# ТЕОРЕТИЧНІ ПЕРЕДУМОВИ І МЕТОДИ ДОСЛІДЖЕННЯ

Існують два основні підходи до вирішення задачі спрощення тріангуляційних моделей: ітеративний і глобальний. Ітеративний підхід дозволяє послідовно спростити модель шляхом застосування деякого локального оператору. Глобальний, в свою чергу, застосовується до вхідної моделі в цілому. Ітеративний підхід, як правило, використовується більш широко, тому що з його допомогою зручніше контролювати результат, ступінь спрощення моделі і т.п. Наприклад, використовуючи такий алгоритм, користувач може встановлювати точну кількість граней результуючої моделі або ж вибирати в якості критерію зупинки певне порогове значення помилки. Виходячи з цього, ми будемо розглядати далі ітеративні алгоритми. Між собою їх можна розділити на декілька категорій, тому що кожен з алгоритмів може бути використаний для своєї специфічної задачі.

## Типи тріангуляційних поверхонь

З топологічної точки зору тріангуляційні поверхні можна поділити за двома основними категоріями: многовид і немноговид; також такі поверхні можуть мати або не мати граничні ребра. Можна сказати, що модель не є многовидом або має границі, якщо хоч одна вершина моделі є немноговидом або граничною вершиною відповідно. Порівняння типів вершин представлено на рис. 1.1.

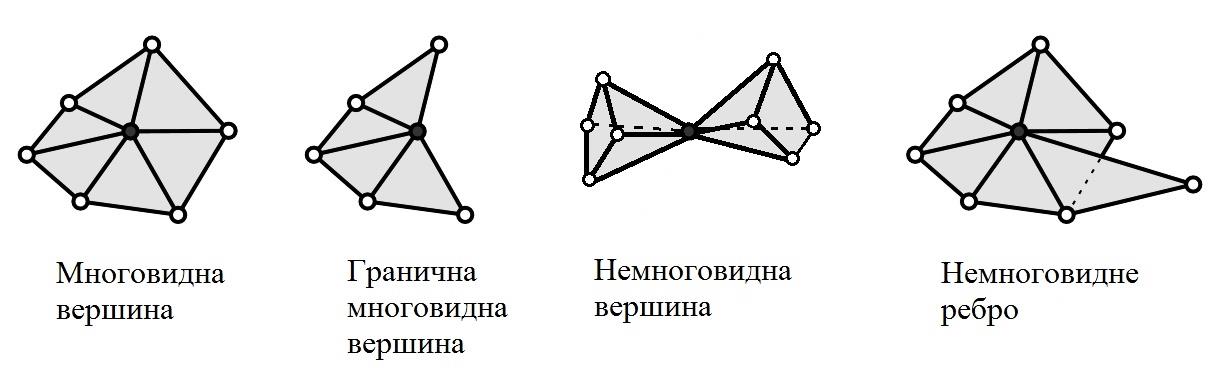


Рисунок 1.1 – Порівняння типів вершин тріангуляційної поверхні

Розглянемо більш детально типи вершин. Вершина тріангуляційної поверхні називається граничною, якщо вона належить до граничного ребра. Ребро, в свою чергу, є граничним, якщо воно має лише одну спільну грань.

Вершина називається немноговидною, якщо вона належить до немноговидного ребра, або якщо нормалі розгорнутої поверхні у малій околиці вершини в 2D не вказують в один і той самий напрямок. Ребро називається немноговидним, якщо воно має більше двох спільних граней.

Як вже було сказано раніше, підходи до спрощення тріангуляційних моделей різняться за типами вхідних даних. І якщо більшість алгоритмів підтримує збереження граничних ребер, то не так багато алгоритмів дозволяє спрощувати немноговидні моделі. Отже, розглянемо алгоритми спрощення тріангуляційних моделей щодо їх многовидності.

## Алгоритми спрощення многовидних тріангуляційних поверхонь

Не дивлячись на те, що підтримка обробки немноговидних поверхонь є більш загальною і дозволяє мати справу з більшою варіативністю об’єктів, існують задачі, для котрих це не потрібно. А отже це дає змогу зробити симпліфікацію вузького типу поверхонь більш якісно. Розглянемо найбільш популярні методи спрощування многовидних тріангуляційних моделей.

### Спрощування поверхні за допомогою енергетичної функції

Хоппе та ін. [1] запропонували алгоритм, мета якого полягає в тому, щоб отримати таку сітку, яка б найкращим чином відповідала множині точок, розташованій по поверхні, і при цьому містила якомога меншу кількість вершин. Така множина точок складається з початкових вершин, що становлять поверхню, і додаткових, які розміщуються на границях і рівномірно по всій поверхні. Для того, щоб досягти мети, необхідно знайти такий симпліціальний комплекс і таку множину вершин , що визначають сітку та мінімізують наступну енергетичну функцію:

|  |  |
| --- | --- |
|  | (1.1) |
|  |  |

Розглянемо складові цієї функції докладніше.

являє собою суму квадратів відстаней до сітки від точок :

|  |  |
| --- | --- |
|  | (1.2) |

, в свою чергу, це штрафна складова, яка збільшується, якщо вершин стає більше, і описується наступною формулою:

|  |  |
| --- | --- |
|  | (1.3) |

де – кількість вершин;

– параметр, який визначається користувачем і контролюючий компроміс між ступенем спрощення сітки і точністю геометричної відповідності.

відповідає за розтягнення ребер щодо їх первісного стану і визначається за формулою:

|  |  |
| --- | --- |
|  | (1.4) |

де – коефіцієнт пружності, встановлений заздалегідь.

Скорочення сітки відбувається ітеративно, шляхом виконання допустимих операцій на її ребрах: видалення ребра, поворот ребра, додавання нового ребра шляхом розщеплення поточного. Вибір допустимих дій проводиться за допомогою процесу оптимізації енергетичної функції. На кожному кроці видаляється елемент, видалення якого дає найменше збільшення енергетичної функції. Варто також відзначити, що при видаленні ребра, нова вершина розміщується або в одну з координат вершин, розташованих на кінцях цього ребра, або посередині ребра.

Пізніше Хоппе [2] встановив, що насправді для ефективного спрощення сіток достатньо всього однієї операції: видалення ребра. Також він запропонував модифікувати енергетичну функцію наступним чином:

|  |  |
| --- | --- |
|  | (1.5) |

визначається аналогічним попередньому алгоритму чином.

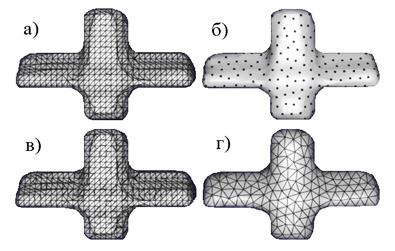
Визначення набуло деяких змін. Хоппе запропонував не задавати коефіцієнт пружності як константу, а міняти його в ході роботи алгоритму. Таким чином, буде постійно зменшуватися і адаптуватися при спрощенні сітки.

вимірює точність скалярних атрибутів сітки (наприклад, таким атрибутом може бути колір), а – геометричну точність кривих розриву сітки. Введення в функцію таких складових замість , дозволяє уникнути неоднозначного параметра , а також дозволяє вказувати явну кількість граней, які необхідно отримати в вихідній сітці. Також вони дають можливість зберегти не тільки геометрію сітки, а й її зовнішній вигляд.

З недоліків даного підходу можна виділити наступні. Розглянемо ситуацію видалення ребра. Як було сказано раніше, нову вершину пропонувалося помістити або в одну з координат вершин, що належать ребру, або посередині. Можна припустити, що це не найефективніше рішення, і можна було б замість цього розглянути функцію помилки, пов'язану з операцією стиснення, і спробувати мінімізувати її значення в порівнянні з простором можливих місць розміщення вершин. Також, не дивлячись на дуже хороші результати, які дає алгоритм, обчислення оптимальної цільової позиції для даного стиснення є нелінійною проблемою і на практиці є досить неефективним, а також складним в реалізації.

### Метод із застосуванням взаємної теселяції

Іншим підходом до спрощення многовидних сіток, що був представлений Турком [3], є реконструювання поверхні, яке включає в себе визначення нового набору вершин, що розподіляється по поверхні вхідної сітки, а потім використання цих вершин для формування нових граней поверхні. Ключовим поняттям у процедурі перебудування поверхні є створення проміжної моделі, що називається взаємною теселяцією поверхні. Новостворена модель містить як вершини з вхідної поверхні, так і нові точки, які також стають вершинами поверхні. Вихідна модель утворюється шляхом видалення кожної вершини, що належала до оригінальної моделі, і відповідної локальної тріангуляції поверхні таким чином, щоб зберегти локальну зв'язність поверхні. Алгоритм розподіляє нові вершини по поверхні сітки, як правило, за певним шаблоном, хоча також може бути використаний спосіб оцінки кривизни поверхні для розподілення більшої кількості нових вершин в областях, де кривизна поверхні має більше значення. Приклад роботи алгоритму наведено на рис. 1.2.



а) оригінальна модель; б) розподілення нових вершин по поверхні; в) взаємна теселяція; г) вихідна модель

Рисунок 1.2 – Етапи спрощення тріангуляційної моделі за допомогою метода із застосуванням взаємної теселяції

Точність цього підходу сильно залежить від методу, обраного для проектування нового набору вершин на поверхню.

### Метод, заснований на вокселях

Алгоритм спрощення, заснований на вокселях, був представлений Хі та ін. [4]. Слід зазначити, що воксель – це найдрібніша одиниця об’ємного зображення аналогічно пікселю у двовимірному зображенні. Отже, алгоритм сегментує вхідну модель в 3D сітку, що складається з вокселів, а потім застосовує фільтр нижніх частот для усунення вокселів з найменшими перекриттями об’ємів із вхідною поверхнею. Потім алгоритм використовує алгоритм маршируючих кубів [5] для генерації нової сітки на основі вокселів що були залишені. Алгоритм маршируючих кубів будує тріангуляційну поверхню, використовуючи підхід розділяй і володарюй. Він розділяє поверхню на куби та визначає, як саме поверхня розташована в певному кубі (перетинає його), а потім рухається (або марширує) до наступного куба.

Через те, що підхід фільтрації засновано на об’ємі, воксельне спрощення погано працює на моделях з гострими краями і квадратними кутами.

## Алгоритми спрощення немноговидних тріангуляційних поверхонь

Незважаючи на хороший результат, який показують алгоритми, що працюють тільки з многовидами, існує досить велика кількість прикладних задач, в яких використовуються немноговиди. Розглянемо найбільш відомі алгоритми, які були розроблені в цій сфері.

### Алгоритм проріджування вершин

Одним з перших алгоритмів спрощення тріангуляційних поверхонь, здатних обробляти немноговиди, став алгоритм проріджування вершин, що був представлений Шроудером та ін. [6]. Робота алгоритму полягає в ітеративному проходженні по всіх вершинах моделі. Під час проходжень для кожної вершини алгоритм визначає, чи можна видалити вершину без порушення топології її локальної околиці граней. Якщо результуюча поверхня буде перебувати на заданій користувачем відстані від оригінальної поверхні, алгоритм видаляє вершину і всі пов'язані з нею ребра. Отриманий отвір в сітці заповнюється шляхом локальної тріангуляції. Процес видалення вершин повторюється з можливим корегуванням критерію видалення, поки не буде досягнута умова виходу. Такою умовою, наприклад, може бути відсоткове зменшення поверхні.

Представлений алгоритм проріджування здатний обробляти немноговиди у разі, якщо він не буде видаляти немноговидні вершини під час ітерацій проріджування. Однак це правило накладає обмеження знизу для алгоритму, тому що він не може створити низькополігональну модель з меншою кількістю вершин, ніж число немноговидних вершин. Крім того, визначена користувачем відстань істотно впливає як на нижню межу, так і на точність алгоритму. Також мінус такого підходу в тому, що спрощення застосовується рівномірно до всього набору даних, що призводить до розмиття даних при нерівномірній структурі сітки.

### Алгоритм кластеризації вершин

Іншим підходом до спрощення немноговидних сіток є кластеризація вершин. Алгоритм кластеризації вершин, представлений Россіньяком і Боррелем [7], надає значення важливості кожній вершині в залежності від розміру її суміжних граней і їх кривизни. Потім алгоритм застосовує сегментацію поверхні: модель розміщують у паралелепіпед, що представляє собою 3D сітку. Залежно від щільності сітки, в кожну її клітину може потрапити як одна, так і декілька вершин. Якщо в клітині виявилося декілька вершин, розраховується, в якому місці буде розташована нова вершина, яка замінить собою попередні. Після того, як в кожній клітині залишається максимум по одній вершині, тріангуляційна модель перебудовується.

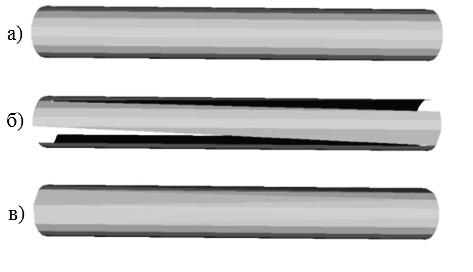
Мінус такого підходу в тому, що, не дивлячись на швидкість роботи і свою узагальненість, складно контролювати якість одержуваної моделі, і результат часто виявляється незадовільним. Також, досить проблематично зазначити бажану кількість граней вихідної моделі. Крім того, результат може залежати від положення моделі щодо паралелепіпеда.

### Алгоритм квадратичної метрики помилки

Можливо, найбільш поширеним підходом до спрощення немноговидних сіток є використання квадратичної метрики помилки (QEM), вперше представленої Гарландом і Хекбертом [8]. QEM вершини являє собою матрицю розміром 4 × 4, що в свою чергу є сумою матриць QEM по кожній грані, зв’язаної з даною вершиною. Коли вершина зливається з іншою вершиною в межах заданого користувачем порога відстані, значення QEM нової вершини може бути обчислено як сума матриць QEM об'єднаних вершин. При об'єднанні вершин алгоритм зберігає відсортовану чергу пріоритетів всіх пар вершин-кандидатів на основі їх об'єднаної QEM. Алгоритм видаляє пару вершин з найменшою помилкою з верхньої частини списку, об'єднує вершини і потім оновлює значення помилок тих пар вершин, які з'єднані з даною. Позиція нової вершини обчислюється на основі рішення задачі мінімізації функції помилки.

Даний підхід дозволяє отримати сітки зі збереженням досить високої точності, навіть при радикальному ступені спрощення. Його відмінною рисою є те, що він дозволяє спрощувати модель, з’єднуючи незв'язні частини моделі, а також дозволяє працювати з немноговидами. Крім того, як вже було сказано, в даному алгоритмі обчислюється найбільш оптимальне положення нових вершин, що не було враховано більшістю інших алгоритмів.

Одним з обмежень оригінального алгоритму QEM є те, що він має тенденцію до великих відхилень на граничних ребрах через зменшення числа суміжних граней, що призводить до виникнення великої кількості пропусків на границях (рис. 1.3). Щоб вирішити цю проблему і забезпечити збереження границь, Гарланд і Хекберт [9] визначили площину граничного обмеження, що проходить перпендикулярно до кожної граничної грані. Для кожної такої площини пропонується обчислити матрицю QEM, яка множиться на досить великий постійний ваговий коефіцієнт, і додати її до матриць QEM вершин, що належать до граничних ребер.



а) оригінальна модель (36 граней); б) спрощена модель за оригінальним алгоритмом QEM (28 граней); в) спрощена модель за алгоритмом QEM з граничними обмеженнями (28 граней)

Рисунок 1.3 – Порівняння результатів роботи оригінального алгоритму QEM та алгоритму QEM з граничними обмеженнями

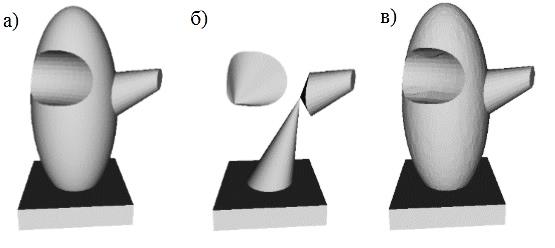
Різні дослідники також запропонували зважувати площину граничного обмеження як квадрат довжини кромки. Проте, всі ці підходи призводять до того, що граничні ребра стають більш пріоритетними, ніж всі інші, що призводить в результаті до спрощених сіток з отворами на їх поверхнях (рис. 1.4).

Бахірат та ін. [10] запропонували покращення алгоритму QEM, що дозволяє уникати недоліку, пов’язаного з підвищенням пріоритету граничних ребер. Дослідниками було запропоновано враховувати значення кривизни граничної кривої для розрахунку QEM площини граничного обмеження. Кривизну граничної кривої пропонується обчислювати наступним способом. Якщо розглядати многовидну модель, то можна припустити, що кожна гранична вершина буде мати рівно дві сусідні граничні вершини. Розглянемо граничну вершину і її сусідні граничні вершини і . Обчислимо кривизну граничної кривої для вершини як:

|  |  |
| --- | --- |
|  | (1.6) |

де 𝑘 – кривизна граничної кривої для .

Для немноговидних поверхонь припущення про точність наявності двох сусідніх граничних вершин може бути невірним. Отже, кривизна, розрахована з урахуванням будь-якої пари сусідніх граничних вершин, призведе до неправильного судження про вершину. Щоб обробляти такі випадки, пропонується визначати немноговидні граничні вершини як складні вершини і не спрощувати їх, накладаючи на них штраф. Отримане значення кривизни слід використовувати як ваговий коефіцієнт для QEM площини граничного обмеження кожної вершини відповідно. Ця схема зважування на основі кривизни для збереження границь надає більшу вагу граничним вершинам з високою кривизною. Це дозволяє уникнути зміщення алгоритму в сторону граничних вершин.

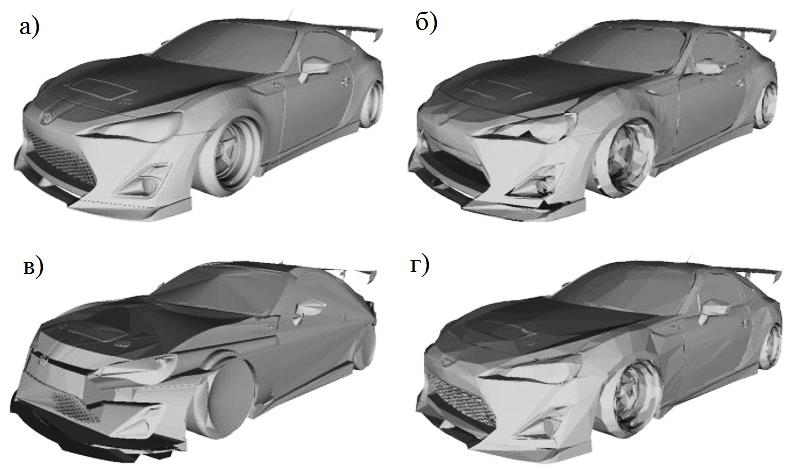


а) оригінальна модель (20544 грані); б) спрощена модель за алгоритмом QEM з граничними обмеженнями (3236 граней); в) спрощена модель за алгоритмом QEM з урахуванням кривизни (3236 граней);

Рисунок 1.4 – Порівняння результатів роботи алгоритму QEM з граничними обмеженнями та алгоритму QEM з урахуванням кривизни

Іншим обмеженням оригінального алгоритму QEM є те, що він не враховує властивості поверхні, такі як нормалі, колір і текстурні координати. Гарланд і Хекберт [9] узагальнили оригінальний алгоритм QEM, щоб також обробляти властивості поверхні. Для обробки нормалей та кольорів матриця помилок 4 x 4 може бути розширена до матриці 6 x 6, яка містить помилку нормалі вершини та її кольору. Аналогічно, матриця помилок 5 × 5 може використовуватися для фіксації помилки текстурних координат вершини. Однак цей підхід розглядає тільки той випадок, коли вся модель порівнюється з одним текстурованим матеріалом, так що між вершинами і координатами текстури існує взаємно однозначна відповідність. Але на практиці кілька текстурних матеріалів можуть бути пов'язані з однією 3D-моделлю. Крім того, декільком значенням з однієї текстури також може бути присвоєна задана вершина. Отже, модель може складатися з вершин з декількома наборами текстурних координат. Для обробки цих випадків Бахірат та ін. [10] запропонували створити нову структуру даних для кожної вершини, яка містить в собі інформацію про сусідні грані і відповідні текстурні координати, пов'язані з даною вершиною. Ці структури оновлюються під час видалення ребер. Також було розширено підхід збереження границь, щоб зберігати ребра, які утворюють границю між декількома текстурами. Ребра, пов'язані з декількома текстурами, являють собою матеріальні границі. Якщо модель має підмножину граней, які не мають інформації про властивості поверхні, їх значення встановлюються рівними нулю. Це гарантує, що порядок видалення відповідних ребер ґрунтується виключно на геометричній помилці, а не на властивостях поверхні, яких немає. Щоб визначити межі матеріалу, використовується вищезгадана структура, яка містить список граней в околиці. Якщо будь-які дві грані в цьому списку мають різний матеріальний індекс, можна вважати, що даній вершині присвоєні два різних матеріали, отже, її можна ідентифікувати як матеріальну граничну вершину. Крім того, якщо будь-яка вершина пов'язана з декількома текстурними значеннями, то вона вважається критичною вершиною. Потім до квадрик, пов'язаних з вершинами на матеріальних границях, застосовується визначений користувачем ваговий коефіцієнт. Таким чином, матеріальні границі залишаються незмінними, а матеріальні показники належним чином зберігаються.

Розглянемо порівняння роботи алгоритмів QEM на прикладі многовидної моделі з границями (рис. 1.5).



а) оригінальна модель; б) спрощена модель за алгоритмом QEM з граничними обмеженнями в) спрощена модель за оригінальним алгоритмом QEM г) спрощена модель за алгоритмом QEM з урахуванням кривизни

Рисунок 1.5 – Порівняння роботи різних модифікацій алгоритму QEM на прикладі многовидної моделі з границями

Як видно з рисунка 1.5, для многовидної моделі з границями найгірший результат дає оригінальний алгоритм. Алгоритм з граничними обмеженнями забезпечує краще збереження границь, а алгоритм з урахуванням кривизни дозволяє зберегти ще більше деталей.

На основі проведеного літературного огляду проблеми спрощення тріангуляційних поверхонь можемо зробити висновок, що в основному алгоритми дають хороший результат на моделях певного типу. Тому при виборі методу слід виходити з того, яка саме перед дослідником стоїть задача. Наприклад, якщо треба отримати дуже якісне зображення немноговиду, і при цьому час розробки алгоритму і час виконання необмежені, слід звернути увагу на алгоритм спрощування поверхні за допомогою енергетичної функції. Якщо навпаки, вхідна модель може бути будь-якого типу та дуже важливо, щоб спрощення відбувалося за якомога менший час, а деталізація при цьому не є важливим критерієм, то в даному випадку можна порекомендувати скористатися методом кластеризації вершин. Виходячи з умов задачі, яка була поставлена перед нами, можна виділити найбільш оптимальний алгоритм з точки зору якості спрощеної моделі, часу розробки алгоритму, швидкості його роботи, а також варіативності вхідних даних. Під ці критерії найбільше підходить алгоритм QEM. Він достатньо швидкий, тому що весь розрахунковий етап проходить на етапі ініціалізації, та при цьому дає хороший результат; також він дозволяє обробляти моделі будь-якого типу.

СПИСОК ДЖЕРЕЛ ІНФОРМАЦІЇ

1. H. Hoppe, T. DeRose, T. Duchamp, J. McDonald, and W. Stuetzle, “Mesh optimization,” in ACM Annual Conference on Computer Graphics and Interactive Techniques (SIGGRAPH), 1993, pp. 19-26.
2. H. Hoppe, “Progressive meshes,” in ACM Annual Conference on Computer Graphics and Interactive Techniques (SIGGRAPH), pp. 99-108, 1996.
3. G. Turk, “Re-tiling polygonal surfaces,” ACM SIGGRAPH Computer Graphics, vol. 26, no. 2, pp. 55-64, 1992.
4. T. He, L. Hong, A. Kaufman, A. Varshney, and S. Wang, “Voxel based object simplification,” in IEEE Conference on Visualization, 1995, pp. 296-303.
5. W. E. Lorensen, and H. E. Cline, “Marching cubes: A high resolution 3D surface construction algorithm,” ACM SiGGRAPH Computer Graphics, vol. 21, no. 4, pp. 163-169, 1987.
6. W. J. Schroeder, J. A. Zarge, and W. E. Lorensen, “Decimation of triangle meshes,” in ACM SIGGRAPH Computer Graphics, vol. 26, no. 2, pp. 65-70, 1992.
7. J. Rossignac, and P. Borrel, “Multi-resolution 3D approximations for rendering complex scenes,” in Modeling in Computer Graphics, 1993, pp. 455-465.
8. M. Garland, and P. S. Heckbert, “Surface simplification using quadric error metrics,” in ACM Annual Conference on Computer Graphics and Interactive Techniques (SIGGRAPH), 1997, pp. 209-216.
9. M. Garland, and P. S. Heckbert, “Simplifying surfaces with color and texture using quadric error metrics,” in IEEE Conference on Visualization, 1998, pp. 263-269.
10. K. Bahirat, C. Lai, R. P. McMahan, and B. Prabhakaran, “A Boundary and Texture Preserving Mesh Simplification Algorithm for Virtual Reality,” in ACM Multimedia Systems Conference on Multimedia Systems, 2017, pp. 50-61.