Министерство образования и науки Российской Федерации
Санкт-Петербургский политехнический университет Петра Великого
Институт компьютерных наук и кибербезопасности
Высшая школа программной инженерии

Работа допущена к защите
Директор ВШПИ
_____ П. Д. Дробинцев
«_____» _____ 2024 г.

ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА
работа бакалавра

СРАВНЕНИЕ МЕТОДОВ СЖАТИЯ АТРИБУТОВ ОБЛАКОВ ТОЧЕК

по направлению подготовки (специальности)
09.03.04 Программная инженерия

Направленность (профиль)
09.03.04_1 «Технология разработки и сопровождения качественного программного продукта»

Выполнил студент гр. 5130904/00104 — Поздняков А. А.

Руководитель старший преподаватель — Фёдоров С. А.

Консультант по нормоконтролю — Локшина Е. Г.

Санкт-Петербург
2024

САНКТ-ПЕТЕРБУРГСКИЙ ПОЛИТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ
ПЕТРА ВЕЛИКОГО

Институт компьютерных наук и технологий

Высшая школа программной инженерии

УТВЕРЖДАЮ

Директор ВШПИ

_____ П.Д. Дробинцев

”18” апреля 2024 г.

ЗАДАНИЕ

на выполнение выпускной квалификационной работы

студенту Позднякову Артемию Анатольевичу, группа 5130904/00104

1. Тема работы: Сравнение методов сжатия атрибутов облаков точек
2. Срок сдачи студентом законченной работы: 20.05.2024
3. Исходные данные по работе:
    - Документация на язык программирования Python;
    - Документация к библиотеке Open3D;
    - Документация к библиотеке NumPy;
4. Содержание работы (перечень подлежащих разработке вопросов):
    - Обоснование актуальности работы
    - Обзор существующих решений
    - Составление требований к разрабатываемому решению
    - Описание реализации
    - Анализ результатов
5. Перечень графического материала (с указанием обязательных чертежей):
6. Консультанты по работе:
7. Дата выдачи задания: 18.04.2024

Руководитель ВКР _____ Фёдоров С. А.

Задание принял к исполнению 18.04.2024

Студент _____ Поздняков А. А.

РЕФЕРАТ

Бакалаврская работа посвящена разработке подхода к оценке различных методов сжатия атрибутов облаков точек на платформе PCCArena. Дан обзор существующих алгоритмов и распространённых кодеков с открытым исходным кодом. В результате сравнительного анализа алгоритмов получены данные об эффективности (!!!) отобранных алгоритмов сжатия атрибутов облаков точек при различных данных и входных параметрах.

В рамках работы была разработана программа для вычисления качественных метрик сжатия облаков точек с использования языка Python. Предложенное решение было внедрёно в существующую платформу оценки методов сжатия облаков точек PCCArena.

ABSTRACT

Nam dui ligula, fringilla a, euismod sodales, sollicitudin vel, wisi. Morbi auctor lorem non justo. Nam lacus libero, pretium at, lobortis vitae, ultricies et, tellus. Donec aliquet, tortor sed accumsan bibendum, erat ligula aliquet magna, vitae ornare odio metus a mi. Morbi ac orci et nisl hendrerit mollis. Suspendisse ut massa. Cras nec ante. Pellentesque a nulla. Cum sociis natoque penatibus et magnis dis parturient montes, nascetur ridiculus mus. Aliquam tincidunt urna. Nulla ullamcorper vestibulum turpis. Pellentesque cursus luctus mauris.

Nulla malesuada porttitor diam. Donec felis erat, congue non, volutpat at, tincidunt tristique, libero. Vivamus viverra fermentum felis. Donec nonummy pellentesque ante. Phasellus adipiscing semper elit. Proin fermentum massa ac quam. Sed diam turpis, molestie vitae, placerat a, molestie nec, leo. Maecenas lacinia. Nam ipsum ligula, eleifend at, accumsan nec, suscipit a, ipsum. Morbi blandit ligula feugiat magna. Nunc eleifend consequat lorem. Sed lacinia nulla vitae enim. Pellentesque tincidunt purus vel magna. Integer non enim. Praesent euismod nunc eu purus. Donec bibendum quam in tellus. Nullam cursus pulvinar lectus. Donec et mi. Nam vulputate metus eu enim. Vestibulum pellentesque felis eu massa.

Quisque ullamcorper placerat ipsum. Cras nibh. Morbi vel justo vitae lacus tincidunt ultrices. Lorem ipsum dolor sit amet, consectetuer adipiscing elit. In hac habitasse platea dictumst. Integer tempus convallis augue. Etiam facilisis. Nunc elementum fermentum wisi. Aenean placerat. Ut imperdiet, enim sed gravida sollicitudin, felis odio placerat quam, ac pulvinar elit purus eget enim. Nunc vitae tortor. Proin tempus nibh sit amet nisl. Vivamus quis tortor vitae risus porta vehicula.

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

Облака точек. Растущая популярность (?ссылка?) технологий компьютерного зрения (CV - Computer Vision) и расширенной реальности (XR - eXtended Reality) влечёт за собой потребность в способах компактного хранения и передачи трёхмерных (далее 3Д) данных. 3Д-данные представляются в виде полигональных сеток - совокупности вершин, рёбер и граней, определяющих форму трёхмерного объекта, или облаков точек, отличающихся от последних отсутствием связей между вершинами. Дополнительно, данные о геометрической структуре объекта могут быть дополнены информацией о его внешних характеристиках.

Важной задачей является создание цифрового представления реальных объектов и сцен. 3Д-сканирование - технология, позволяющая считывать форму физического объекта и его внешних характеристик, таких как цвет или отражающая способность поверхности. В общем случае, результатом процесса 3Д-сканирования является конечное множество точек в трёхмерном пространстве[7, с. 10].

Полигональные сетки аппроксимируют непрерывную поверхность исходного объекта. В свою очередь, облака точек сохраняют мельчайшие подробности структуры поверхности объекта вплоть до миллиметра[7, с. 33]. Задача получения 3Д-модели объекта в виде полигональной сетки по имеющемуся облаку точек, считанных с его поверхности, решается методами реконструкции поверхности[7]. Для обратного преобразования достаточно удалить связи между вершинами в полигональной сетке.

Большая точность позволяет использовать облака точек для машинной обработки в системах компьютерного зрения, взаимодействующих с реальным миром. Примером подобных систем являются беспилотные автомобили, использующие лидары - локаторы, испускающие световые волны оптического

диапазона с дальнейшей регистрацией отраженных импульсов[11, с. 7]. Другим примером являются системы расширенной реальности, в данном случае облака точек используются для совмещения позиций виртуальных объектов и физических объектов, находящихся рядом с человеком[11, с. 15].

Для точного описания поверхности, облака точек должны быть достаточно плотными. Точки в облаке представляют собой дискретные образцы непрерывной поверхности, а полигональные сетки аппроксимируют данную поверхность полигонами[11, с. 4]. Всё это влечёт за собой большой размер облаков точек (может тут тоже ссылка?), в связи с чем возникает задача разработки способов компактного хранения подобных объектов.

Алгоритмы сжатия облаков точек решают задачу компактного хранения и передачи облаков точек. Популярные кодеки были реализованы в рамках проектов Draco[3] и PCL[20]. В настоящее время различными авторами предлагаются новые методы сжатия геометрической структуры облаков точек[13][19] и их атрибутов[6][16][8]. Алгоритм кодирования данных в этих решениях можно разделить на этапы реконструкции изначальной геометрической структуры объекта и кодирования атрибутов облака в соответствии с полученной структурой. Работа по стандартизации PCC-кодеков была начата MPEG в 2017 году[12], также данной группой был предложен собственный кодек и разработана тестовая модель на его основе - TMC13[4].

Большое количество постоянно появляющихся подходов к сжатию облаков точек делает актуальной задачу разработки программы для оценки работы PCC-кодеков. Подобная программа может быть использована исследователями для подсчёта метрик разрабатываемых ими кодеков.

Цель работы - разработка подхода к сравнению алгоритмов сжатия атрибутов облаков точек. В рамках данной работы необходимо решить следующие задачи:

- Проанализировать существующие PCC-кодеки
- Изучить релевантные метрики, отображающие эффективность (??) / ка-

чество (??) сжатия атрибутов облаков точек

- Разработать программу подсчёта метрик
- Получить метрики для отобранных PCC-кодеков
- Проанализировать результаты работы

ГЛАВА 1. ОБЗОР ПОДХОДОВ К СРАВНЕНИЮ МЕТОДОВ СЖАТИЯ ОБЛАКОВ ТОЧЕК

В общем случае, информация может быть сжата, если она является избыточной. Сжатие без потерь основано на уменьшения избыточности информации. В сжатии с потерями вводится новое понятие - нерелевантная информация, то есть такая информация, которая слабо влияет на восприятие изображения человеком. Сжатие с потерями, таким образом, основано не только на уменьшении избыточности, но и на определении нерелевантной информации и уменьшении количества подобной информации[1, с. 265].

В случае, когда визуальная информация предназначена прежде всего для восприятия человеком, единственной корректной метрикой визуального качества изображения является субъективная оценка некоторой отобранной группой людей[10]. Данный подход, однако, не является машинным и, следовательно, не подлежит автоматизации. Для решения данной проблемы были разработаны метрики, помогающие автоматически предсказать *воспринимаемое качество (perceived quality)* изображения. К подобным метрикам относится метрики качества изображения SSIM и VIF(Visual Image Fidelity).

1.1. Подходы к оценке качества сжатия изображений

Потребность в оценке качества сжатия визуальной информации возникла после изобретения методов сжатия подобной информации. Многие из качественных метрик сжатия 3Д-данных, были изначально разработаны для оценки качества сжатия обычных изображений.

Введем понятие реконструированного изображения. Пусть оригинальное изображение A было закодировано в формате F_1. Закодируем оригинальное изображение с использованием алгоритма сжатия и получим сжатое изображение B, закодированное в формате F_2. Теперь произведем декомпрессию изображения B и закодируем его с помощью формата F_1. Полученное изображение

C в формате F_1 - реконструированное изображение.

Метрики качества для медиа могут быть классифицированы согласно тому, с чем сравнивается реконструированное изображение[10]:

- С оригинальным, неискаженным изображением
- С набором признаков, извлеченных из оригинального изображения
- С самим собой (!!!)

Другая классификация предполагает разделение метрик по признаку того, на чём они основаны[15, с. 6]:

- Математически-обоснованные метрики, учитывающие лишь интенсивность искажения. К подобным метрикам относятся среднеквадратичная ошибка (MSE) и пиковое отношение сигнал-шум (PSNR)
- Низкоуровневые метрики, которые учитывают видимость искажений, как, например функции чувствительности контраста (CSF)
- Высокоуровневые метрики, которые основаны на гипотезе о том, что человеческое зрение адаптировано к извлечению структурной информации из изображения. К подобным метрикам относится индекс структурной похожести (SSIM) и визуальная точность изображения (VIF)

Математически-обоснованные метрики качества

Одной из стандартных метрик качества сжатия изображений является пиковое отношение сигнал-шум (Peak Signal to Noise Ratio - далее PSNR). Данная метрика достаточно проста для вычисления, однако она имеет лишь ограниченное, приблизительное отношение к ошибкам, которые воспринимает человеческий глаз. Иными словами, большее значение PSNR означает меньшее расхождение между оригинальным и сжатым изображением, но не гарантирует, что зрителям понравится реконструированное изображение[1, с. 279].

Обозначим пиксели исходного изображения как P_i, а пиксели реконструированного изображения как Q_i (где $0 \leq i \leq n$). Для начала определим понятие

среднеквадратичной ошибки (Mean Squared Error - далее MSE) между двумя изображениями как

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(P_i - Q_i)^2 \tag{1.1}$$

Иными словами, среднеквадратичная ошибка для изображений является суммой квадратов ошибок для каждого из пикселей. Тогда метрика PSNR может быть определена как

$$\text{PSNR} = 10\log_{10}\frac{\max_i |P_i|^2}{MSE} \tag{1.2}$$

При этом $\max_i |P_i|$ - пиковое значение сигнала. Для чёрно-белого изображения пиковым значением является 1, для изображения в оттенках серого при глубине 8 бит на пиксель данное значение равно 255. Так как используется логарифм отношения, результирующее значение измеряется в децибелах.

Метрика PSNR имеет значение лишь в сравнении с показателями данной метрики для того же (или схожего) кодека и тех же входных данных при отличных входных параметрах[9]. Несмотря на это, данная метрика продолжает использоваться исследователями для сравнения качества работы существующих кодеков с предлагаемыми ими решениями[5].

Другой похожей метрикой является отношение сигнала к шуму (Signal to Noise Ratio - далее SNR). В данном случае, рассматривается не пиковое значение сигнала, а среднеквадратичное.

$$\text{SNR} = 10\log_{10}\frac{\frac{1}{n}\sum_{i=1}^{n}P_i^2}{MSE} \tag{1.3}$$

Значение отношения сигнала к шуму квантования (Signal to Quantization Noise Ratio - далее SQNR) представляет собой меру влияния квантования на качество сигнала. Данную метрику можно определить как отношение мощности сигнала к разнице между сигналом после квантования и оригинальным значе-

нием сигнала.

$$\text{SQNR} = 10 \log_{10} \frac{\text{signal power}}{\text{quantization error}} \tag{1.4}$$

Высокоуровневые метрики качества

Математически-обоснованные метрики основаны на предположении, что уменьшение воспринимаемого качества изображения напрямую связано с величиной шума. Так, например, MSE даёт объективную оценку мощности (?) шума, однако два зашумленных изображения с одинаковым значением MSE могут ошибки разного рода, некоторые из которых являются более заметными чем другие. Для решения данной проблемы были предложены высокоуровневые метрики качества изображений.

Рассмотрим индекс структурной похожести (SSIM), данная метрика была предложена в качестве способа предсказания воспринимаемого качества изображения. Она представляет собой более точную и соответствующую человеческому восприятию метрику для оценки качества сжатия изображений, поскольку учитывает структурные и текстурные аспекты изображения, в то время как MSE и PSNR ориентированы на разницу в значениях пикселей без учета визуальных особенностей.

SSIM (Structural Similarity Index)

Вычисляется сравнением трех основных аспектов изображений: яркости (luminance), контрастности (contrast) и структуры (structure). Вот формула для вычисления SSIM:

$$\text{SSIM}\,(x, y) = \frac{(2\mu_x\mu_y + c_1)\,(2\sigma_{xy} + c_2)}{\left(\mu_x^2 + \mu_y^2 + c_1\right)\left(\sigma_x^2 + \sigma_y^2 + c_2\right)} \tag{1.5}$$

где:

- x и y - сравниваемые изображения

- μ_x и μ_y - средние значения пикселей изображений x и y
- σ_x^2 и σ_y^2 - дисперсии значений пикселей изображений x и y
- σ_{xy} - ковариация между значениями пикселей изображений x и y
- c_1 и c_2 - константы для обеспечения устойчивости деления

Этот индекс обычно принимает значения от -1 до 1, где 1 указывает на идеальное сходство между изображениями, а значения ближе к -1 указывают на более сильные различия.

SSIM сравнивает локальные окрестности пикселей изображений, а не их абсолютные значения. Он учитывает не только яркость и контрастность, но и структуру изображения, что делает его более подходящим для оценки качества изображений с точки зрения человеческого восприятия

VIF (Visual Information Fidelity)

VIF обеспечивает более точную оценку качества сжатия, так как учитывает не только структурные аспекты изображения, но и сложные визуальные особенности, такие как текстуры, края и детали. Это делает VIF более подходящей метрикой для оценки реального восприятия качества изображения человеком.

Вычисление Visual Information Fidelity (VIF) включает несколько этапов, включающих оценку сходства между двумя изображениями с учетом их визуальной информации. Вот общий алгоритм вычисления VIF:

- Разбиение изображений на блоки: Сначала изображения разбиваются на небольшие блоки пикселей. Обычно используются квадратные блоки определенного размера.
- Вычисление локальных статистических параметров: Для каждого блока изображения вычисляются локальные статистические параметры, такие как среднее значение, дисперсия и ковариация пикселей.
- Вычисление локальных масштабных параметров: Для каждого блока изображения также вычисляются локальные масштабные параметры, которые оценивают структурную информацию в блоке.

- Вычисление глобальных статистических параметров: Глобальные статистические параметры вычисляются на основе суммирования или усреднения локальных параметров по всем блокам изображений.
- Вычисление VIF: Затем производится расчет VIF, используя локальные и глобальные параметры. В общем, VIF представляет собой взвешенную сумму сходства между локальными статистическими параметрами двух изображений.
- Нормализация VIF: Иногда VIF может быть нормализован для получения значения в диапазоне от 0 до 1, где 1 указывает на идеальное сходство между изображениями.

Это общий алгоритм, и реальная реализация VIF может включать дополнительные детали и оптимизации. Однако основная идея заключается в том, чтобы учитывать как локальные, так и глобальные статистические параметры изображений для оценки их визуального сходства.

В целом, VIF обеспечивает более глубокую и точную оценку качества сжатия изображений, учитывая разнообразные визуальные особенности и взаимосвязи между блоками изображения, что делает его ценным инструментом при сравнении различных методов сжатия изображений.

1.2. Подходы к оценке качества сжатия облаков точек

Идея многих подходов к оценке качества сжатия облаков точек взята из более глубоко проработанной области оценки качества изображений и видео. Облака точек, однако представляют собой гораздо более сложный объект, что влияет не только на то, какие стандартные математически-обоснованные качественные метрики можно к ним применить, но и на сам процесс вычисления данных метрик. Рассмотрим терминологию, связанную с наличием потерь при сжатии облаков точек[12]:

- Геометрическая структура с потерями - декодированная сжатая геометрическая структура не обязательно численно совпадает с изначальной гео-

метрической структурой. Количество точек в декодированном облаке также может не совпадать с количеством точек в изначальном облаке

- Геометрическая структура без потерь - декодированная сжатая геометрическая структура численно совпадает с изначальной геометрической структурой в отношении значений (x, y, z). Количество точек в декодированном облаке также совпадает с количеством точек в изначальном облаке
- Атрибуты с потерями - декодированные сжатые атрибуты не обязательно численно совпадают с изначальными атрибутами
- Атрибуты без потерь - декодированные сжатые атрибуты полностью численно совпадают с изначальными атрибутами

Очевидно, что метрики качества релевантны только для методов сжатия геометрической структуры с потерями и методов сжатия атрибутов с потерями. Рассмотрим, как некоторые из вышеупомянутых метрик можно адаптировать для оценки качества сжатия облаков точек.

Искажение геометрической структуры можно оценить, вычислив среднеквадратичную ошибку искажения. Для этого необходимо определить, что собой представляет ошибка для единичной точки. Точку оригинального облака назовём оригинальной точкой, данная точка в ходе сжатия с потерями и последующей декомпрессии определенным образом преобразуется, соответствующую точку в реконструированном облаке назовём реконструированной точкой. Возможно несколько случаев:

1. Координаты оригинальной и реконструированной точки полностью совпадают
2. Координаты оригинальной и реконструированной точки отличаются незначительно
3. Реконструированной точки не существует. Сжатия с потерями геометрической структуры не гарантирует совпадение количества точек в ориги-

нальном и реконструированном облаке, соответственно, исходная точка может быть редуцирована или заменена несколькими другими точками

Для вычисления ошибки выбирается точка реконструированного облака, наиболее близкая к координатам оригинальной точки. В случаях 1 и 2 данной точкой будет реконструированная точка, в случае 3 некоторая другая точка. Ошибка представляет собой расстояние от оригинальной точки до наиболее близкой к её координатам точки реконструированного облака. Исходя из данного определения возможно вычислить среднеквадратичную ошибку и пиковое отношение сигнал-шум. Обозначим исходные точки как $x_1, x_2, \ldots, x_n$, а реконструированные как $x'_1, x'_2, \ldots, x'_n$, определим среднеквадратичную ошибку как

$$\text{MSE} = \frac{1}{n} \sum_{i=0}^{n} (x_i - x'_i)^2 \tag{1.6}$$

Примем за пиковое значения сигнала максимальное расстояние от точки до её ближайшего соседа внутри оригинального облака. Тогда пиковое отношение сигнал-шум можно определить как

$$\text{PSNR} = 10 \log_{10} \frac{(\text{max distance})^2}{\text{MSE}} \tag{1.7}$$

Высокоуровневые метрики для оценки качества изображений также возможно адаптировать для оценки облаков точек. Для этого необходимо произвести рендеринг облака и получить одно или несколько обычных изображений данного облака. Далее, по полученным изображениям возможно рассчитать высокоуровневые метрики качества, такие как SSIM и VIF. Указанный подход целесообразен в случае, когда данные предназначены для восприятия человеком, например, при использовании облака для рендеринга или в системах расширенной реальности.

1.3. Системы оценки качества сжатия облаков точек

Исследователями были разработаны различные системы оценки качества сжатия облаков точек. Рассмотрим принципы их работы и выделим метрики, которые могут быть подсчитаны с помощью данных систем.

mpeg-pcc-dmetric

Данная система была разработана MPEG в рамках работы по стандартизации методов сжатия облаков точек. MPEG выделяет следующие категории облаков точек[12]:

- Статические объекты и сцены
- Динамические объекты
- Динамически считанные объекты и сцены

Категории 1 и 3 были объединены и работа над стандартизацией в данной области ведётся в рамках проекта G-PCC. Работа над стандартизацией методов, работающих с облаками категории 2 ведётся в рамках проекта V-PCC. MPEG также были предложены собственные методы для сжатия облаков точек и разработаны тестовые модели, реализующие данные кодеки, тестовая модель проекта V-PCC имеет название TMC2, тестовая модель проекта G-PCC имеет название TMC13(ссылка??). Для качественной оценки работы данных тестовых моделей MPEG была разработана утилита mpeg-pcc-dmetric, представляющая собой консольное приложение, принимающее на вход два облака точек - оригинальное и реконструированное, и вычисляющая определенные показатели искажения геометрической структуры и атрибутов для данных облаков.

Листинг 1.1: Пример использования утилиты mpeg-pcc-dmetric, параметр fileA - оригинальное облако, fileB - реконструированное облако, color=1 - также вычисляются значения искажения цветов.

```
1 ./pc_error --fileA=./oskull.ply --fileB=./pskull.ply --color=1
```

На листинге 1.1 приведен пример использования данной утилиты. В дан-

ном случае, в итоговый отчёт работы утилиты будет включено минимальное и минимальное и максимальное расстояние до ближайшей точки в оригинальном облаке, среднеквадратичная ошибка точек, среднеквадратичная ошибка для каждой компоненты цвета, а также пиковое отношение сигнал-шум для точек и каждой компоненты цвета.

Данное программное обеспечивание поддерживается MPEG, однако его исходный код, как и исполняемые файлы, не находятся в открытом доступе. Для доступа к данному ПО необходимо отправить запрос MPEG, при этом доступ выдаётся лишь для некоммерческого использования и только в исследовательских целях. Данные обстоятельства делают невозможным использование данной утилиты в разработке проприетарных кодеков, а непрозрачная система выдачи доступов усложняет проведение независимых исследований.

GeoCNNv1 geo_dist

Авторами кодека GeoCNNv1(ссылка?) была разработана собственная система для подсчета качественных метрик. Исходный код утилиты опубликован на Github(ссылка?) и написан на языке C++ с использованием библиотеки PCL(ссылка?).

Листинг 1.2: Пример использования утилиты geo_dist, параметр a - оригинальное облако, b - реконструированное облако.

```
pc_error -a test/sphere100.pcd -b test/noisy_sphere100.pcd
```

На листинге 1.2 приведен пример использования данной утилиты, в результате работы программы вычисляются лишь искажения геометрической структуры облака точек. Также как и в утилите от MPEG, данная утилита вычисляет минимальное и максимальное расстояние до ближайшего соседа в оригинальном облаке, корень среднеквадратичной ошибки и соответствующее пиковое отношение сигнал-шум. Важным отличием является то, что данная утилита способна вычислять указанные метрики не только только для случая точка-точка, но и для случая точка-плоскость (это вообще что??).

Таким образом, данная система обладает большей функциональностью с точки зрения вычисления геометрического искажения облака, однако не позволяет вычислять искажение цветов и других атрибутов. Несмотря на то, что данная программа имеет открытый исходный код, данная кодовая база не поддерживается разработчиками, так как во второй версии кодека - GeoCNNv2 (ссылка?), ими было принято решение использовать для оценки качества утилиту mpeg_pcc_dmetric. Другим важным обстоятельством является отсутствие лицензии у данного проекта (ссылка?), в таком случае к исходному коду применяются стандартный копирайт, который не даёт другим разработчикам права модифицировать и распространять данное ПО. Иными словами, несмотря на доступность исходного кода ПО в публичном доступе, оно не является ПО с открытым исходным кодом(ссылка?).

PCCArena

Другими исследователями(ссылка?) была предложена комбинированная система для сравнения различных методов сжатия облаков точек. Данная система использует mpeg_pcc_dmetric для оценки искажения геометрической структуры и атрибутов облака точек. Также, в PCCArena была адаптирована решение VMAF от Netflix, предназначенное для оценки качества сжатия изображений и видео с помощью высокоуровневых метрик, а также некоторых методов машинного обучения(ссылка?). Облако точек проходит через рендеринг, в результате которого и получается изображение, которое может быть оценено системой VMAF.

Система имеет открытый исходный код, имеющий лицензию MIT, что позволяет использовать свободно распространять и модифицировать данное ПО, а также использовать в коммерческих нуждах. Однако, для запуска программы необходим исполняемый файл mpeg_pcc_dmetric, так как данная утилита не включена в исходный код системы.

1.4. Новая система оценки качества сжатия облаков точек

Рассмотренные системы оценки качества сжатия имеют различные достоинства и недостатки.

Таблица 1: Метрики, вычисляемые различными рассмотренными системами.

	mpeg_pcc_dmetric	geo_dist
$d_{\max}$	+	+
$d_{\min}$	+	+
$\mathrm{MSE}_{\mathrm{geometry}}$	+	+
$\mathrm{MSE}_{\mathrm{attribute}}$	+	-
$\mathrm{PSNR}_{\mathrm{geometry}}$	+	+
$\mathrm{PSNR}_{\mathrm{attribute}}$	+	-
Точка-точка	+	+
Точка-плоскость	-	+

Таблица 2: Характеристики различных рассмотренных систем.

	mpeg_pcc_dmetric	geo_dist
Полнота метрик	+-	+-
Оценка искажения атрибутов	+	-
Поддерживаемость	+	-
Возможность расширения	-	-
Открытый исх. код	-	-

В таблицах 1 и 2 приведена сравнительная характеристика рассмотренных систем. Из проведенного анализа можно сделать вывод, что разработка новой системы оценки качества сжатия облаков точек - актуальная задача. К разрабатываемому решению можно сформулировать следующие требования:

- Возможность вычисления стандартных метрик искажения (MSE и PSNR) геометрической структуры и атрибутов облака точек по алгоритму точка-

точка и точка-плоскость

- Использование библиотек, имеющих признание в сообществе разработчиков. Использование стандартных решений вместо написания собственной реализации стандартных функций упрощает разработку, защищает от ошибок, а также делает более простым освоение(??) новыми разработчиками кодовой базы проекта
- Использование технологий разработки качественного программного обеспечения. Документированный код, спроектированный для возможности простого и удобного расширения, использование популярных в сообществе руководств по стилю кода и линтеров также положительно влияет на вовлечение новых разработчиков в работу над проектом с открытым исходным кодом
- Покрытие тестами. Тестирование кода позволяет проверить (верифицировать??) соответствие ПО требованиям, а также безопасно модифицировать его без нарушения существующей функциональности
- Использование лицензии MIT. Данная лицензия позволяет распространять и модифицировать ПО любым разработчикам, а также допускает использование в коммерческих проектах. Благодаря использованию данной лицензии, любой исследователь или организация, разрабатывающая собственный кодек, будут иметь возможность объективно сравнить его показатели с другими существующими кодеками.

ГЛАВА 2. ОБОСНОВАНИЕ ВЫБОРА ТЕХНОЛОГИЙ И СРЕДСТВ РАЗРАБОТКИ

Правильный выбор технологий и средств разработки имеет значительное влияние на простоту и удобство не только самой разработки и дальнейшей поддержки ПО, но и на качественные характеристики данного ПО. Выбор языка программирования и библиотек напрямую влияет на то, какие готовые решения будут доступны разработчику. В свою очередь, использование готовых, протестированных решений в значительной степени защищает программное обеспечение от ошибок, а также удешевляет процесс разработки.

2.1. Язык программирования и библиотеки

Существующие программы, взаимодействующие с облаками точек разрабатываются на языках программирования C++ и Python. При этом, к наиболее популярным библиотекам для работы с облаками точек можно отнести PCL и Open3D, обе библиотеки реализованы на C++, однако Open3D также имеет привязки для языка Python. Принимая во внимание отсутствие строгих требований, касающихся задержек и длительности обработки, для реализации был выбран язык Python (тут не совсем так!!). В качестве математической библиотеки выбран NumPy как стандарт де-факто при реализации математических операций на Python.

Для модульного тестирования был выбран фреймворк pytest. Он широко поддерживается сообществом, обладает большей простым и чистым синтаксисом по сравнению с фреймворком unittest, входящим в стандартную библиотеку Python. Pytest поддерживает генерацию JUnit-отчётов о тестировании, которые поддерживаются и могут быть корректно отображены на популярных платформах, таких как Allure и Gitlab. Для функционального тестирования выбран фреймворк behave, данная библиотека совместима с синтаксисом Cucumber, позволяющим описывать тест-кейсы на естественном языке.

Другой важной составляющей разработки является поддержание единого стиля кода, согласованного с принятыми в сообществе нормами. Для проверки и поддержания стиля кода существуют линтеры, в качества линтера был выбран pylint, а в качестве стиля кода - официальное руководство PEP8. Pylint поддерживает статический анализ типов, что помогает избежать большого количества ошибок при разработке на языке, имеющем динамическую типизацию.

ГЛАВА 3. СИСТЕМА ПОДСЧЁТА МЕТРИК

3.1. Алгоритм поиска реконструированной точки

Алгоритм сопоставления точек

Ранее уже был описан алгоритм, по которому вычисляется среднеквадратичная ошибка и пиковое отношение сигнал-шум по двум облакам точек - оригинальному и реконструированному. На его основе можно сформулировать алгоритм сопоставления точек в двух облаках разной размерности.

Пусть X - итерируемое облако точек, а Y - облако точек поиска, пронумеруем точки в каждом облаке так, что $x_0, x_1, \ldots, x_n$ - точки облака X, где n - количество точек в облаке X, а $y_0, y_1, \ldots, y_m$ - точки облака Y, где m - количество точек в облаке Y. Каждая из точек представляет собой вектор, принадлежащий векторному пространству $\mathbb{R}^3$. Составим облако Y' состоящее из n точек по следующему алгоритму: для каждой точки $x_i \in X$ найдем точку $y_j \in Y$, наиболее близкую к x_i, добавим в облако Y' точку $y'_i = y_j$. В итоге, количество точек в облаках X и Y' будет одинаковым, при этом, для любого $i \in [1, n]$, y'_i будет ближайшим соседом точки x_i в облаке Y'. Облако Y' будем называть *облаком соседей* облака X.

В дальнейшем, метрики, вычисляемые подобным способом будем называть *направленными метриками*. Для каждой метрики, вычисляющейся путем прямого сопоставления точек двух облаков, в зависимости от конкретного способа сопоставления, можно сформулировать следующие 3 значения(ссылка?):

- Левостороннее значение. В данном случае, итерируемым облаком является оригинальное облако, а облаком поиска - реконструированное
- Правостороннее значение. В данном случае, итерируемым облаком является реконструированное облако, а облаком поиска - оригинальное
- Симметричное значение. Данное значение является худшим из двух предыдущих. Если значение метрики прямо (обратно) пропорционально

качеству облака, то берется меньшее (большее) из значений

KD-дерево

Для реализации описанного алгоритма необходимо выбрать алгоритм для поиска ближайшей точки к данной. Для этой задачи может быть использована структура данных KD-дерево (K-Dimensional - имеющее размерность K)[11, с. 23]. Эта структура данных значительно упрощает операцию поиска ближайшего соседа, что является важным, поскольку данная операция должна быть выполнена для каждой точки облака.

KD-дерево представляет собой бинарное дерево, в котором каждый узел-лист является точкой в пространстве размерности k. Рассмотрим алгоритм построения KD-дерева:

1. Если количество точек в области меньше, чем заданное число - размер листа, то данные точки сохраняются в лист дерева, цикл останавливается
2. Последовательность точек в данной области сортируется по x-координате
3. Проводится гиперплоскость, разделяющая последовательность точек пополам и перпендикулярная оси x.
4. Точки, находящиеся левее гиперплоскости, добавляются в область левого поддерева узел, точки, находящиеся правее гиперплоскости - в область правого поддерева
5. Шаги 1-4 повторяются для поддеревьев, при этом ось сортировки меняется. Порядок смены может быть цикличным($x, y, z, x, \dots$) или адаптивным

Далее, построенное дерево можно использовать для эффективного поиска ближайшей точки к заданной по следующему алгоритму:

1. Согласно обычному алгоритму поиска в двоичном дереве, определяется лист, к которому принадлежит заданная точка
2. Последовательным поиском среди точек, хранящихся в узле, определяется ближайшая к заданной точка

3. На основе выбранной ближайшей точки оценивается худшее расстояние
4. В случае, если худшее расстояние меньше, чем расстояние от точки до некоторой разделяющей гиперплоскости, производится повторный поиск в области, отделённой данной гиперплоскостью

Сложность поиска 1 ближайшей точки в сбалансированном KD-дереве составляет $O(\log N)$ в среднем[11, с. 25], что позволяет эффективно (??) производить поиск при построении облака соседей. Таким образом, необходимые структуры данных включают в себя:

- Оригинальное облако точек
- Реконструированное облако точек
- KD-дерево, построенное по реконструированному облаку. Используется для построения облака соседей оригинального облака
- KD-дерево, построенное по оригинальному облаку. Используется для построения облака соседей реконструированного облака
- Облако соседей оригинального облака точек. Используется для вычисления левосторонних значений метрик
- Облако соседей реконструированного облака точек. Используется для вычисления правосторонних значений метрик

3.2. Выбор метрик для оценки качества сжатия атрибутов

Оценка искажения геометрической структуры

Сжатие геометрической структуры облака и сжатие его атрибутов происходит отдельно, однако кодирование атрибутов напрямую зависит от результата сжатия геометрической структуры. Таким образом, для оценки качества сжатия атрибутов облака, также необходимо оценить искажение геометрической структуры облака.

Для данной цели используются метрики, основанные на известных значениях СКО и пикового отношения сигнал-шум[14]:

- Ассиметричное расстояние Чамфера

- Симметричное расстояние Чамфера
- Пиковое отношение сигнал-шум для расстояния Чамфера
- Метрика Хаусдорфа

Ассиметричное расстояние Чамфера является направленной метрикой и имеет левостороннее и правостороннее значения. Имеет следующую формулу:

$$\text{ACD}\left(P_1, P_2\right) = \frac{\sum_{p_1 \in P_1} \min_{p_2 \in P_2} \|p_1 - p_2\|_2^2}{|P_1|} \tag{3.1}$$

где P_1, P_2 - облака точек, $p_1 \in P_1, p_2 \in P_2$ - точки левого и правого облаков соответственно, а $|P_1|$ - количество точек в левом облаке. Легко заметить, что данная формула совпадает с ранее описанной формулой для вычисления СКО(1.6).

Симметричное расстояние Чамфера является средним между левосторонним и правосторонним значением ассиметричного расстояния Чамфера:

$$\text{CD}\left(P_1, P_2\right) = \frac{\left(\text{ACD}\left(P_1, P_2\right), \text{ACD}\left(P_2, P_1\right)\right)}{2} \tag{3.2}$$

Пусть M - максимальный диаметр оригинального облака среди осей x, y, z, диаметром облака D по некоторой оси является расстояние между двумя наиболее удалёнными точками в облаке относительно данной оси

$$\begin{gathered} e_{p_1,p_2} = p_1 - p_2 \\ \text{Proj}_x = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix} \\ D_x = \max_{p_1,p_2 \in P} \|\text{Proj}_x e_{p_1,p_2}\| \\ M = \max\left(D_x, D_y, D_z\right) \end{gathered} \tag{3.3}$$

Примем симметричное расстояние Чамфера CD за значение шума, а M - за пиковое значение сигнала. Тогда пиковое отношение сигнал-шум для расстояния Чамфера может быть выражено следующим образом

$$\text{CD-PSNR} = 10 \log_{10} \frac{M^2}{CD} \tag{3.4}$$

Расстояние Чамфера и производные от него метрики показывают, насколько в среднем ошибается алгоритм. Для того, чтобы оценить ошибку сжатия в худшем случае, необходимо вычислить метрику Хаусдорфа - максимальное расстояние среди всех пар точек в оригинальном и реконструированном облаке

$$\text{HD} = \max \left(\max_{p_1 \in P_1} \left(\min_{p_2 \in P_2} \|p_1 - p_2\|_2^2 \right), \max_{p_2 \in P_2} \left(\min_{p_1 \in P_1} \|p_1 - p_2\|_2^2 \right) \right) \tag{3.5}$$

Оценка искажения нормалей точек

Качество сжатия нормалей напрямую влияет на реконструкцию геометрической структуры и позволяет определить, в какой плоскости лежит каждая конкретная точка. Данные факторы влияют как на рендеринг облака, так и на распознавание объектов методами машинного зрения. Для оценки искажения нормалей будем использовать *проективные метрики*[17]. Данный вид метрик подразумевает проекцию вектора ошибки вдоль нормали в данной точке.

$$\begin{aligned} e^{\text{point}}_{p_1,p_2} &= \|p_2 - p_1\| \\ e^{\text{plane}}_{p_1,p_2} &= \left| \left(e^{\text{point}}_{p_1,p_2}, n_{p_2} \right) \right| \end{aligned} \tag{3.6}$$

При использовании проекции вектора ошибки вдоль нормали вместо обычного вектора ошибки, мы получаем соответствующие аналоги метрик,

описанных ранее: ACD^P, CD^P, CD-PSNR^P, HD^P. Данные метрики позволяют оценить искажение поверхности объекта и не требуют больших вычислительных мощностей.

Оценка искажения цветов

Очевидно, что передача цвета играет большую роль в том, как люди воспринимают объекты. Системы машинного зрения также используют информацию о цвете для распознавания объектов[2]. Для оценки искажения цвета также может быть применено пиковое отношение сигнал-шум, однако важной деталью в данном случае является цветовая схема, которая используется для вычислений.

Семейство цветовых схем YCC более точно соотносится с человеческим восприятием, чем RGB[18, с. 291]. По этой причине, в разрабатываемом решении искажение цвета будем считать не только в цветовой схеме RGB, но и в Y'CbCr. Преобразование будем производить по стандарту ITU-R BT.709, используя следующую операцию

$$\begin{pmatrix} Y' \\ C_B \\ C_R \end{pmatrix} = \begin{pmatrix} 0.2126 & 0.7152 & 0.0722 \\ -0.1146 & -0.3854 & 0.5 \\ 0.5 & -0.4542 & -0.0458 \end{pmatrix} \begin{pmatrix} R' \\ G' \\ B' \end{pmatrix} \tag{3.7}$$

Данное преобразование является достаточно простым, так как требует лишь одной операции умножения матрицы на вектор для каждого значения цвета. Вычисление ошибки в данной цветовой схеме также поддерживается утилитой mpeg-pcc-dmetric. Яркостная компонента Y' в данном случае является наиболее важной, поскольку именно оттенки серого содержат структурную информацию.

Итоговый перечень метрик

Таким образом, включая стандартные метрики СКО и пиковое отношение сигнал-шум, в разрабатываемом решении предлагается вычислять следующие метрики:

Таблица 3: Метрики, вычисляемые в разрабатываемом решении.

	Лев.	Прав.	Симм.	Проец.
$MSE_{коорд}$	+	+	+	+
$PSNR_{коорд}$	+	+	+	+
CD	+	+	+	+
HD	+	+	+	x
MSE_{RGB}	+	+	+	x
$PSNR_{RGB}$	+	+	+	x
$MSE_{Y'CbCr}$	+	+	+	x
$PSNR_{Y'CbCr}$	+	+	+	x

3.3. Архитектура разрабатываемого решения

Вычисляемые метрики образуют достаточно сложный граф зависимостей в сравнении с достаточно небольшой сложностью реализации каждой из них. Из-за этой особенности появляется необходимость в управлении данными зависимостями.

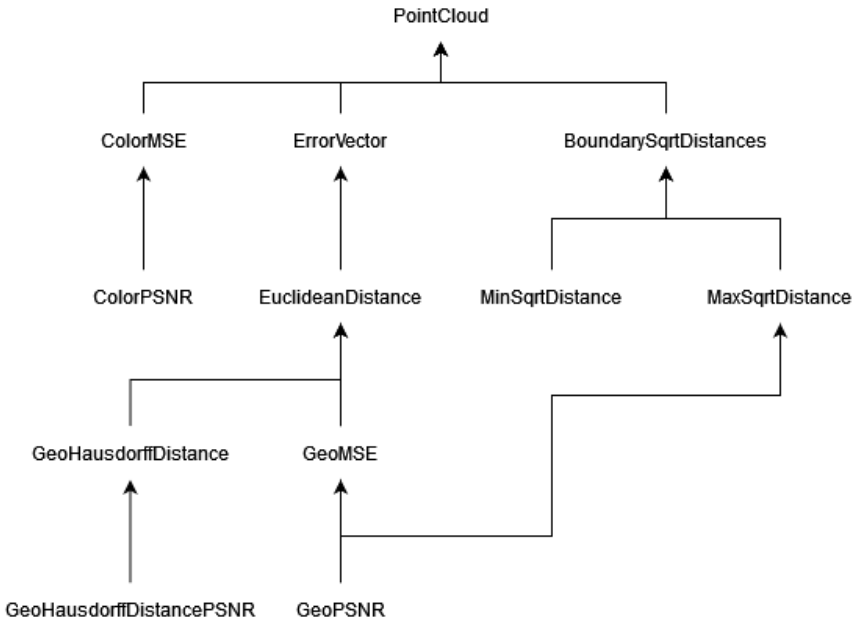

Рис. 3.1: Граф зависимостей вычисляемых метрик

Для решения задачи управления зависимостями в данном проекте был выбран шаблон проектирования ”Одиночка”, этот шаблон подразумевает, что в программе может существовать лишь один экземпляр класса-одиночки. Представим каждую метрику в виде отдельного класса, при этом адаптируем шаблон ”Одиночка”, изменив его основное условие - в каждый момент времени может существовать лишь один экземпляр класса, обладающий данными входными параметрами. В качестве входных параметров выбраны условия ”является левосторонней” и ”является правосторонней” для направленных метрик, а также условия ”точка-точка” или ”точка-плоскость” для проецируемых метрик. При этом все проецируемые метрики также являются направленными, поскольку зависят от значения ошибки, которое является направленным. Таким образом, направленные метрики могут иметь не более 2 экземпляров, а проецируемые - не более 4.

Данное решение, с одной стороны, позволяет избежать повторного вычисления уже полученных значений, а с другой - получить необходимое значение в любом месте кода. Минусом данного решения является наличие глобального состояния, которое может стать причиной гонок при появлении нескольких по-

токов в программе. В случае дальнейшего усложнения программы и появления в ней многопоточности, объекты должны быть защищены мьютексами.

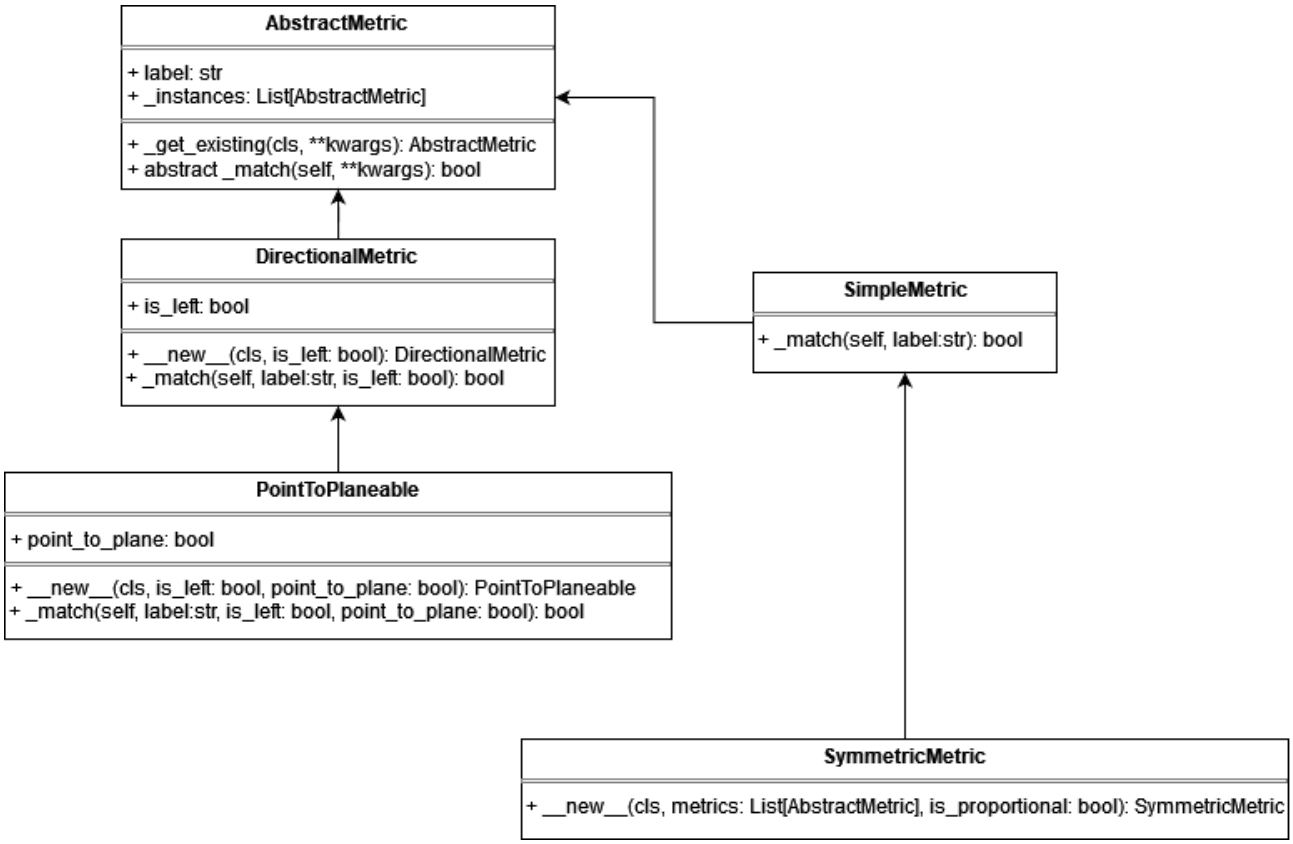

Рис. 3.2: UML-диаграма базовых классов программы

ГЛАВА 4. РЕЗУЛЬТАТЫ РАБОТЫ

ЗАКЛЮЧЕНИЕ

Nam dui ligula, fringilla a, euismod sodales, sollicitudin vel, wisi. Morbi auctor lorem non justo. Nam lacus libero, pretium at, lobortis vitae, ultricies et, tellus. Donec aliquet, tortor sed accumsan bibendum, erat ligula aliquet magna, vitae ornare odio metus a mi. Morbi ac orci et nisl hendrerit mollis. Suspendisse ut massa. Cras nec ante. Pellentesque a nulla. Cum sociis natoque penatibus et magnis dis parturient montes, nascetur ridiculus mus. Aliquam tincidunt urna. Nulla ullamcorper vestibulum turpis. Pellentesque cursus luctus mauris.

СПИСОК ИСТОЧНИКОВ

1. *(auth.) P. D. S.* Data Compression: The Complete Reference. — Springer, 2007. — ISBN 9781846286025; 1846286026; 9781846286032; 1846286034.
2. 2020 Autonomous Vehicle Technology Report [Электронный ресурс]. — URL: https://www.wevolver.com/article/2020.autonomous.vehicle.technology.report (дата обр. 10.05.2024).
3. Draco [Электронный ресурс]. — URL: https://github.com/google/draco (дата обр. 08.05.2024).
4. TMC13 [Электронный ресурс]. — URL: https://github.com/MPEGGroup/mpeg-pcc-tmc13 (дата обр. 08.05.2024).
5. Toward A Practical Perceptual Video Quality Metric [Электронный ресурс]. — URL: https://netflixtechblog.com/toward-a-practical-perceptual-video-quality-metric-653f208b9652 (дата обр. 10.05.2024).
6. Attribute compression of 3D point clouds using Laplacian sparsity optimized graph transform / Y. Shao [и др.] // 2017 IEEE Visual Communications and Image Processing (VCIP). — IEEE, 12.2017. — DOI: 10.1109/vcip.2017.8305131. — URL: http://dx.doi.org/10.1109/VCIP.2017.8305131.
7. *Bellocchio F.* 3D Surface Reconstruction: Multi-Scale Hierarchical Approaches. — 1-е изд. — Springer, 2013. — ISBN 9781461456315; 1461456312; 9781461456322; 1461456320.
8. Cluster-based two-branch framework for point cloud attribute compression / L. Sun [и др.] // The Visual Computer. — 2023. — Нояб. — ISSN 1432-2315. — DOI: 10.1007/s00371-023-03146-9. — URL: http://dx.doi.org/10.1007/s00371-023-03146-9.

9. *Huynh-Thu Q.*, *Ghanbari M.* Scope of validity of PSNR in image/video quality assessment // Electronics Letters. — 2008. — Т. 44, № 13. — С. 800. — ISSN 0013-5194. — DOI: 10.1049/el:20080522. — URL: http://dx.doi.org/10.1049/el:20080522.

10. Image quality assessment: from error visibility to structural similarity / Z. Wang [и др.] // IEEE Transactions on Image Processing. — 2004. — Т. 13, № 4. — С. 600—612. — DOI: 10.1109/TIP.2003.819861.

11. *Kuo(auth.) S. L. Z. K.-C. J.* 3D Point Cloud Analysis: Traditional, Deep Learning, and Explainable Machine Learning Methods. — 1st ed. — Springer, 2021. — ISBN 9783030891794; 3030891798; 9783030891800; 3030891801.

12. *MPEG*. Call for Proposals for Point Cloud Compression V2 : тех. отч. / ISO/IEC. — Hobart, AU, 04.2017. — MPEG2017/N16763.

13. Multiscale Point Cloud Geometry Compression / J. Wang [и др.]. — 2020. — DOI: 10.48550/ARXIV.2011.03799. — URL: https://arxiv.org/abs/2011.03799.

14. PCC arena: a benchmark platform for point cloud compression algorithms / C.-H. Wu [и др.] // Proceedings of the 12th ACM International Workshop on Immersive Mixed and Virtual Environment Systems. — ACM, 06.2020. — (MMSys '20). — DOI: 10.1145/3386293.3397112. — URL: http://dx.doi.org/10.1145/3386293.3397112.

15. *Pedersen M.*, *Hardeberg J. Y.* — 2012. — DOI: 10.1561/0600000037.

16. Point Cloud Attribute Compression via Successive Subspace Graph Transform / Y. Chen [и др.] // 2020 IEEE International Conference on Visual Communications and Image Processing (VCIP). — IEEE, 12.2020. — DOI: 10.1109/vcip49819.2020.9301784. — URL: http://dx.doi.org/10.1109/VCIP49819.2020.9301784.

17. Point Cloud Quality Assessment Using Multi-Level Features / J. Lv [и др.] // IEEE Access. — 2024. — Т. 12. — С. 47755—47767. — ISSN 2169-3536. — DOI: 10.1109/access.2024.3383536. — URL: http://dx.doi.org/10.1109/ACCESS.2024.3383536.

18. *Poynton C. A.* Digital Video and HD. — Oxford, England : Morgan Kaufmann, 10.2001. — (The Morgan Kaufmann Series in Computer Graphics).

19. *Quach M.*, *Valenzise G.*, *Dufaux F.* Improved Deep Point Cloud Geometry Compression. — 2020. — arXiv: 2006.09043 [cs.CV].

20. *Rusu R. B.*, *Cousins S.* 3D is here: Point Cloud Library (PCL) // IEEE International Conference on Robotics and Automation (ICRA). — Shanghai, China : IEEE, 05.2011.

ПРИЛОЖЕНИЕ А. НАЗВАНИЕ ПРИЛОЖЕНИЯ

Текст приложения 1

ПРИЛОЖЕНИЕ Б. НАЗВАНИЕ ПРИЛОЖЕНИЯ