выборочное представление пространства объектов, фотореалистичный рендеринг.

Предмет дослідження – демонстрація методу застосування у міській візуалізації, модифікація точних методів, з метою зменшення часу рендеринга на один кадр, і відображення їх на архітектурі.

Актуальність: На сьогодні рендеринг є важливим інструментом для створення фотореалістичної візуалізації, мета якої затверджувати та впроваджувати концепції, виявляти проблеми та продавати продукти.

# 1 ОГЛЯД ІСНУЮЧИХ АЛГОРИТМІВ

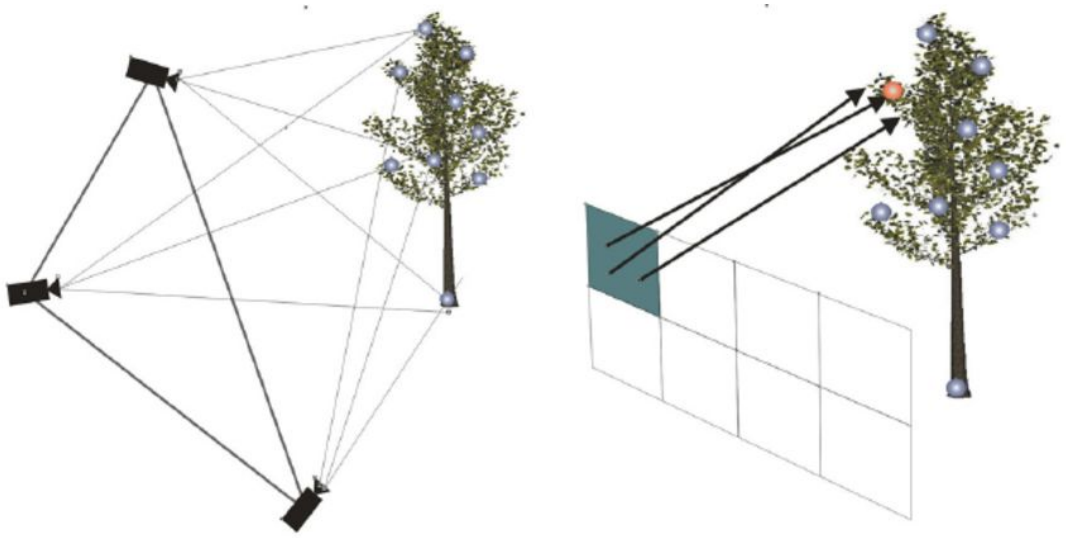


Рис. 1. Зліва: вибірка геометрії з використанням трьох перспективних камер LDI з клітинкою перегляду. Справа: промені відкидаються з клітинки перегляду для обчислення інтеграла Монте-Карло. Карта текстури на площині вибірки записує вклади для різних місць розташування клітин перегляду.

Ми пропонуємо примітив рендеринга, заснований на точках, щоб замінити геометрію, видиму з клітинки перегляду. Система реалізує наступний конвеєр:

- Геометрія вибірки: ми розраховуємо точки вибірки геометрії, використовуючи три перспективних LDI, щоб отримати щільну вибірку для клітинки перегляду (рис. 1, зліва). Ми називаємо площину, яка визначається трьома місцями розташування камери, площиною вибірки.
- Зовнішній вигляд вибірки: згладжений зовнішній вигляд кожної точки розраховується для різних місць розташування клітинки перегляду з використанням інтеграції Монте-Карло. Ми знімаємо промені з прямокутної області, яка міститься в трикутнику, утвореному трьома камерами (див. Рис. 1, праворуч). Кожна точка пов'язана з текстурною

картою, яка кодує вклади зовнішнього вигляду для різних місць розташування клітинки перегляду.

- Відображення в реальному часі: імпостори на основі точок в повній мірі використовують існуюче обладнання для рендеринга для перетворення кожної точки, а також для її затінення за допомогою пов'язаної з нею карти текстури в залежності від поточного місця розташування засобу перегляду. Ніяких додаткових програмних розрахунків не потрібно.

У віддаленій геометрії є деякі особливості, що випливають з її складності, які ускладнюють спрощення:

- Один аспект полягає в тому, що через обмеження дозволу зображення і перспективної проекції, як правило, кілька трикутників проектуються на один піксель. Це робить згладжування і фільтрацію важливою проблемою.
- Інший аспект полягає в тому, що ми не можемо покладатися на поверхні для управління процесом спрощення: трикутники, що проектуються на один піксель, можуть походити від розрізнених поверхонь і навіть від різних об'єктів. Це означає, наприклад, що для такого пікселя немає чітко визначеної нормалі.

Як правило, спрощення може ґрунтуватися або на геометричних умовах, як в більшості підходів рівня деталізації, або на уявленнях на основі зображень.Однак ми відзначаємо, що жодне з спрощених уявлень, запропонованих на сьогоднішній день, не відповідає всім цілям, зазначеним вище:

- Багатошарові зображення глибини (LDI) і аналогічні уявлення вносять зсув в шаблони вибірки і не можуть пристосуватися до ситуацій, в яких багато примітиви проектуються на кожен піксель виду. При наявності

такої мікрогеометрії ми також повинні знати, що зовнішній вигляд може змінюватися в залежності від напрямку перегляду. Це вимагає подання з залежною від напрямку інформацією, такий як світлове поле.

- Підходи світлового поля досить потужні, щоб впоратися зі складним і спрямованим зовнішнім виглядом. Проте, геометрична інформація повинна бути включена в світлове поле, щоб отримати прийнятну якість в межах розумного кількості місця для зберігання [3]. Можливим початковим пунктом для побудови імпостора було б поверхневе світлове поле [28], але, як вже вказує назва цього примітиву, для цього потрібна поверхня (та її параметризація).

Ми представляємо нову виставу для складної геометричної моделі, яка особливо корисна для побудови об'єктів імпостора, щоб замінити далеке поле для даної клітинки перегляду. Пропоноване уявлення ефективно розділяє геометрію, представлену у вигляді набору тривимірних точок, і зовнішній вигляд, представлений у вигляді спрощеного світлового поля, обчисленого по вихідної моделі. У нашому алгоритмі точки вибираються з ряду точок огляду, обраних таким чином, щоб наблизитися до мінімального критерію вибірки в просторі зображення по всій клітинці перегляду. Тому наш механізм відбору проб запобігає появі отворів. В контексті дуже складних моделей, що складаються з безлічі незалежних примітивів, тривимірні точки не можуть розглядатися як представники чітко визначеної поверхні, як в методах surfel або QSplat [20, 21]. Навпаки, ми визначаємо правильну характеристику внеску яскравості в зображення від цих точок, яка може бути обчислена з використанням інтеграції Монте-Карло і закодована як карта текстури для моделювання ефектів, що залежать від виду. Результуюче уявлення є компактним, швидко рендерится з використанням існуючого графічного обладнання, створює високоякісне зображення з невеликим накладенням, не

має відсутніх або зникаючих об'єктів при переміщенні точки огляду і правильно представляє залежать від виду зміни в межах клітинки перегляду (такі як оклюзія / роз'єднання) ефекти або зміни зовнішнього вигляду).

### 1.1 Алгоритм та деталі реалізації

Одинарна чотириканальна текстура може містити чотири поля висоти, які можуть представляти багато томів. Можна додати більше текстурних даних, щоб розширити цю форму до будь-якої форми. Ми генеруємо рекламний щит (тобто квадратик, який завжди стикається з камерою) для використання у якості вікна для перегляду наших полів висоти з будь-якого заданого напрямку, як показано на рис. 2.

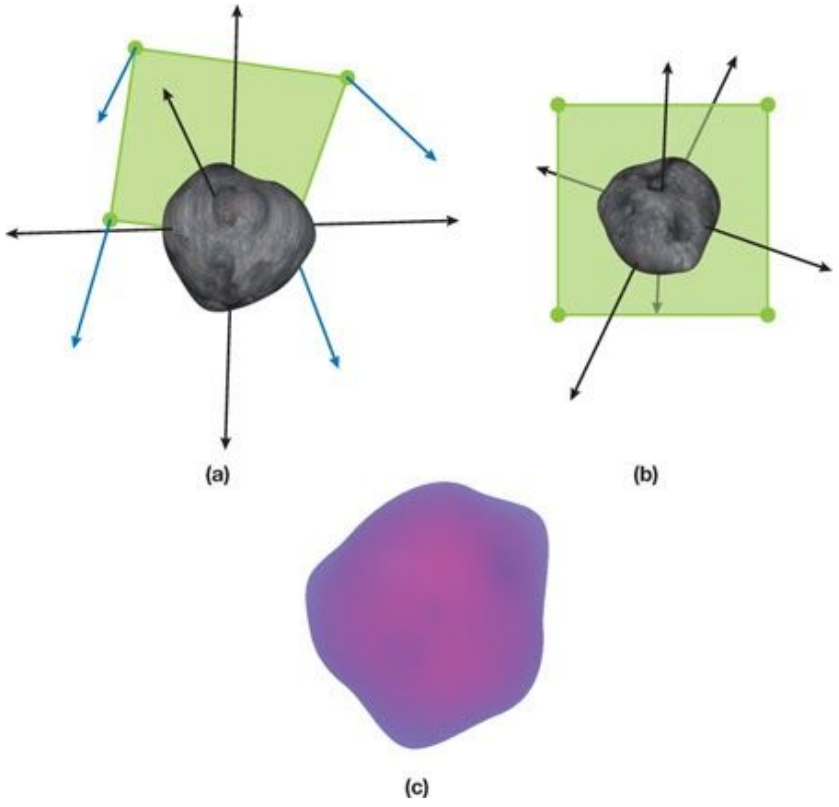



Рис. 2 Створення білборда

Зображення на рис.2(b) показує зображення координат текстури нашого щита після того, як вони були перетворені в тривимірну площину і обертаються навколо наших функцій (моделі) в центрі (представлених астероїдом). На рис.2(a) показано перспективу нашої функції, яка відображатиметься на

нашому рекламному щиті, а на рис.2(c) показана функція (у растровій формі), яка представляє наш астероїд.

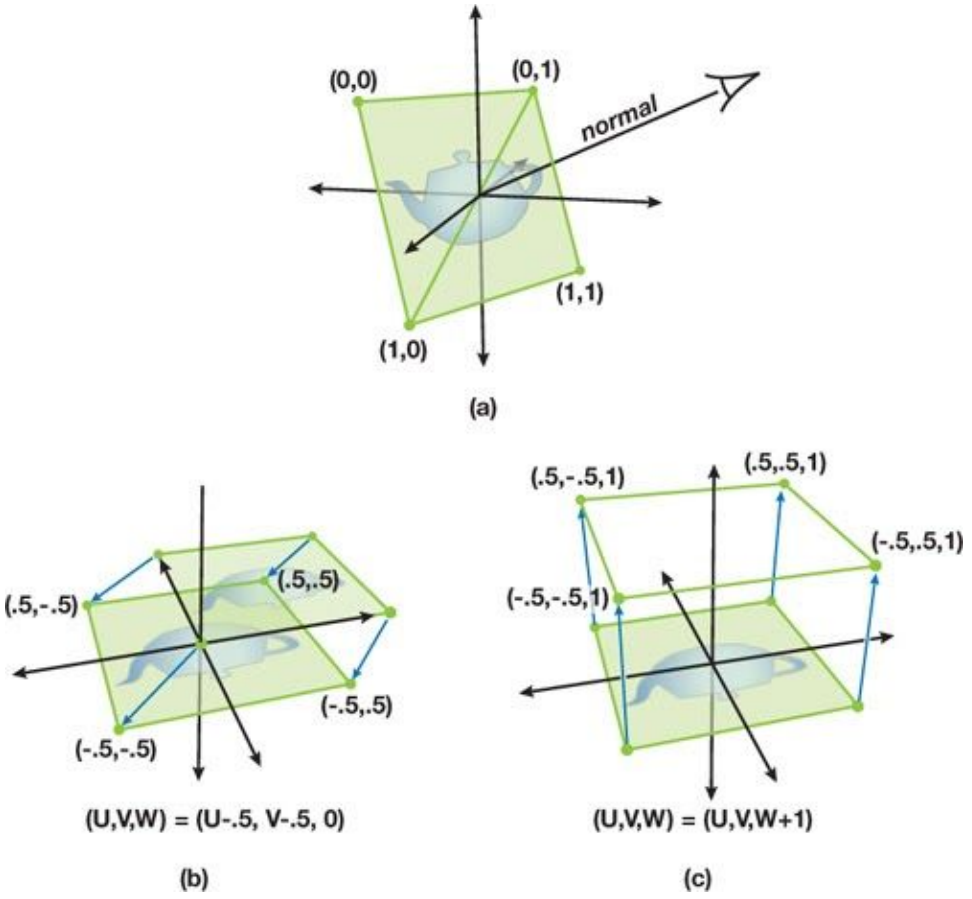



Рис.3 Геометрія проекцій справжніх імпосторів.

Щоб розширити цю концепцію, див. рис 3(a). Ми бачимо компонент цього методу, який діє на геометрію у світовому просторі. Камера переглядає квадратик, який обертається таким чином, щоб його вектор був нормальним, який утворюється між камерою та центром квадратика. Як показано, координати текстур присвоюються кожній вершині. На рис.3(b) розкривається текстурний простір, традиційно двовимірний простір, де ми додаємо третій вимір, W і зміщуємо координати УФ на –0,5. На рис.3 (c) ми просто встановлюємо W компонент наших координат текстури 1.

Майте на увазі, що карта текстури містить лише УФ-компоненти, тому вона пов'язана з двома вимірами і тому може бути переведена лише уздовж

УФ-площини, де будь-яке значення W буде посилатися на одну і ту ж точку. Хоча пов'язана з УФ-площиною, карта текстури може представляти дані про обсяг, обробляючи кожну з чотирьох змінних, що утворюють піксель як точки на осі W. На рис.4 ми бачимо проекцію об’єму текстури на тривимірний простір. Координати текстур кожної вершини обертаються навколо джерела - так само, як обертався наш початковий квадратик навколо її походження - для виготовлення білборда.

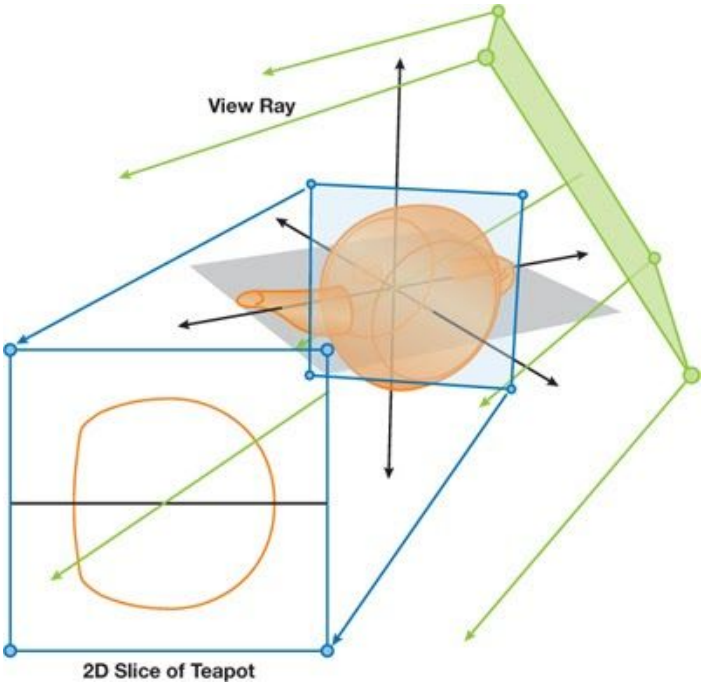



Рис.4 Променевий вигляд проекції, щоб взяти 2D-зріз.

Тепер ми вводимо оглядовий промінь у нашу концепцію. Промінь огляду виробляється шляхом відтворення променя від камери до кожної вершини. Під час растеризації як координати текстури, так і оглядові промені, що зберігаються у кожній вершині, інтерполюються по всій фрагментованій поверхні нашого квадратика. Це концептуально схоже на лиття променів або відстеження променів, оскільки тепер у нас є екран перегляду точок початку, де кожна точка має відповідний вектор для визначення променя. Слід також зазначити, що кожен окремий промінь проектується як пряма лінія на нашу

площину карти текстури, і тому виймає 2D зріз із тривимірного об'єму для оцінки зіткнень. Це детально показано на рис.4 .

На даний момент існує два окремих варіанти, залежно від бажаного типу матеріалу. Для непрозорих об'єктів лиття променів - найшвидший і найточніший варіант. Упорядкований підхід до цього варіанту представлений у Полікарпо та Олівейра 2006 р., Використовуючи марш променів з подальшим двійковим пошуком, як показано на рис.5 . На рис.5 (a) показаний лінійний пошук у дії. Промінь огляду випробовується через рівні проміжки часу, щоб побачити, чи перетинався він з гучністю. Як тільки знайдено перший перетин, він і попередня точка використовуються як початкова і кінцева точки бінарного простору пошуку. Потім цей простір повторно ділиться навпіл, поки не буде знайдена точна точка перетину в обсязі.

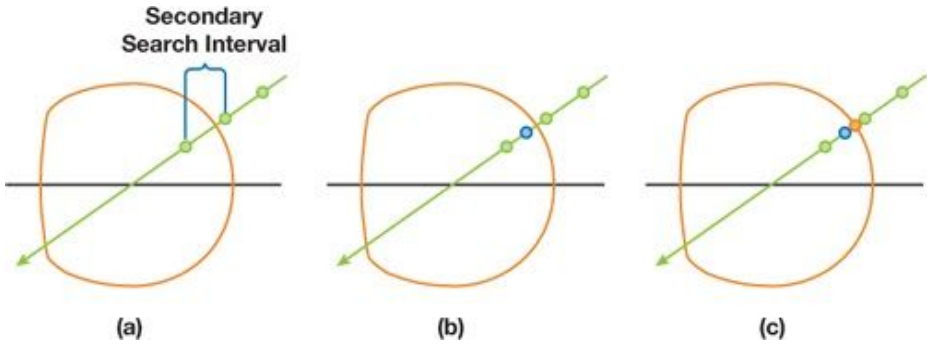



Рис.5 Променевий марш та Двійковий пошук для визначення перетину гучності.

Через характер дизайну GPU недоцільно з точки зору ефективності виходити з циклу рано; отже, повинна бути виконана максимальна кількість променів і зчитування текстури незалежно від того, наскільки рано в процесі пошуку знайдено перше зіткнення. Замість того, щоб ігнорувати ці "вільні" дані, ми пропонуємо метод для додавання підтримки для візуалізації для напівпрозорих типів матеріалів за допомогою локалізованого наближення.

Замість того, щоб здійснювати двійковий пошук вздовж оглядового променя, ми використовуємо точки вздовж осі W для визначення лінійного рівняння для кожного поля висоти. Потім ми перевіряємо ці лінійні рівняння на перетин з оглядовим променем аналогічним чином, як показано в Tatarchuk 2006.

Як показано на рис. 6 , перетин, що падає між верхньою та нижньою межами, зберігається як кінцева точка перетину. Оскільки матеріал напівпрозорий, норма перевіряється на цю точку. Плюс оглядовий промінь заломлюється відповідно до показника заломлення або утримується постійним для шейдера, або зберігається в додатковому кольоровому каналі в одній із карт текстури. Після виявлення нового напрямку зору промінь проходить, і процес повторюється для будь-яких додаткових поверхонь, які проникли. Підсумовуючи пройдені відстані через будь-який об'єм, ми визначаємо загальну пройдену відстань через нашу модель і використовуємо її для обчислення просвічуваності для цього пікселя.

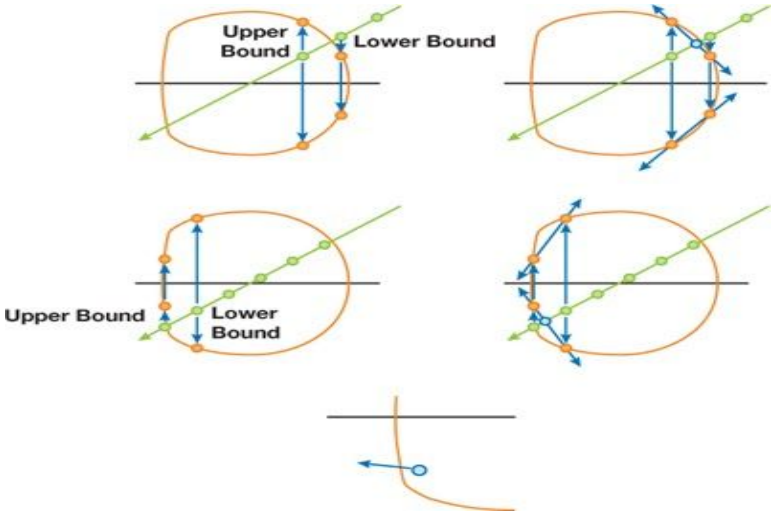



Рис.6 Розширення тестів перетину для підтримки напівпрозорості

# 2 ТОЧКОВІ ІМПОСТОРИ

## 2.1 Складність появи

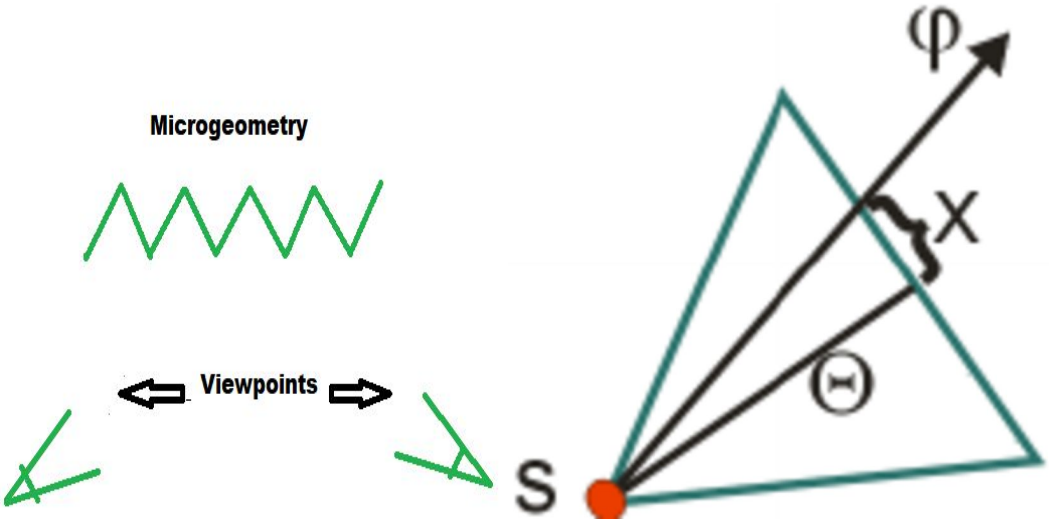



Рис. 7. Ліворуч: Мікрогеометрія: якщо структура на малюнку проектує лише піксель для клітинки перегляду, потрібна інформація про напрямок, щоб вона відображалася в затіненому світлі в лівій камері, а в правій - темна. Праворуч: на малюнку показано співвідношення параметрів плеоптичної функції та параметрів функції плеоптичного зображення.

Плеоптична функція P (s, ф) повністю моделює те, що видно з будь-якої точки s простору в будь-якому заданому напрямку ф.Таким чином, надання є процесом реконструкції частин плеоптичної функції із зразків.Інформація про геометрію та зовнішній вигляд предметів сцени сприяє плеоптичній функції. Поточні алгоритми візуалізації зазвичай підкреслюють один із таких аспектів:

- Світлові поля записують промені незалежно від геометричної інформації, навіть якщо всі вони потрапляють на ту саму розсіяну поверхню.

- Інші, геометричні підходи, як правило, моделюють зовнішній вигляд лише в тій мірі, дозволеної моделлю освітлення. Вони не можуть враховувати мікрогеометрію, тобто геометрію, яка проектує менше ніж піксель. Мікрогеометрія може мати різний вигляд, якщо дивитися з різних кутів (див. Рис. 7, зліва).Зазвичай це призводить до випромінювання артефактів або усунення розмивання, тим самим можливе відкидання важливої інформації.

Наше точкове подання містить як геометричну інформацію (тривимірне розташування), так і інформацію про зовнішній вигляд, що залежить від напрямку. Тому він охоплює всі ці аспекти:

- Точковий відбір може легко адаптуватися до неструктурованої геометрії за відсутності поверхонь або структурованої моделі.
- Точка на розсіяній, локально рівній поверхні, яка виступає на більше пікселя, має однаковий вигляд з усіх боків і може бути закодована одним кольором.
- Об'єкти із світловими ефектами, що залежать від виду (наприклад, дзеркальне освітлення), показують різні кольори, якщо дивитися з різних кутів.
- Мікрогеометрія також залежить від зору і кодується так само, як світлові ефекти.

**2.2 Геометрична вибірка**

Перший крок у створенні точкового відображення примітиву - це визначитися з геометричним розташуванням зразкових точок. Точки кодують частину плеоптичної функції, визначену клітинкою перегляду та геометрією для вибірки. Щоб дозволити реконструкції обладнання, не повинно бути жодних отворів під час проектування точок у будь-який можливий вигляд із клітинки перегляду.

Достатнім критерієм того, що поверхня, що відбирається в точці, не матиме отворів у проекції, - це те, що прогнозовані точки можуть бути трикутними, щоб максимальна проектована довжина краю була меншою за бічну довжину пікселя [10].

Загалом важко довести, що конкретне представлення на основі точки відповідає цьому критерію. Для одиничного збільшення та ортографічного перегляду три ортогональні LDIs забезпечують таку адекватну вибірку [13].

Незважаючи на те, що ця стратегія вибірки працює з нашим методом, вона не дуже підходить для об’єктів далекого поля, оскільки щільність вибірки не пристосована до можливих напрямків перегляду з клітинки перегляду і, як правило, призводить до великої кількості точкових зразків. Тому наш метод вибирає перспективні LDIs для кращого розподілу зразків відносно клітинки перегляду.

### 2.3 Функція пленоптичного зображення

Геометричний крок вибірки приводить до набору n точок $p_1$ , ..., $p_n$ з фіксованими місцями у 3D просторі. У цьому розділі ми отримуємо метод визначення значень кольору для кожної з цих точок.

Ми досліджуємо значення точки у світлі нашої мети: для кожного місця перегляду та положення камери в клітинки перегляду зображення, отримане шляхом надання всіх точок $p_1$ , ..., $p_n$ , має бути правильно відфільтрованим зображенням дальних об'єктів поля (тобто один зріз пленоптичної функції P (s, ф). Наше виведення покаже, що для цього потрібно кілька кольорів на точку.

Більшість алгоритмів візуалізації використовують z-буфер або упорядкування глибини для визначення видимості примітивів, які вони надають. Точка, однак, нескінченно мала. Його значення в основному визначається його реконструкцією. У багатьох методах реконструкції (як,

наприклад, при апаратній візуалізації) на піксель видно лише одна точка і визначає колір пікселя. Завдяки кінцевій роздільній здатності зображення точка може бути видимою або оклюзійною залежно лише від орієнтації камери перегляду. Пленоптична функція недостатньо потужна для моделювання цієї форми оклюзії.

Отже, ми вводимо пленоптичну функцію зображення PIF (s, θ, x). Параметр s представляє положення 3D-переглядача, θ - орієнтація на камеру, а x - 2D піксельна координата в локальній системі координат камери.Рис.7 (праворуч) ілюструє, як домени PIF і пленоптична функція відносяться один до одного: будь-яка пара (θ, x) відповідає одній орієнтації променів ϕ (θ, x) від пленоптичної функції, тому в ідеальних умовах ( тобто нескінченно точні дисплеї тощо), PIF пов'язаний з пленоптичною функцією P через те, що це відображення багато в одному.

$$PIF(s, \theta, x) = P(s, \phi(\theta, x))$$

PIF з його додатковими параметрами дозволить нам включити термін видимості, притаманний сучасним методам реконструкції, безпосередньо в наші розрахунки.

Тепер ми інтерпретуємо точки через зображення, які вони створюють (їх функції зображення), коли вони відображаються за допомогою певного алгоритму візуалізації - у нашому випадку апаратного візуалізації з буфером.

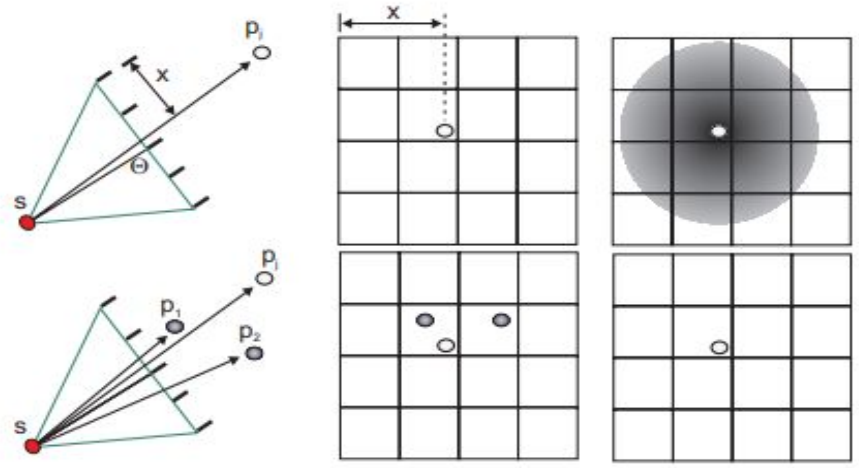



Рис. 8. Ці фігури пояснюють функцію зображення точки $p_j$ та вплив видимості. У лівій колонці зображено 2D перетин для фіксованих параметрів камери (s, θ). Зверніть увагу, що ми показуємо лише один компонент 2D координат зображення x. Центральний стовпчик показує точку $p_j$ на отриманому зображенні. Правий верхній рисунок показує функцію зображення $r_j$ (s, θ, x). Хоча верхній рядок вважає лише точку $p_j$, у нижньому рядку додано дві додаткові точки $p_1$ та $p_2$. Тут точка $p_1$ окулює точку $p_j$ за рахунок буфера z, внаслідок чого термін видимості $\upsilon_j$ (s, θ) точки $p_j$ дорівнює 0. Тому $Q_j$ (s, θ, x) (у правій нижній фігурі) дорівнює 0 для всіх координат зображення x.

Якщо ми розглянемо лише одну точку $p_j$, то точка визначає функцію зображення $r_j$ (s, θ, x). Ця функція задає безперервне зображення для кожного набору параметрів камери (s, θ). Кожне з цих зображень встановлюється на типовий фільтр реконструкції монітора, Гаусса з центром на пікселі, на який точка проектує (див. Рис. 8, вгорі).

Однак дуже важливо бачити точку $p_j$ по відношенню до інших точок у наборі точок. Ми повинні враховувати термін видимості $\upsilon_j$, який дорівнює 1, якщо точка pj видно на зображенні, визначеному через параметри камери (s,

θ) та 0 в іншому випадку. Це дає нам фактичну функцію зображення $Q_j$ точки (див. Рис. 8, внизу):

$$Q_j\ (\mathrm{s}, \theta, \mathrm{x}) = \upsilon_j\ (\mathrm{s}, \theta)\, r_j\ (\mathrm{s}, \theta, \mathrm{x})$$

З алгебраїчної точки зору ми можемо розглядати функції $Q_j$ , 1 ≤ j ≤ n як базові функції, що охоплюють кінцевомірний підпростір простору всіх PIF. Кінцевий елемент $PIF_{finite}$ цього кінцевомірного підпростору може бути записаний у вигляді зваженої суми базових функцій точок:

$$PIF_{finite}(\mathrm{s}, \theta, \mathrm{x}) = \sum_j c_j Q_j(s,\ \theta,\ x)$$

Зауважте, що вага $c_j$ - значення кольору, присвоєне точці $p_j$ . Ваги $c_j$ слід вибирати так, щоб зробити $PIF_{finite}$ максимально наближеним до PIF. Цього можна досягти, мінімізуючи $PIF_{finite}$ - PIF (стосовно евклідової норми щодо простору функцій над (s, θ, x)), і отримані ваги можуть бути потім знайдені через функції подвійної основи, як

$$c_j = \iiint q_j(s,\ \theta,\ x) PIF(s,\ \theta,\ x) ds\ d\theta\ dx$$

де $q_j$ (s, θ, x) - функція подвійної основи (подвійні базисні функції визначаються через $\langle q_j, Q_k \rangle = \delta_{jk}$ ).

Це уявлення призначає тільки одне значення кольору для кожної точки незалежно від місця розташування s в осередку виду. Таким чином, ми робимо модель більш потужною, замінюючи ваги $c_j$ на функції $c_j$ (s). Для подання цих функцій вони дискретно ставляться до простору можливих місць перегляду (це за духом аналогічно діафрагмі камери, необхідній при вибірці світлих полів).

З огляду на базис $B_i$ (s) для цього простору, ми можемо записати кожен $c_j$ (s) в рамках цієї бази з коефіцієнтами, знайденими через подвійні бази $b_i$ (s):

$$c_j \text{ (s)} = \sum_i c_{ij} \, B_i \text{ (s)}, \text{ де } c_{ij} = \int c_j \text{ (s)} \, b_i \text{ (s)} \, ds$$

Один з можливих виборів для основи $B_i$ (s) - використовувати регулярне підрозділ площини з нормальним напрямком від осередку огляду до об'єктів. Ваги $c_{ij}$ для кожної точки $p_j$ можуть бути збережені у карті текстури. Це означає, що $c_j$ (s) будуть шукати на карті текстури, яка, як правило, реконструюється з білінеарним ядром, тому $B_i$ (s) насправді є лише білінеарною основою.

Збираючи всі частини разом, процес візуалізації описується формулою

$$PIF_{finite} \text{ (s, θ, x)} = \sum_{i,j} c_{ij} Q_j \text{ (s, θ, x)} \, B_i \text{ (s)}$$

і ваги, щоб зробити $PIF_{finite}$ найбільш схожим на PIF, обчислюються як

$$c_{ij} = \iiint PIF(s, \theta, x) \, q_j(s, \theta, x) b_i(s) ds \, d\theta \, dx \quad (1)$$

Останнє завдання - знайти подвійні базисні функції. Якби наші базисні функції $Q_j$ $B_i$ були ортогональні, ми б мали $q_j = Q_j$ і $b_i = B_i$ (крім питань нормалізації).

У не ортогональному випадку, однак, обчислення реальних дуалів є втомливим: геометрична оклюзія означає, що кожна з них була б різною; дійсно доведеться перевернути велику матрицю для кожного, щоб знайти (дискретно) версію дуалу. Ми обрали приблизний підхід, натхненний теорією сигналу: і білінеарну основу $B_i$, і фільтр реконструкції монітора $r_j$ можна розглядати як наближення до ідеального (і ортогонального) фільтра реконструкції, функції sinc. На звичайній практиці, ми наближаємо ідеальний

фільтр низьких частот (сигнально-теоретична версія подвійних базових функцій) за допомогою Гаусса.

Інтеграл (1) можна оцінити, використовуючи інтеграцію Монте-Карло з важливими функціями $b_i$ та $q_j$. Зразки PIF (s, θ, x) = P (s, ϕ (θ, x)) обчислюються за допомогою простого відстежувача променів.

# 3 РЕЗУЛЬТАТИ ТЕСТУВАННЯ

У цьому розділі ми опишемо вибір, зроблений метод в нашій конкретній реалізації.

## 3.1 Отримання геометричних зразків

Для розміщення трьох камер LDI, які використовуються для отримання геометричних зразків, для параметризації позицій глядача у вікні огляду вибирається площина вибірки: як у світлих полях, ми використовуємо площину, орієнтовану від клитинки виду у далеке поле.Потім обчислюємо перетин опорних площин оглядової комірки та дальнього поля з площиною вибірки. Вибираємо три камери на трикутник, який щільно обводить отриманий многокутник (тобто трикутник з мінімальною окружністю).

Для того, щоб визначити дозвіл LDI, необхідний для уникнення дірок у перепроектуванні відібраних точок, наш метод пропонує механізм вибірки, заснований на 2D спостереженні: припустимо, у нас є дві камери на обох кінцях сегмента можливих місць перегляду у 2D, і короткий край, який переглядають дві камери. Ці дві камери використовуються для вибірки кінцевих точок краю. Переустановка відбувається з камери, розміщеної в будь-якому місці сегмента між двома камерами. Тепер може бути показано, що "найгірша" невідповідність між кутом перестановки та кутом вибірки з'являється, якщо край паралельний відрізку, що з'єднує дві камери, і якщо центр краю рівновіддалений від обох камер.

Натхненний цим, ми розробили евристику в 3D, яка враховує такі ефекти:

- Рух всередині клітинки виду: для трьох камер, що дивляться на площину, паралельну площині відбору проб, найгірша роздільна здатність взаємного відбору проб виникає в точці на площині, що рівновіддалена від трьох камер. Ця точка - це окружність трикутника,

визначена трьома камерами, спроектованими на відповідну площину. Кут вибірки трьох камер повинен бути обраний таким чином, щоб сусідні зразки проектували менше пікселя при перегляду безпосередньо з клітинки (де відбувається максимальна проекція).

- Перспектива: перспективні камери мають різну кутову роздільну здатність у просторі: функція (наприклад, трикутник) проектує на більше пікселів на межі фрустума огляду, ніж у центрі, якщо їх бачити під одним кутом огляду. Роздільну здатність вибірки має бути відповідно збільшена на коефіцієнт між кутовою роздільною здатністю в центрі камери та на межі огляду. Наприклад, для поля зору 45◦ цей коефіцієнт становить приблизно 1,17.
- Орієнтація шаблону вибірки: якщо для відбору проб використовуються перспективні камери, зразок відбору зразків на об'єктах у просторі нахилений щодо камери перегляду. Отже, з огляду на мінімальний кут вибірки від точки, роздільну здатність має бути обчислена за діагоналлю пікселя, а не за краєм пікселя. На це припадає коефіцієнт √2.

Роздільна здатність вибірки вибирається відповідно до цих факторів. На практиці ми використовуємо просту евристику, щоб зменшити кількість балів: багато балів достатньо відібрані вже однією камерою. Це означає, що тріангуляція зразка вибірки з цієї камери в невеликому мікрорайоні не містить краю, який може надати більше пікселя в будь-якому режимі. Якщо точка, записана однією камерою, лежить в області, яка відбирається краще і достатньою мірою іншою камерою, ми видаляємо її. Зазвичай це зменшує кількість точок на 40–60%, що призводить до співвідношення приблизно 2–3 балів, прогнозованих на екранний піксель.

### 3.2 Інтегрування методом Монте-Карло

Метою кроку інтегрування Монте-Карло є оцінка інтегралу (1) для отримання ваги $c_{ij}$ функцій основи реконструкції. Домен в s - це прямокутник на площині вибірки. Індекс $c_{ij}$ відповідає одній точці $p_j$ та одному прямокутнику на площині вибірки, представленому texel $t_i$ на карті текстури для точки $p_j$. Вибираємо точку огляду s на цей прямокутник (згідно $b_i$ (s), розподіл Гаусса), випадкову орієнтацію камери θ, в якій точка знаходиться у фрустумі, і знімаємо промінь оклюзії, щоб перевірити, чи ця точка бачна у вибраній камері(це відповідає оцінці $\upsilon_j$ (s, θ)). Якщо його видно, ми вибираємо промінь відповідно до $q_j$ (s, θ, x) ( розподіл Гаусса, зосередженого на пікселі, на який точка проектує у вибраній камері) і додаємо його внесок у $t_i$. Промені знімаються до тих пір, поки дисперсія інтеграла не впаде нижче порогу, обраного користувачем.

### 3.3 Стиснення і рендеринг

Бали надаються за допомогою апаратного забезпечення OpenGL z-буфера. Інформація про зовнішній вигляд для кожної точки зберігається у текстури , параметризованій площиною вибірки. Координати текстури точки розраховуються як перетин оглядового променя до точки з площиною вибірки. Це також можна інтерпретувати як перспективну проекцію точки відбору проби в оглядовій фрустум, де вершина визначається точкою огляду, а межами площини проекції чотирма сторонами обмежувального прямокутника на площину відбору проб.

Перспективні проекції можуть бути виражені за допомогою однорідної текстури-матриці 4x4, наданої OpenGL. Однак, оскільки перемикання текстур для кожної точки є дорогим, ми пакуємо стільки точкових текстур в одну більшу текстуру, наскільки це дозволяє. Це вимагає додавання фіксованого

зміщення на точку до кінцевої координати текстури, що, хоча і не доступне в стандартному OpenGL, може бути здійснено за допомогою розширення програми вершини [5].

Інтерполяція між зразками текстур (що відповідає базовій функції $B_i$ (s)) проводиться за допомогою білінеарної фільтрації. Більш низьку якість, але швидший попередній перегляд подання можна візуалізувати, використовуючи лише один колір у точці та без текстури. Однак інформація про спрямованість у цьому випадку буде втрачена.

Щоб стиснути наше представлення, ми обчислюємо дисперсію кольорової інформації текстури, щоб визначити точки, яким потрібен лише один колір для всієї комірки подання. Подальше стиснення можна отримати, використовуючи апаратне квантове векторне квантування [6], яке забезпечує фіксований коефіцієнт стиснення 8:1. Зауважте, що це адаптується до складності сцени: регіони з низькою сприйнятою геометричною складністю завжди будуть представлені простими кольоровими точками.

Для наших тестів ми використовували 800 МГц Pentium III з графічною карткою GeForce II GTS. Програма вершин, яка використовується для обчислення координат текстури, моделюється в програмному забезпеченні. Щоб створити враження очікуваної продуктивності апаратної реалізації, ми також створили прості текстуровані точки.

Три різні тестові сцени використовуються для демонстрації поведінки точкових імпосторів. У таблиці 1 показані результати для кожної сцени на основі вихідної роздільної здатності екрана 640x480 пікселів. Він включає кількість точок в імпосторах, приблизну кількість пікселів, охоплених імпосторами для перегляду в центрі клітинки перегляду(на основі вікна, що обмежує екран), і отриману кількість точок на проектований піксель. У таблиці також показано час, необхідний для обчислення імпостора, та вимоги

до пам'яті. Останні два рядки представляють середню частоту кадрів, досягнуту для деяких позицій у клітинці перегляду.

Як правило, можна помітити, що для покриття площі екрану розміром приблизно 300x100 пікселів потрібно близько 75000 точок.

Хоча ми використовуємо неоптимізований відстежувач променів, час попередньої обробки все ще є розумним. Для кожної точки було знайдено карту текстури 8x2, яка достатньо захоплює ефекти, залежні від огляду, для розглянутих комірок подання (визначаючи розмір текстури)

| Results | scene1 | scene2 | scene3 |
|---|---|---|---|
| #points | 80,341 | 87,665 | 31,252 |
| #points/screen pixel | 2.35 | 2.42 | 2.8 |
| approx. #screen pixels | 34,000 | 36,000 | 11,000 |
| Preproc. time (min) | 22 | 41 | 31 |
| Memory (MB) | 1.6 | 1.75 | 0.6 |
| Rendering performance SW (Hz) | 36 | 32 | 98 |
| Rendering performance HW (Hz) | 60 | 54 | 160 |

Таблиця 1. У таблиці показані результати від імпосторів трьох сцен, обчислених для роздільної здатності екрана 640x480 пікселів. Апаратна візуалізація імітується шляхом надання простих текстурованих точок.

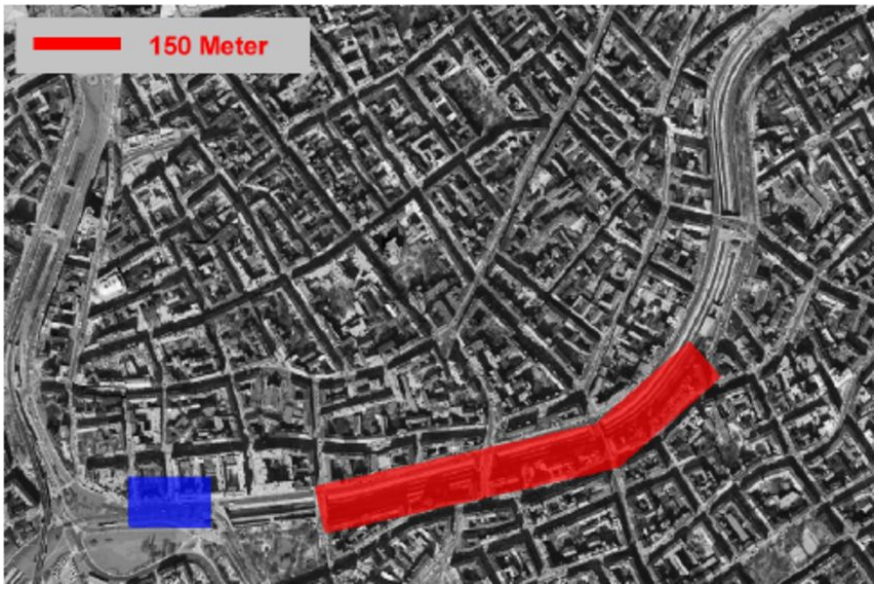


Рис. 9. На малюнку зображений імпостор зі сцени 3, розміщеної у місті. Оглядова клітина, показана у темному прямокутнику, має ширину 63 метри та довжину 120 метрів. Імпостор, показаний на світлому полігоні, довжиною близько 700 метрів і розміщений за 200 метрів від оглядової клітини.

Для кожної точки було виявлено карту текстури 8x2, яка достатньо охоплює ефекти, залежні від огляду, для розглянутих комірок подання (визначення розміру карти текстури автоматично вимагає подальшого дослідження). Вимоги до пам'яті перераховані без застосування зменшення текстур на одних кольорах на основі дисперсії.

Одним із сильних моментів нашого представлення є висока якість надання за наявності детальної геометрії. На рис. 11 (див. Рис. Б.1. - перший рисунок додатка Б) показано фільтрування багатьох тонких гілок дерев перед рухомим окуляром. Рис. 10 (див. Рис. А.1. - перший рисунок додатка А) демонструє правильну антиаліазію навіть для екстремальних кутів огляду на фасадах будівлі. На рис. 12 (див. Рис. В.1. - перший рисунок додатка В) показано, як самоврядування на основі точок може бути використане для поліпшення якості візуалізації далекого поля у міській сцені, що містить кілька полігонів на піксель. Слід зазначити, що покращення геометрії ще помітніше при переміщенні точки огляду. Крім того, імпостор на рис 12. (знизу) замінює геометричну модель близько 95 000 вершин, але складається лише з приблизно 30 000 точок.

Це показує, що імпостор не тільки покращує якість візуалізації дальнього поля, але і зменшує навантаження візуалізації графічного обладнання. На рис. 9 показано розміщення цього імпостора та його клітинки виду у моделі міста та створюється враження типових відносних розмірів оглядових клітинок та імпосторів. У сучасній експериментальній системі оглядові клітинки формуються з вуличних сегментів та імпосторів, розміщених на кінцях

відрізків вулиці, таким чином, як попередні системи імпосторів [8, 26]. Наша тестова сцена має 4 квадратні кілометри і складається з 2,1 мільйонів полігонів. Після застосування консервативного алгоритму видимості області [27] ми виявили оглядові осередки із занадто великою кількістю видимих багатокутників [2]. Приблизна оцінка вимог пам’яті за допомогою стратегії розміщення жорстоких зусиль призводить до отримання близько 165 Мб для 437 імпосторів. Зауважте, однак, що розробка добрих стратегій розміщення імпосторів не є однозначною та не підлягає постійним дослідженням. Наша майбутня робота також спрямована на зменшення потреби в пам’яті шляхом обміну інформацією між сусідніми клітинками перегляду та пошуку гарантій для забезпечення мінімальної частоти кадрів.

# ВИСНОВКИ

Ми представили точкові імпостори,  як нове високоякісне представлення на основі зображення для візуалізації в режимі реального часу.

Значення цього подання полягає в розділенні між задачею геометричної вибірки та поданням зовнішності. Відбір проб здійснюється за допомогою комбінування шаруватих зображень глибини для отримання належного покриття зображення для всієї клітинки перегляду. На основі математичного аналізу точкових моделей ми обчислюємо параметри візуалізації за допомогою інтегрування методом Монте-Карло, виключаючи більшість артефактів. Інформація про візуалізацію компактно закодована і може бути представлена на сучасному обладнанні. Точкові імпостори демонструють велику перспективу для всіх ситуацій, коли геометрично дуже складна модель являє собою далеке поле, як, наприклад, у міських проходах детальних моделей. Часи візуалізації вказують на те, що це подання застосовно до програм візуалізації в режимі реального часу, де потрібен час кадру понад 60 Гц.

Справжні імпостори - це швидкий, ефективний метод для надання великої кількості анімованих, непрозорих, відбиваючих або заломлюючих об'єктів на GPU. Це генерує імпостори з невеликою помилкою візуалізації та пропонує притаманний рівень деталізації пікселя. Представляючи дані обсягу у вигляді полів декількох висот, що зберігаються в традиційних картах текстур, використовується векторна обробка сучасних графічних процесорів і досягається висока частота кадрів, але з низькою потребою в пам'яті.

Відмовившись від обмежень, які за своєю суттю зберігають відображення переміщення за пікселем, відображаючи техніку деталей підповерхових поверхонь, ми домоглися нового методу відображення приголомшливих кількостей штучної геометрії.

## СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

[1] Daniel Aliaga, Jon Cohen, Andrew Wilson, Eric Baker, Hansong Zhang, Carl Erikson, Keny Hoff, Tom Hudson, Wolfgang Sturzlinger, Rui Bastos, Mary Whitton, Fred Brooks, and ¨ Dinesh Manoclia. MMR: An interactive massive model rendering system using geometric and image-based acceleration. In 1999 Symposium on interactive 3D Graphics, pages 199– 206, 1999.

[2] Daniel G. Aliaga and Anselmo Lastra. Automatic image placement to provide a guaranteed frame rate. Computer Graphics, 33:307–316, 1999

[3] Jin-Xiang Chai, Xin Tong, Shing-Chow Chan, and Heung-Yeung Shum. Plenoptic sampling. In Siggrapph 2000, Computer Graphics Proceedings, pages 307–318, 200

[4] Chun-Fa Chang, Gary Bishop, and Anselmo Lastra. LDI tree: A hierarchical representation for image-based rendering. In Siggraph 1999, Computer Graphics Proceedings, pages 291–298. ACM Siggraph, 1999.

[5] NVIDIA Corporation. Nv vertex program extension specification, 2000. available at http://www.nvidia.com/Marketing/Developer/DevRel.nsf/Pro-grammingResources Frame.

[6] NVIDIA Corporation. Using texture compression in opengl, 2000. available at http://www.nvidia.com/Marketing/Developer/DevRel.nsf/WhitepapersFrame.

[7] Lucia Darsa, Bruno Costa Silva, and Amitabh Varshney. Navigating static environments using image-space simplification and morphing. In 1997 Symposium on Interactive 3D Graphics, pages 25–34, 1997. ISBN 0-89791-884-3.

[8] Xavier Decoret, Franc¸ois Sillion, Gernot Schaufler, and Julie Dorsey. Multi-layered impostors for accelerated rendering. Computer Graphics Forum, 18(3):61–73, 1999. ISSN 1067-7055.

[9] Steven J. Gortler, Radek Grzeszczuk, Richard Szeliski, and Michael F. Cohen. The lumigraph. In SIGGRAPH 96 Conference Proceedings, pages 43–54, 1996. held in New Orleans, Louisiana, 04-09 August 1996.

[10] J. P. Grossman and William J. Dally. Point sample rendering. In Rendering Techniques ’98, pages 181–192, 1998.

[11] Marc Levoy and Pat Hanrahan. Light field rendering. In SIGGRAPH 96 Conference Proceedings, pages 31–42, 1996. held in New Orleans, Louisiana, 04-09 August 1996.

[12] Marc Levoy and Turner Whitted. The use of points as a display primitive. Technical Report TR 85-022, University of Carolina at Chapel Hill, 1985.

[13] Dani Lischinski and Ari Rappoport. Image-based rendering for non-diffuse synthetic scenes. In Rendering Techniques ’98, pages 301–314, 1998.

[14] P. Maciel and P. Shirley. Visual navigation of large environments using textured clusters. SIGGRAPH Symposium on Interactive 3-D Graphics, pages 95–102, 1995.

[1]. William R. Mark, Leonard McMillan, and Gary Bishop. Post-rendering 3D warping. In 1997 Symposium on Interactive 3D Graphics, pages 7–16, 1997. ISBN 0-89791-884-3.

[16] Nelson Max. Hierarchical rendering of trees from precomputed multi-layer Z-buffers. In Eurographics Rendering Workshop 1996, pages 165–174, 1996. ISBN 3-211-82883-4.

[17] Leonard McMillan and Gary Bishop. Plenoptic modeling: An image-based rendering system. In SIGGRAPH 95 Conference Proceedings, pages 39–46, 1995. held in Los Angeles, California, 06-11 August 1995.

[18] Alexandre Meyer and Fabrice Neyret. Interactive volumetric textures. In Rendering Techniques ’98, pages 157–168, 1998.

[19] Gavin Miller, Steven Rubin, and Dulce Ponceleon. Lazy decompression of surface light fields for precomputed global illumination. In Rendering Techniques ’98, pages 281–292, 1998.

[20] Hanspeter Pfister, Matthias Zwicker, Jeroen van Baar, and Markus Gross. Surfels: Surface elements as rendering primitives. In Siggraph 2000, Computer Graphics Proceedings, pages 335–342, 2000.

[21] Szymon Rusinkiewicz and Marc Levoy. QSplat: A multiresolution point rendering system for large meshes. In Siggraph 2000, Computer Graphics Proceedings, pages 343–352, 2000.

[22] G. Schaufler and W. Sturzlinger. A three-dimensional image cache for virtual reality. In ¨ Proceedings of EUROGRAPHICS’96, 1996.

[23] Gernot Schaufler. Per-object image warping with layered impostors. In Rendering Techniques ’98, pages 145–156, 1998.

[24] Jonathan Shade, Dani Lischinski, David Salesin, Tony DeRose, and John Snyder. Hierarchical image caching for accelerated walkthroughs of complex environments. In SIGGRAPH 96 Conference Proceedings, pages 75–82, 1996. held in New Orleans, Louisiana, 04-09 August 1996.

[25] Jonathan W. Shade, Steven J. Gortler, Li-wei He, and Richard Szeliski. Layered depth images. In SIGGRAPH 98 Conference Proceedings, pages 231–242, 1998. ISBN 0-89791- 999-8.

[26] Franc¸ois Sillion, G. Drettakis, and B. Bodelet. Efficient impostor manipulationfor real-time visualization of urban scenery. Computer Graphics Forum, 16(3):207–218, 1997. Proceedings of Eurographics ’97. ISSN 1067-7055.

[27] Peter Wonka, Michael Wimmer, and Dieter Schmalstieg. Visibility

preprocessing with occluder fusion for urban walkthroughs. In Rendering Techniques 2000, pages 71–82, 2000.

[28] Daniel N. Wood, Daniel I. Azuma, Ken Aldinger, Brian Curless, Tom Duchamp, David H. Salesin, and Werner Stuetzle. Surface light fields for 3D photography. In Siggraph 2000, Computer Graphics Proceedings, pages 287–296, 2000.

[29] Blinn, James F. 1978. "Simulation of Wrinkled Surfaces." In Proceedings of SIGGRAPH '78, pp. 286–292.

[30] Cook, Robert L. 1984. "Shade Trees." In Computer Graphics (Proceedings of SIGGRAPH 84) 18(3), pp. 223–231.

[31] Hart, John C. 1996. "Sphere Tracing: A Geometric Method for the Antialiased Ray Tracing of Implicit Surfaces." The Visual Computer 12(10), pp. 527–545.

[32] Hirche, Johannes, Alexander Ehlert, Stefan Guthe, and Michael Doggett. 2004. "Hardware-Accelerated Per-Pixel Displacement Mapping." In Proceedings of the 2004 Conference on Graphics Interface, pp. 153–158.

[33] Kaneko, Tomomichi, Toshiyuki Takahei, Masahiko Inami, Naoki Kawakami, Yasuyuki Yanagida, Taro Maeda, and Susumu Tachi. 2001. "Detailed Shape Representation with Parallax Mapping." In Proceedings of the ICAT 2001 (The 11th International Conference on Artificial Reality and Telexistence), pp. 205–208.

[34] Kautz, Jan, and H. P. Seidel. 2001. "Hardware-Accelerated Displacement Mapping for Image Based Rendering." In Proceedings of Graphics Interface 2001, pp. 61–70.

[35] Kolb, Andreas, and Christof Rezk-Salama. 2005. "Efficient Empty Space Skipping for Per-Pixel Displacement Mapping." In Proceedings of the Vision, Modeling and Visualization Conference 2005, pp. 359–368.

**ДОДАТОК А**

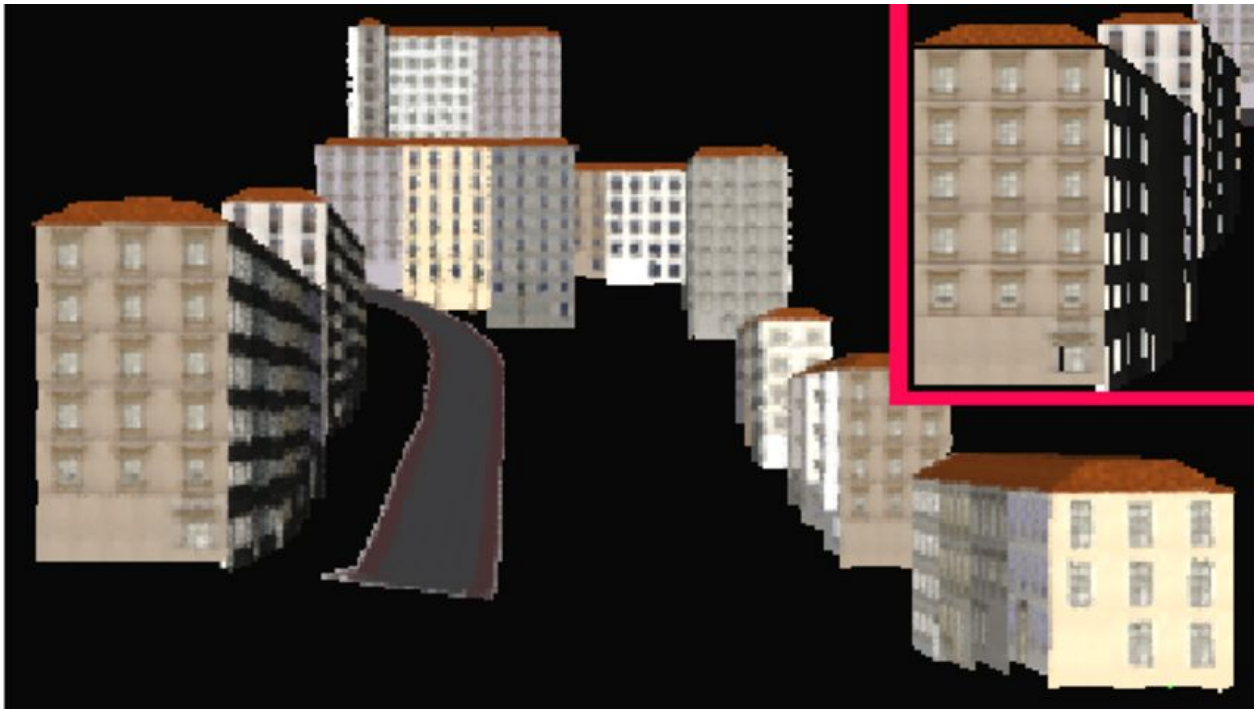


Рис. 10. Імпостор для ряду будівель. Вставка показує згладжування, яке вийшло б, якби була використана геометрія.

**ДОДАТОК Б**



Рис. 11. Зверніть увагу на правильну фільтрацію дерев проти фасаду будівлі та виразну підсвітку у вікнах для імпостора (вгорі) та суворого псевдоніму дерев за геометрією(знизу)

**ДОДАТОК В**



Рис. 12. Вгорі: імпостор для ряду будівель (вставка: геометрія). Центр: Фільтрування дерев для імпостора (вгорі) проти геометрії (знизу). Знизу: Імпостор для проходження по місту. Зверніть увагу на правильну фільтрацію імпостора(вгорі) порівняно з геометрією(знизу).