В. В. Санжаров, В. А. Фролов, А. Г. Волобой, В. А. Галактионов, Д. С. Павлов, Система генерации наборов изображений для задач компьютерного зрения на основе фотореалистичного рендеринга, *Препринты ИПМ им. М. В. Келдыша*, 2020, 080

DOI: 10.20948/prepr-2020-80

ИПМ им.М.В.Келдыша РАН • Электронная библиотека

Препринты ИПМ • Препринт № 80 за 2020 г.

ISSN 2071-2898 (Print)
ISSN 2071-2901 (Online)

Санжаров В.В., Фролов В.А., Волобой А.Г., Галактионов В.А., Павлов Д.С.

Система генерации наборов изображений для задач компьютерного зрения на основе фотореалистичного рендеринга

Рекомендуемая форма библиографической ссылки: Система генерации наборов изображений для задач компьютерного зрения на основе фотореалистичного рендеринга / В.В.Санжаров [и др.] // Препринты ИПМ им. М.В.Келдыша. 2020. № 80. 29 с. http://doi.org/10.20948/prepr-2020-80
URL: http://library.keldysh.ru/preprint.asp?id=2020-80

О р д е н а Л е н и н а
ИНСТИТУТ ПРИКЛАДНОЙ МАТЕМАТИКИ
имени М.В. Келдыша
Р о с с и й с к о й а к а д е м и и н а у к

В.В. Санжаров, В.А. Фролов, А.Г. Волобой,
В.А. Галактионов, Д.С. Павлов

Система генерации наборов изображений для задач компьютерного зрения на основе фотореалистичного рендеринга

Москва — 2020

Санжаров В.В., Фролов В.А., Волобой А.Г., Галактионов В. А., Павлов Д.С.

Система генерации наборов изображений для задач компьютерного зрения на основе фотореалистичного рендеринга

В данной работе предлагается подход к генерации наборов изображений путем фотореалистичного рендеринга с управляемой рандомизацией параметров 3D-сцен. Полученные таким образом наборы изображений могут быть использованы в задачах компьютерного зрения, например, для обучения моделей искусственного интеллекта. Описывается архитектура системы, реализующей предлагаемый подход, приводятся примеры применения к конкретным задачам.

Ключевые слова: фотореалистичный рендеринг, генерация 3D сцен, компьютерное зрение, GPU

Vadim Vladimirovich Sanzharov, Vladimir Aleksandrovich Frolov, Alexei Gennadievich Voloboy, Vladimir Alexandrovich Galaktionov, Denis Sergeevich Pavlov

Image datasets generation system for computer vision applications based on photorealistic rendering

In this paper we present an approach to image datasets generation based on photorealistic rendering with controlled parameter randomization of 3d scenes. These datasets can be used, for example, for training artificial intelligence models in computer vision. We describe the architecture of system implementing proposed approach and show examples of applying it to specific problems.

Key words: synthesis of images, training of neural networks, generation of interiors.

Работа выполнена при поддержке Российского фонда фундаментальных исследований, 18-31-20032 мол_а_вед.

1. Введение

Одна из основных проблем, возникающих при решении задач, использующих нейросетевые модели, – получение достаточного количества исходных данных для обучения и тестирования. Для обучения современных алгоритмов компьютерного зрения требуются наборы данных изображений значительного объема: десятки и сотни тысяч изображений для обучения на статических изображениях, и на порядок больше – для анимации [1, 2]. Сбор подобных объемов данных требует определенных технических возможностей и сопряжен со значительными временными и финансовыми затратами. Проблема количества данных также подразумевает необходимость достаточного представления разных объектов (классов), которые должна распознавать целевая модель, т.е. набор данных должен быть сбалансирован относительно представления в нем разных классов. Это может быть трудно достижимо, поскольку определенные классы могут очень редко встречаться в наборах данных реального мира [3].

Помимо количества данных, важным является и наличие метаинформации о них. Например, в задачах распознавания в компьютерном зрении нужна разметка изображения по классам, на основе которой и будет обучена целевая модель. Разметка большого набора изображений вручную или полуавтоматически может быть затратна. Кроме того, такая разметка часто не имеет необходимой точности, что связано с человеческим фактором и несовершенством средств разметки.

В связи с этим наборы реальных данных, т.е. полученных из реального мира с помощью, например, фото- и видеокамер, могут страдать как недостаточным качеством (как самих данных, так и разметки), так и недостаточным количеством.

Но еще большей проблемой является то, что во время проведения исследований специалисты-аналитики должны изменять входные наборы данных для проверки определенных гипотез. А из-за невозможности быстро получить новый набор данных (в связи с упомянутыми выше проблемами) аналитики могут рассматривать только подмножества существующего набора данных, что значительно ограничивает потенциал исследований.

Одно из решений заключается в использовании синтетических наборов данных, то есть полученных искусственным путем. В случае изображений – синтезированных с помощью алгоритмов рендеринга. Проблема количества данных может быть решена с помощью алгоритмов настройки и выбора оптических свойств материалов и поверхностей, что позволяет быстро генерировать практически неограниченное количество обучающих примеров с любым распределением классов объектов. Кроме того, можно создавать обучающие примеры, которых очень мало или которые полностью отсутствуют наборах данных реального мира. Например, аварийные ситуации на дороге или на производстве, боевые действия, объекты, существующие только в виде

дизайн-проектов или прототипов. Проблема мета-информации при этом решается автоматически – она полностью доступна на этапе генерации данных. Рендер-система создает точные попиксельные маски для разметки изображений по отдельным объектам, а также позволяет получить и другую информацию, такую как карты глубины, карты нормалей, разметку по материалам поверхности и пр.

Несмотря на перечисленные преимущества подхода на основе генерации синтетических наборов данных, на практике при разработке конкретных программных инструментов возникает определенный круг новых проблем.

Основная проблема заключается в непосредственно в генерации 3D-сцен. Вне зависимости от конкретного подхода к генерации целью является создание большого набора разнообразных сцен. Эта цель подразумевает осмысленное размещение объектов в сцене, выбор конкретных 3D-моделей для включения в сгенерированную сцену, установку оптических свойств моделей материалов для объектов на сцене, чтобы они имитировали реальные объекты. Создание 3D-сцен в компьютерной графике на сегодняшний день невозможно без 3D-художников и/или технических художников. Однако для больших наборов данных создание 3D-сцен вручную практически исключено. Кроме того, существующие инструменты художников не подходят для использования специалистами в области компьютерного зрения. Поэтому необходимо создание инструментов компьютерной графики, ориентированных на прямое применение в области обучения моделей искусственного интеллекта (ИИ). При этом такие инструменты должны являться гибкими и расширяемыми, что позволило бы применять их к широкому кругу задач компьютерного зрения.

В этой работе мы предлагаем подход, направленный на построение системы генерации наборов изображений, обладающей расширяемой модульной архитектурой и использующей алгоритмы синтеза фотореалистичных изображений.

2. Существующие решения

На сегодняшний день существует большое количество успешных применений фотореалистичного рендеринга для обучения моделей ИИ. Для нас наибольший интерес представляет непосредственно сама система генерации синтетических данных в этих работах, - каким образом формируются 3D-сцены и их наполнение, насколько возможно применение такого же решения для генерации других наборов данных.

По этим признакам можно условно разделить существующие подходы на следующие группы:

- использующие готовые 3D-сцены, отобранные под задачу,
- использующие методы дополненной реальности,
- осуществляющие процедурную генерацию 3D-сцен,
- с ручной работой по созданию 3D-сцен.

2.1 Использование готовых ресурсов

Использование готовых 3D-сцен позволяет практически полностью избежать основной проблемы синтетических данных. Но сильно ограничивает возможности по генерации и, в некотором смысле, сводит её к использованию реальных наборов данных – работа ведется только с доступными 3D-сценами.

В работе [4] авторы использовали базу данных из 91 высоко детализированной 3D-модели, собранную ими из разных источников. Эти модели были визуализированы с использованием случайного прямого освещения и случайных фотографий в качестве фонового изображения. Т.е. процесс генерации синтетических данных в данном случае был максимально простым, 3D-модели использовались без всякой обработки, а сцены представляли собой просто модель на некотором фоне. Такой подход позволяет достичь фотореалистичного качества изображения только в случае высокого качества моделей (а соответственно, и их высокой стоимости) и использования такой же рендер системы, в которой эти модели были исходно подготовлены. Это сильно ограничивает подбор набора 3D-моделей под конкретную задачу. Также невелики возможности по внесению изменений в модели для повышения вариативности сгенерированных изображений в связи со сложностью взаимодействия с закрытой рендер-системой. Что подтверждается тем, что авторы использовали лишь рандомизацию освещения и фона, но не оптических свойств моделей. Тем не менее в работе [4] показан положительный эффект использования синтетических данных, т.к. даже при использовании таких простых инструментов возможно увеличить вариативность набора данных.

В работе [5] были сгенерированы несколько сотен тысяч изображений из 45 тысяч сцен для обучения моделей семантической сегментации и оценки глубины. В качестве источника 3D-сцен использовался набор данных SUNCG [6], а в качестве рендер-системы – Mitsuba Renderer [7]. Использование 3D-сцен, вместо просто 3D-моделей позволяет генерировать намного больше разнообразных выходных изображений за счет изменения положения виртуального наблюдателя. В остальном процесс генерации данных в [5] также максимально простой. Кроме того, следует отметить, что использование открытой рендер-системы, реализующей алгоритмы фотореалистичного рендеринга, само по себе не позволило авторам достичь фотореализма результирующих изображений. Причиной этому являются исходные сцены, которые содержат лишь базовые настройки материалов и освещения, ограничиваясь только текстурами для диффузной компоненты. Другие работы [8-10], использовавшие тот же набор сцен в качестве исходного, также не достигли фотореалистичного уровня для сгенерированных изображений.

2.2 Дополненная реальность

В связи с высокой стоимостью и трудоемкостью задач по созданию 3D-моделей и сцен на практике рассматриваются любые возможности по сокращению человеко-часов. Один из способов это сделать – подходы на

основе дополненной реальности. В этом случае искусственные объекты встраиваются в реальные сцены (фотографии).

Дополненная реальность используется в работах [11-14]. В частности, в работе [11] используется освещение на основе изображений для отдельных 3D-моделей автомобилей, которые затем встраиваются в изображения набора данных KITTI [15]. Такой подход позволяет значительно упростить как вычислительную сложность синтеза изображений, так и создание 3D-сцен. В качестве исходных данных для генерации используется разметка изображений, в которые производится встраивание, определяющая допустимое местоположение 3D-моделей в результирующей сцене.

Подходы на основе дополненной реальности обладают рядом преимуществ. Во-первых, встраивая одни и те же 3D-модели в разные положения на разных фотографиях, можно достичь значительной вариативности результата при сравнительно небольшом количестве исходных моделей (в [11] использовалось 28 моделей). Во-вторых, происходит работа с конкретным набором реальных данных, что снижает необходимость в применении методов адаптации доменов [16] для постобработки синтезированных изображений.

Существенным недостатком дополненной реальности являются *необходимость реконструкции* освещения и окружения, а также необходимость реализации алгоритмов *автоматического размещения* встраиваемого объекта внутри некоторого реконструированного представления сцены, что в некоторых случаях может быть несложным (например, в [11] использовали методы сегментации для восстановления движущейся плоскости дороги), но является фундаментально трудной задачей в общем случае.

2.3 Процедурное моделирование

Процедурное моделирование – это дорогой и трудоёмкий процесс. Оно требует времени и обычно ограничивается определенной задачей (например, моделирование грязи или ржавчины), а иногда и конкретным алгоритмом рендеринга. Но однажды созданные процедурные модели могут использоваться для генерации практически бесконечного количества вариантов, обладают высоким и настраиваемым качеством для рендеринга.

В работе [17] метод процедурного моделирования был использован для создания генерации городских сцен. Здания, дороги и общий план города генерировались процедурно, в то время как другие модели (автомобили, пешеходы, растительность и др.) выбирались случайным образом из подготовленной базы данных. Также были рандомизированы дополнительные параметры для автомобилей – тип, количество, размещение в сцене и цвет. В работе не упоминается использованная рендер-система и средства подготовки 3D-моделей, но можно предположить, что использовалась собственная. Это предположение основывается на том, что, если процедурные модели были подготовлены независимо от системы рендеринга, их может быть довольно сложно поместить в существующую стороннюю систему. А учитывая тот факт, что условия освещения включают только модели солнца и неба, проще создать

систему рендеринга для этой конкретной задачи, которая будет поддерживать необходимые процедурные модели.

Такая процедурная генерация сцен с использованием собственной рендер-системы позволяет достичь высокой вариативности результатов генерации. Основным недостатком является высокая трудоемкость исходной разработки процедурных моделей и разработка или модификация системы рендеринга для поддержки этих моделей.

Сравнительно недавно появились работы, основанные на процедурном моделировании с помощью машинного обучения [18-20], но здесь возникает проблема курицы и яйца: сначала нам необходимо обучить эти модели. Также эти подходы являются достаточно новыми и еще не апробированы на практике обычных задач рендеринга. Тем не менее следует отметить перспективность данного направления.

Также развиваются направления, использующие различные специальные подходы к описанию и генерации сцен, такие как предметно-ориентированные языки [21], графы сцен [22], стохастические грамматики [23].

2.4 Ручное создание сцен

В работе [24] достигнуты высокий фотореализм и качество сгенерированных изображений за счет использования существующего конвейера создания контента, во многом повторяющего таковой в индустрии кинопроизводства, использующего высококачественные 3D-модели, подготовленные в программном продукте Autodesk Maya, и рендер-систему Arnold. Сцены и 3D-модели были созданы 3D-художниками вручную специально для этой работы или приобретены и изменены. Настройка освещения также производилась вручную. Для повышения вариативности использовалось физическое моделирование для рандомизированного размещения объектов, при этом сами области размещения выбирались вручную. Очевидно, что этот подход страдает от основного недостатка подходов к созданию 3D-сцен в задачах кинопроизводства, разработке компьютерных игр и тренажеров: высокой стоимости и больших трудозатрат. В [24] использовалось только 6 сцен с 30 различными объектами, что значительно меньше, чем в других рассмотренных решениях. Среди других недостатков – использование проприетарных программных продуктов (Maya и Arnold), что ограничивает внедрение этой работы, и медленное время рендеринга, о котором сообщают авторы: от 15 до 720 секунд в зависимости от предустановок качества на 16-ядерных процессорах Xeon с объемом оперативной памяти в 112 ГБ. Следовательно, достаточно быстрое создание набора данных в этом случае возможно только при наличии значительных вычислительных ресурсов. Важно отметить, что авторы сообщают, что точное моделирование контекста сцены было намного более значимым (+ 16% для точности CNN) по сравнению с точным переносом света (+ 6% для точности CNN). Это указывает на важность именно создания высококачественной системы генерации сцен.

Другие работы [25] показывают, что современные игровые движки также могут успешно использоваться для обучения моделей ИИ. Поэтому выбор между игровым движком или системой рендеринга для кинопроизводства должен в первую очередь зависеть от того, насколько легко будет моделировать целевую реальную ситуацию.

2.5 Комбинированные подходы

Отдельно следует упомянуть работу [26], в которой предлагается специализированный конвейер создания 3D-сцен с открытым исходным кодом. Предлагаемый конвейер настраивается с помощью заданных пользователем файлов конфигурации и содержит модули загрузчика, модули визуализации и модули выборки. Модули загрузчика могут импортировать существующие сцены из разных форматов данных (авторы использовали множество наборов данных, включая SUNCG и набор данных Replica [27]). Модули выборки предоставляют возможности рандомизации, что является наиболее интересной частью конвейера. Они позволяют генерировать позиции для размещения объектов (камеры, источников света, 3D-моделей) с различными распределениями и ограничениями, такими как проверка близости, чтобы объекты не располагались слишком близко. Объекты могут быть выбраны на основе определенных пользователем условий, и можно управлять их свойствами, например, выбирать материалы из заданных пользователем наборов и изменять их параметры, текстуры, применять карты смещения и т.д. в зависимости от заданных условий. Это позволило авторам, например, преобразовывать материалы в наборе данных SUNCG, чтобы они больше подходили для фотореалистичного рендеринга. Другой пример – загрузка объектов из ShapeNet и вставка их в набор данных SUNCG с применением физического моделирования, чтобы объекты падали на землю.

Таким образом, в [26] представлена гибкая система, позволяющая трансформировать существующие 3D-сцены, а также создавать новые, используя пользовательские сценарии. Помимо высокой гибкости, значительным преимуществом этого решения является тот факт, что он использует Blender в качестве базового программного продукта с открытым исходным кодом, с фотореалистичной системой рендеринга и доступными сложными инструментами моделирования.

Основным недостатком этой работы является тот факт, что на сегодняшний день она еще не применялась к какой-либо конкретной задаче обучения ИИ, в отличие от многих ранее упомянутых работ. Предложенная авторами система тем не менее дает широкие возможности для пользователя, перенося на него задачи по непосредственной реализации конкретных алгоритмов и подходов к генерации сцен. Благодаря этому получившееся решение является действительно универсальным, но точная оценка этой работы может быть произведена только после публикации её приложений к конкретным задачам.

2.6 Вывод по существующим решениям

В данном разделе были представлены типичные представители разных подходов к решению задачи генерации синтетических данных с использованием фотореалистичного рендеринга.

Мы считаем, что направление [26] в целом наиболее перспективное. Однако, на наш взгляд, необходима интеграция такого решения с конкретными решениями для генерации 3D-сцен, которая может потребовать некоторых модификаций, которые необходимо сделать, чтобы применить такие методы, как [26], и заставить их работать на практике.

Следует отметить довольно ограниченное применение подходов к рандомизации параметров материалов – большинство существующих работ ограничиваются рандомизацией цвета. Изменение других параметров моделей материалов, а также использование процедурного текстурирования содержат богатый потенциал для повышения визуального разнообразия синтезированных изображений.

Также в большинстве существующих решений не упоминается вопрос контролируемости и воспроизводимости случайной генерации сцен, который является весьма важным в контексте генерации набора данных с нужными распределениями объектов и их характеристик.

В данный работе мы предлагаем систему генерации синтетических данных с использованием фотореалистичного рендеринга, обладающую расширяемой модульной архитектурой. За счет использования рандомизации нескольких параметров моделей материалов (в отличие от простого изменения цвета) и процедурного текстурирования возможно достижение высокого уровня вариативности результирующих изображений для небольшого размера исходного набора 3D-моделей. Формализованный подход к случайной выборке параметров генерируемых 3D-сцен позволяет получать контролируемый и воспроизводимый результат генерации.

Предлагаемое решение можно комбинировать с различными методами генерации сцены, некоторые из которых были нами интегрированы и протестированы на реальных задачах обучения нейросетевых моделей компьютерного зрения.

3. Предлагаемое решение

Дизайн нашей системы предполагает несколько ключевых компонентов:

- программное обеспечение для 3D-моделирования;
- программное обеспечение для рендеринга;
- база данных и/или облачное хранилище;
- инструменты постобработки изображений;
- система генерации сцен.

Предлагаемая архитектура системы изображена на рис.1

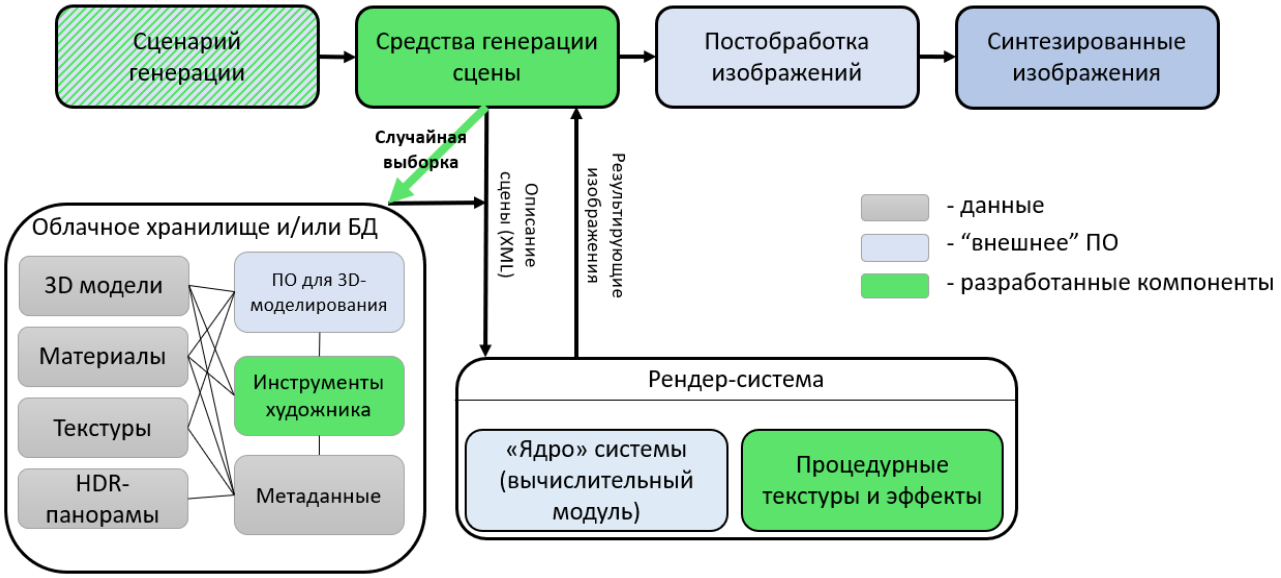

Рис. 1. Предлагаемая архитектура системы генерации синтетических данных

Сценарий генерации определяет настройки для всей процедуры генерации, т.е. для все остальных компонентов системы: классы 3D-моделей, типы освещения (внутри или снаружи, день или ночь и т. п.), настройки случайной генерации (какие параметры выбираются рандомизируются и с каким распределением), выходные данные системы рендеринга (маски сегментации, бинарные маски объектов, нормали, карта глубины и др.), постобработка изображения, которая должна выполняться после рендеринга.

Облачное хранилище или база данных содержит базовые ресурсы для генерации сцен:

- 3D-модели с настройкой тегов материалов (механизм тегов будет рассмотрен позже);
- материалы – окончательно настроенные материалы с назначенными тегами;
- текстуры – набор текстур-изображений и карт нормалей для использования в материалах;
- HDR-панорамы, используемые для освещения на основе изображений, представляющие различные условия освещения;
- метаданные – информация, которая используется сценариями рандомизации для выбора 3D-моделей, материалов и других ресурсов в зависимости от сценария генерации и внутренней информации (например, тегов материалов и целей).

Инструменты художника – специально разработанные инструменты в виде расширения к ПО для 3D-моделирования, позволяющие настраивать ограничения для процесса рандомизации на этапе подготовке 3D-моделей и материалов.

Средства генерации сцены формируют запрошенное количество описаний сцен в соответствии с входным сценарием. Сгенерированное описание сцены предназначено для непосредственного использования системой рендеринга.

Система фотореалистичного рендеринга – в данной работе использовалась рендер-система Hydra Renderer [28], реализующая современные алгоритмы фотореалистичного рендеринга, обеспечивающие высокую производительность (что особенно важно, поскольку требуется генерировать большое количество изображений). Hydra Renderer использует хорошо структурированное представление сцены в формате XML, которое легко редактировать программно и таким образом задавать сцену с помощью средств генерации. Это описание также включает в себя данные о том, какие *процедурные эффекты* следует использовать и каковы их входные параметры (если есть). Hydra Renderer поддерживает пользовательские расширения для процедурных текстур [29], и мы активно используем эту функциональность.

Инструменты *постобработки изображений* корректируют выходные изображения для получения дополнительных данных или для объединения их с фотографиями в случае использования подхода к генерации на основе дополненной реальности. Примером дополнительных данных, созданных на этом этапе, являются ограничивающие прямоугольники для отдельных объектов. Поскольку результирующая нейросетевая модель должна показывать улучшение качества работы, например по критерию точности классификации, на некотором наборе данных реального мира, необходимо иметь возможность выполнять генерацию набора данных, который бы обладал характеристиками, с одной стороны схожими с этим набором данных реального мира, а с другой стороны, дополнял бы его. Схожесть подразумевает в том числе и характеристики самих изображений – наличие оптических искажений, параметры, обусловленные сенсорами, используемыми на практике для получения изображений. Для этих целей компонент постобработки изображений может включать, например, добавление хроматических аберраций, эффекта дисторсии, размытие, преобразование и деформацию изображения, добавление шума и пр. В зависимости от изображений в базовом наборе данных реального мира, с которым идет работа. Все эти задачи могут быть выполнены с использованием простых скриптов или программного обеспечения для композитинга с открытым исходным кодом, такого как Natron [30], и не требуют сложной и требовательной к вычислениям обработки с помощью нейронных сетей.

3.1 Механизм случайной выборки при генерации сцены

Одна из основных особенностей нашей работы по сравнению с существующими подходами – воспроизводимые результаты генерации сцены, которые позволяют генерировать сцены с такими же желаемыми характеристиками для одинаковых входных параметров. Фактически это превращает создание набора данных в задачу формирования выборки.

Имея входной вектор (x_0, x_1, ... x_n) случайных чисел с плавающей точкой от 0 до 1, мы присваиваем каждой случайной переменной определенное значение во время разработки сценария генерации сцены. Например, если используется подход к генерации сцены на основе дополненной реальности наподобие [4, 11] для оценки позы 3D-модели, то получим:

1. пара (x_0, x_1) для описания угла поворота камеры, дополнительно установив допустимые интервалы углов, например от 0 до 70 градусов по вертикали и от 0 до 360 по горизонтали;
2. x_2 для выбора цвета кузова автомобиля из палитры (чтобы ограничить желаемые цвета);
3. x_3 для выбора HDR-панорамы для освещения;
4. x_4 для задания поворота HDR-панорамы вокруг вертикальной оси (Y);
5. x_5 для выбора 3D модели автомобиля.

Далее, производя случайную выборку 6D вектора (x_0 … x_5), получим 3D-сцены и затем их изображения. Такой подход дает нам важные свойства и широкие возможности для дальнейших экспериментов.

- Применяя квазислучайные последовательности (мы использовали последовательность Соболя), мы получим следующие свойства:
  - запуск последовательности Соболя сначала для каждой 3D-модели (т. е. генерировать (x_0 … x_4) с квазислучайным подходом, но выбирать 3D-модели по одной) позволяет нам генерировать одинаковые положения камеры для всех 3D-моделей, что важно для тренировки алгоритмов оценки позы;
  - поскольку последовательность Соболя дает хорошее равномерное распределение в 5D-6D пространстве, сгенерированные образцы также будут равномерно распределены по всем желаемым параметрам: цветам автомобиля, повороту камеры и условиям освещения.
- Применение неравномерного распределения для x_5 позволяет нам отбирать больше автомобилей желаемых типов. А применяя табулированное распределение для x_2, мы можем сгенерировать выборку, например, с большим количеством автомобилей черного и серого цветов.
- Установив разные значения и распределения для обучающего и проверочного наборов данных, мы можем легко проверить некоторые гипотезы. Например, мы можем сгенерировать обучающий набор данных с поворотом камеры только на 25, 50 и 70 градусов, но затем проверить, достаточно ли этого для распознавания при поворотах в 10, 30 и 60 градусов в нашем тестовом наборе данных.
- Применяя кластеризацию многомерных точек, описывающих сцену, мы автоматически получаем важное свойство для рендеринга: когда мы переходим от текущего кадра к следующему, большая часть изменений локализована, и, таким образом, рендер-системе не нужно

выполнять дорогостоящие операции загрузки данных с диска в память графического процессора часто. Это даёт позволяет нам эффективно применять кэширования 3D моделей и текстур. Например, в нашем примере мы хотели бы сгруппировать все сгенерированные точки по (x_3, x_5), чтобы впоследствии визуализировать подряд сцены с одной и той же 3D-моделью и одной и той же HDR-панорамой (рис. 2).

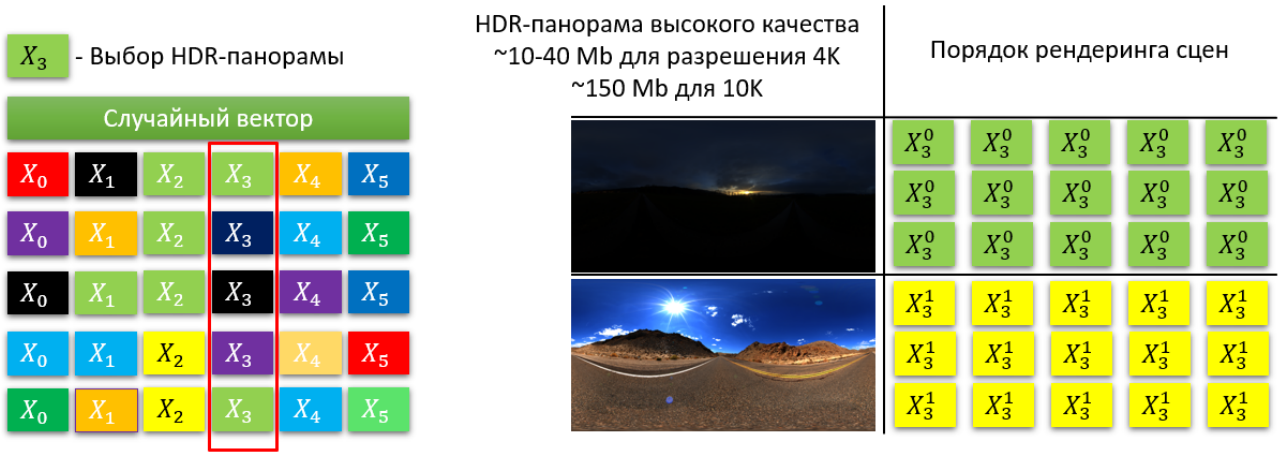

Рис. 2. Пример кластеризации для оптимизации процесса рендеринга: сначала происходит отрисовка сцен с одной HDR-панорамой, затем со следующей и т.д. Таким образом, снижается частота загрузки новых данных на GPU.

Обратим ещё раз внимание на последний пункт, т.к. мы считаем его критически важным. Можно предположить, что эту оптимизацию можно было бы пропустить или добиться её другими способами. Например, если логика рандомизации явно указывает системе рендеринга загрузить конкретную HDR-панораму с конкретной 3D-моделью, а затем сгенерировать все примеры для этой пары. Также можно сгенерировать процедурный контент непосредственно на графическом процессоре с помощью вычислительных шейдеров или т. н. mesh шейдеров. Мы считаем такой путь бесперспективным на практике по следующим причинам.

- Пользователь, составляющий сценарий рандомизации (предположительно, специалист по компьютерному зрению или ИИ), скорее всего, не знает подробностей о возможностях системы рендеринга и в сложных случаях не сделает это оптимальным образом. Даже профессиональные программисты графики могут создавать неоптимальные сценарии рандомизации из-за незнания особенностей работы конкретной системы рендеринга. Таким образом, система рендеринга должна каким-то образом сообщать рандомизатору, какой параметр следует использовать для группировки или сортировки, и это должно происходить автоматически, не усложняя логику работы сценария генерации сцены ручными оптимизациями.

- Хотя некоторые конкретные процедурные подходы могут быть реализованы непосредственно на графическом процессоре (что в нашей работе было сделано для процедурных текстур), практически невозможно избежать дорогостоящего взаимодействия с дисковыми накопителями или инструментами моделирования, работающими на центральном процессоре (например, векторные карты смещения, анимация или моделирование на основе физических моделей). Причина этого в том, что для реалистичного моделирования в большинстве случаев нам приходится полагаться на множество существующих продуктов, которые могут работать в течение минут или даже часов (хотя и дают реалистичные результаты).
- Для систем, которые применяют алгоритмы рендеринга в реальном времени и напрямую передают результат в нейросетевую модель на GPU, эта оптимизация имеет решающее значение. Даже в нашем случае («оффлайн» рендер-система с трассировкой пути на GPU) мы получаем ускорение от 1,5 до 3 раз для рассматриваемого случая, потому что входные изображения HDR имеют высокое разрешение (8Kx4K, что является нормальным для HDR-панорам). Быстро загрузить такую текстуру с диска невозможно в силу физических ограничений. Кроме того, есть много случаев, когда не хватает памяти для всех текстур и моделей, чтобы поместить их все сразу в память.

3.2 Инструменты художника

На практике написание сценариев рандомизации оказывается довольно трудоемкой задачей, потому что после написания первой версии необходимо запустить сценарий, проверить сгенерированные изображения (по крайней мере, несколько десятков изображений выборочно), изменить сценарий, если что-то пошло не так, снова запустить сценарий и т.д. Чтобы усовершенствовать данный процесс, часть работы по рандомизации была реализована в качестве инструмента художника, которые интегрирован с программным обеспечением 3D-моделирования. Основная задача, решаемая этим инструментом, – обеспечить настройку случайных распределений параметров материалов уже на этапе подготовки 3D-моделей и материалов так, чтобы при генерации получался приемлемый с точки зрения реализма результат.

В отличие от традиционных подходов к процессу создания 3D-моделей (в 3ds Max, Blender или другом программном обеспечении), наше решение позволяет художнику создавать не одну, а целый набор 3D-моделей с желаемым распределением оптических свойств поверхности – параметров моделей материалов и процедурных текстур (которые на самом деле также являются свойствами материала). На рис. 3 показан разработанный графический интерфейс для материала, а на рис. 4 показан предварительный просмотр результатов рандомизации.

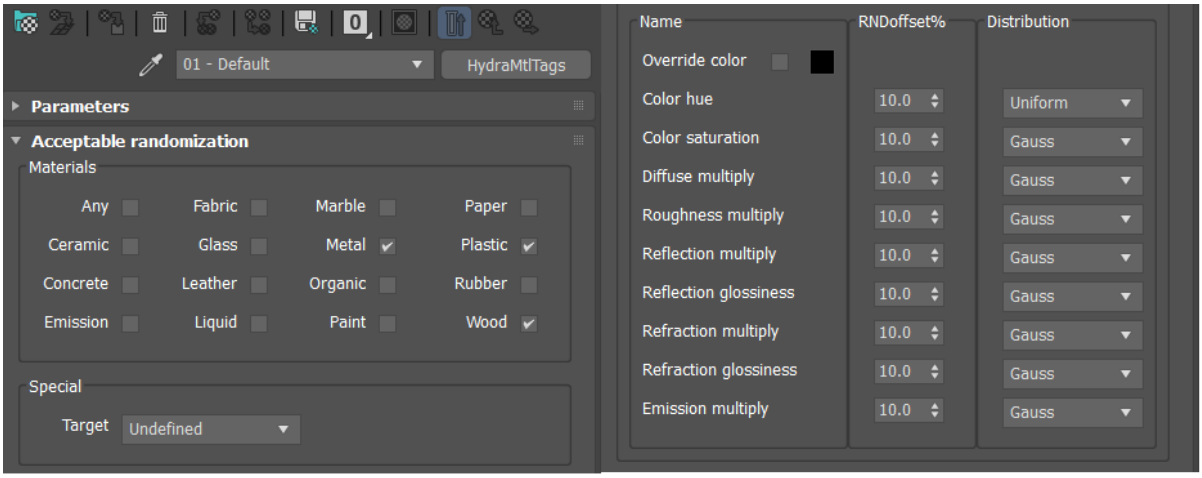

Рис. 3. Интерфейс пользователя средств художника для настройки распределений материалов. Вкладка “Acceptable randomization” позволяет назначать для материалов теги и/или цели. В правой части интерфейса – настройки распределений для отдельных параметров материалов.

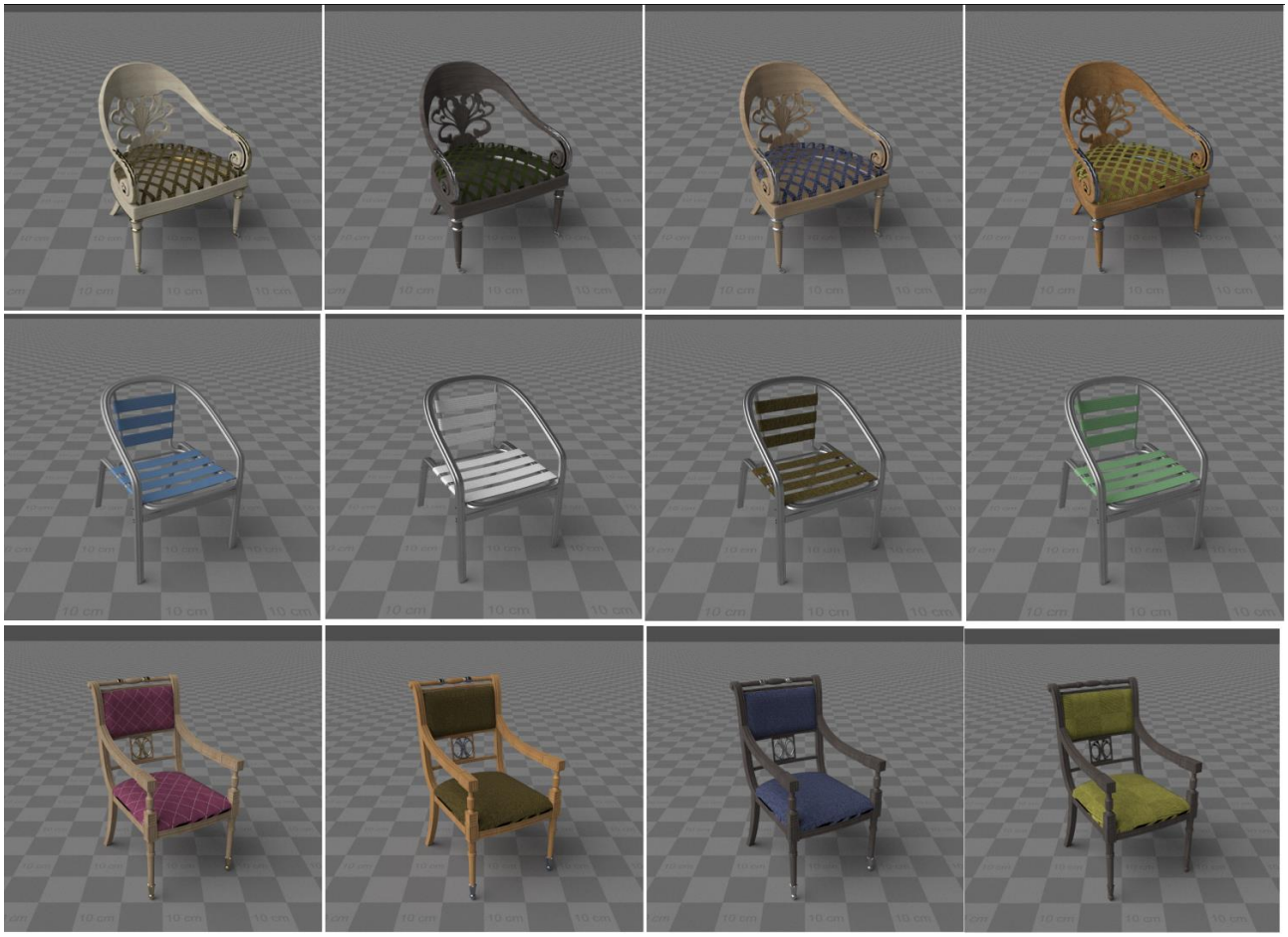

Рис. 4. Пример предварительного просмотра для результатов рандомизации: результат назначения случайных материалов разным 3D-моделям стульев.

Чтобы назначение случайных материалов для 3D-моделей давало адекватный с точки зрения реализма результат, необходимо учитывать две особенности.

Во-первых, два материала одной и той же природы, например дерево, фактически нельзя менять местами во всех случаях. Возьмем, к примеру, деревянный карандаш и деревянный пол. Оба этих объекта имеют материал дерева, но обладают разной микроструктурой древесины и, что наиболее важно, разными текстурными матрицами (в частности компонентами, отвечающими за масштабирование). Поэтому назначение материала пола на карандаш выглядит не просто странно, но и совершенно неправильно. Когда камера располагается близко карандашу, мы увидим на нем уменьшенные копии, например, паркетной доски. А если виртуальный наблюдатель расположен далеко, то, скорее всего, мы получим просто какой-то неизвестный цвет (усреднённый цвет по всем пикселам текстуры).

Во-вторых, существует множество материалов, которые можно применять только для определенных типов объектов: материал экрана телевизора или монитора компьютера может отображаться на рекламном плакате и наоборот, но не может отображаться на чашке или на любом другом предмете интерьера. А материалы для чашек могут содержать текстуру, обернутую вокруг её 3D-модели. Изображения для таких текстур представляют собой вытянутые по горизонтали прямоугольники (обычно 2x1), и нанесение такой текстуры на многие другие объекты дает совершенно нереалистичный внешний вид с растянутым изображением.

Исходя из этих особенностей мы разделили все рандомизированные материалы на две разные категории: универсальные и специальные. Универсальные материалы используют список тегов (рис. 3, верхняя левая часть). Теги определяют, какие типы конечных материалов могут заменить этот материал при генерации сцены. Например, на рис. 3 отмечены теги для пластика, металла и дерева, что означает, что данный материал может быть заменен другим материалом с такими же тегами.

Специальные материалы означают, что такие материалы могут появляться только на определенных типах объектов, называемых целями (рис. 3, нижняя левая часть). Например, экран телевизора, дорожный знак, обложка книги, автомобильные шины. Из-за специфических текстурных координат и фактического содержимого текстуры для таких случаев невозможно заменить эти материалы на какие-либо другие и, наоборот, – применить их к другим объектам. Поэтому целевой материал можно заменить только на другой целевой материал для такой же цели.

Работа по подготовке 3D-моделей и материалов с использованием разработанных инструментов разделена на два этапа: заполнение базы данных материалов и заполнение базы данных объектов. На первом этапе создаются настроенные материалы и для них указываются теги или цель. Один материал может иметь несколько типов в случае, если он может быть объединен с

другими материалами или использован как часть какой-либо сложной модели материала. Также устанавливаются желаемые параметры распределения (правая часть интерфейса на рис. 3), а затем тестируются на каком-то простом объекте.

На втором этапе для 3D-моделей назначаются материалы-заглушки, для которых указываются только теги или цели. На этом этапе можно запустить проверку рандомизатора (рис. 4), изменить теги / цель или вернуться к первому этапу, если необходимо создать дополнительные материалы. В идеале следует разделить интерфейсы для первой фазы (окончательная настройка материалов) и второй фазы (настройка материала-заглушки), поскольку они фактически устанавливают разные параметры, но технически это было неудобно. Примеры работы рандомизации материалов, полученные с помощью разработанной системы, представлены на рис. 5.

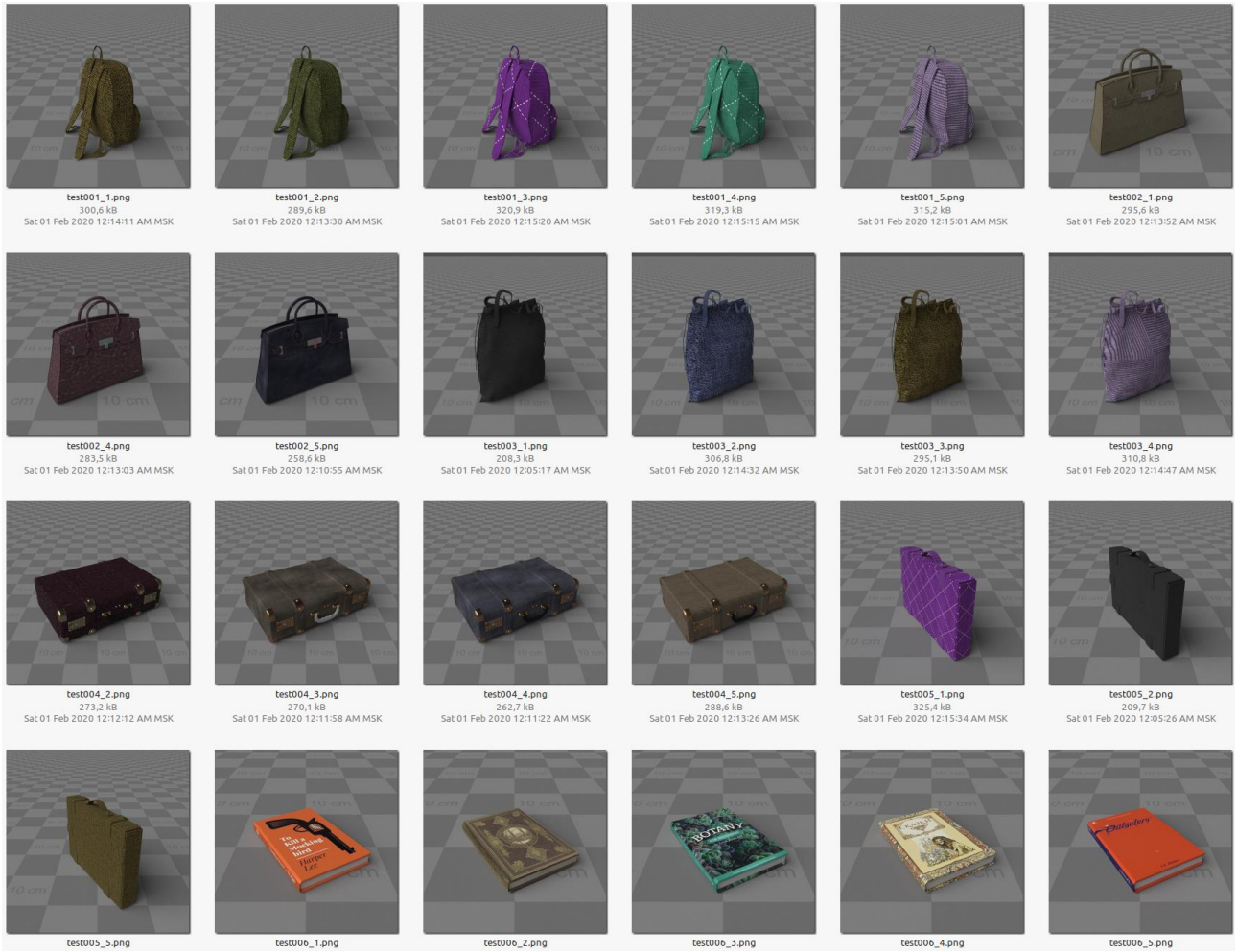

Рис. 5 Пример сгенерированных изображений для разных моделей в директории, скриншот из файлового менеджера.

3.3 Использование процедурных текстур

Как упоминалось ранее, в разработанном подходе активно использовались процедурные текстуры. Основная цель использования процедурных текстур

заключалась в добавлении дополнительных деталей визуализированным 3D-моделям для создания более реалистичных изображений. В рамках работы были реализованы процедурные текстуры, имитирующие такие эффекты, как грязь, ржавчина, царапины, обледенение (рис. 6).

Рис. 6. Примеры процедурных текстур

Все эти текстуры параметризованы, что позволяет легко изменять внешний вид текстуры на этапе генерации сцены. Большинство этих процедурных эффектов основано на функциях шума, следовательно, параметры функции шума, такие как амплитуда, частота и постоянство (или коэффициенты, примененные к этим параметрам каким-либо образом), могут быть использованы в качестве параметров сцены и могут задаваться случайными величинами. Кроме того, одним из параметров процедурных текстур является относительный размер объекта. Этот параметр позволяет управлять распространением процедурного эффекта по поверхности объекта – например, процедурная текстура грязи может быть настроена так, чтобы она накладывалась только на нижние части модели. Подобная гибкость невозможна с обычными текстурами-изображениями.

Важно отметить, что использование процедурного подхода к текстурированию в некоторых случаях является критичным, потому что 3D-модели могут не иметь текстурных координат. И это достаточно частый случай даже с высококачественными 3D-моделями из открытых источников. Процедурные текстуры могут принимать мировые (или локальные) координаты точки поверхности в качестве входных данных, что может использоваться для отображения текстур. Кроме того, даже при наличии текстурных координат на модели, нет никакой гарантии, что текстуры, например, грязи или ржавчины

будут выглядеть правильно на разных 3D-моделях из-за коэффициента масштабирования этих моделей при преобразовании из локального пространства в мировое. В таких случаях использование трехмерных процедурных текстур, зависящих от мировой позиции, является единственным вариантом.

4. Способы генерации расположения объектов

Архитектура предлагаемой системы позволяет легко интегрировать разные механизмы генерации расположения объектов. В процессе применения предлагаемой системы к различным конкретным задачам генерации данных, было использовано несколько разных подходов.

4.1 Автомобили и дорожные знаки

Первый подход реализовывал схему с использованием дополненной реальности, во многом схожую с [11], где для имеющихся изображений из набора данных [15] полуавтоматически строилась разметка траекторий на участках дороги (рис. 7), после чего производилась случайная выборка положений на этих траекториях.

Рис. 7. Разметка траекторий на исходных изображениях (слева) и встраивание 3D-моделей на случайные позиции на траекториях (справа).

4.2 Аугментация в фотографии интерьеров

Второй опробованный подход также использовал дополненную реальность. В этом случае встраивание объектов производилось с использованием карты глубины. Было использовано два разных варианта. В обоих вариантах из карты глубины генерировалось облако точек, которое затем подвергалось обработке – удалению статистических выбросов, снижению уровня детализации с помощью воксельной сетки, восстановлению нормалей. В первом варианте далее выполнялась сегментация плоскостей из облака точек. После чего с использованием описанного в разделе 3.1 подхода одним случайным числом выбиралась одна из плоскостей, на которой затем выбиралась точка для встраивания 3D-модели в зависимости от того, является плоскость вертикальной или горизонтальной. На рис. 8 и 9 представлены примеры работы.

Рис. 8. Пример реконструкции плоскостей для изображения из набора данных NYUv2 [31]

Рис. 9. Примеры встраивания объектов (3D-моделей стульев)

4.3 Генератор интерьеров

Третий подход использовал полную генерацию интерьерных 3D-сцен. Далее рассмотрим его подробнее.

Разработанное решение для генерации интерьеров можно условно разделить на три разные подзадачи: создание макета, расстановка мебели и размещение объектов. Для генерации макета был реализован метод, основанный на сочетании методов «плотной упаковки» [32] и «наизнанку» [33].

Для расстановки мебели был реализован подход, управляемый художником. Существуют более совершенные методы расстановки [34, 35], но для нас было важно использовать контролируемый метод, который вписывается в описанную в разделе 3.1 концепцию случайной выборки. В разработанном подходе сначала производится обход по периметру стен макета, и происходит случайная расстановка мебели по одному из заранее подготовленных одномерных шаблонов. Затем, чтобы расставить мебель внутри комнаты, используется

несколько заранее подготовленных художником бинарных 2D изображений-шаблонов. Мы назначаем 4 случайных числа для частоты и номера одномерного и двухмерного шаблонов. В процессе генерации сцены случайным образом выбирается шаблон и частота.

Поскольку реальные объекты, в отличие от точек, имеют размеры, после генерации необходимо выполнить обнаружение пересечений объектов. И, если было обнаружено хотя бы одно пересечение, необходимо использовать правило «tabula rasa» [36], очистив все и запустив генерацию экземпляра сцены заново. В противном случае будет получено неправильное распределение [37].

Наконец, для размещения более мелких объектов на поверхностях других объектов (например, на столах) был использован аналогичный подход на основе шаблонов. После проведения нескольких экспериментов с физическим моделированием, как в [24], оказалось, что на практике трудно добиться адекватного и воспроизводимого результата: объекты собирались в кучи, падали сквозь поверхности стола или пола или находились в бессмысленных конфигурациях (мышка на мониторе, книга на тарелке и т. д.). В результате мы остановились на подходе, при котором были подготовлены несколько шаблонов с настроенными ограничивающими прямоугольниками и углами поворота для каждого типа объектов на столе или на поверхности пола. Позиции объектов могут быть сгенерированы только в пределах ограничивающих прямоугольников, и, если происходит столкновение, применяется правило «tabula rasa», упомянутое ранее.

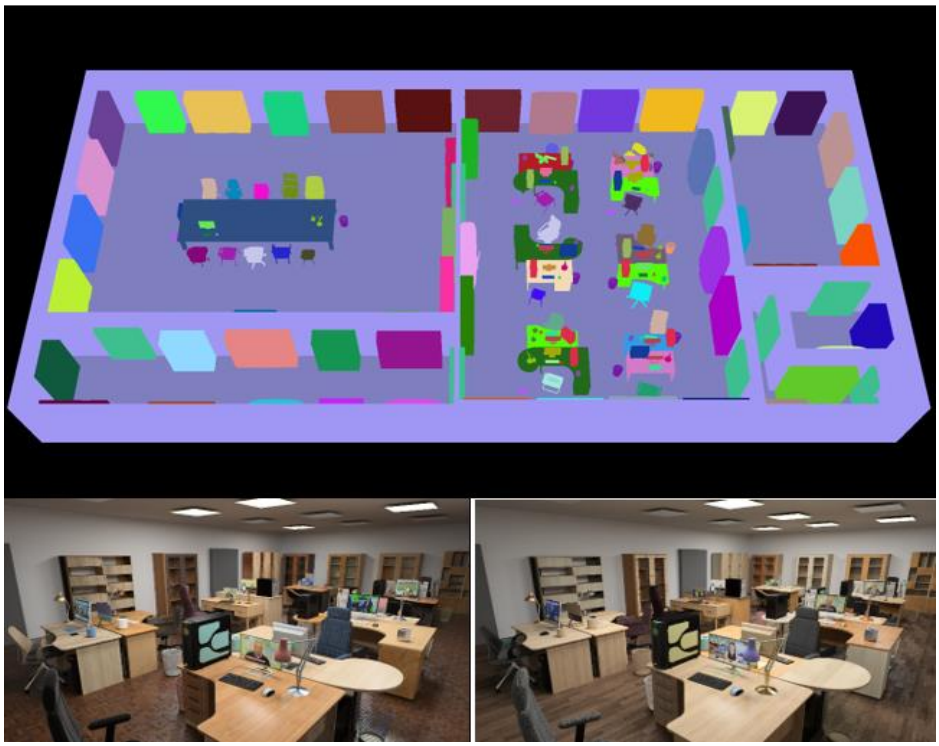

Рис. 10. Примеры сгенерированных интерьерных сцен

Таким образом, расстановка объектов также реализована через вектор входных случайных чисел. Хотя в данном случае длина вектора оказывается намного больше из-за количества объектов, и здесь мы использовали только псевдослучайные числа. Примеры сгенерированных сцен представлены на рис. 10.

4.4 Генератор движений людей

Для создания синтетического датасета с движениями человека был использован SURREAL [13]. Мы реорганизовали существующий генератор 3D моделей согласно общему подходу нашей работы таким образом, чтобы контролировать форму модели человека и варьировать базовый меш между "худым/полным", "низким/высоким" и т.д., подавая на вход вектор случайных чисел. Наша реализация позволяет генерировать короткие последовательности движений синтетических людей с различными данными, включая глубину, сегментацию, скелет модели, оптический поток и другие (рис. 11).

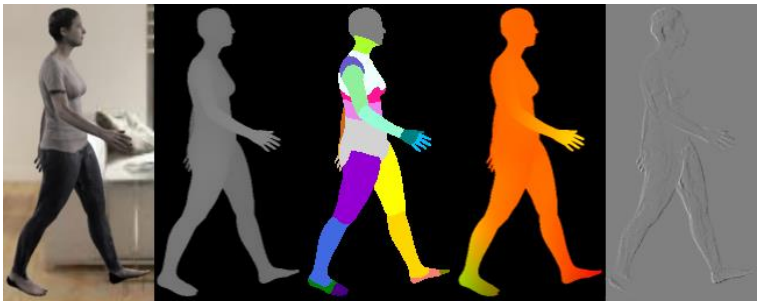

Рис. 11. Глубина, сегментация, оптический поток и др.

Генератор движений человека использовался в разработанной ранее программе аугментации наряду с 3D моделями мебели из базы для генерации выборок на основе дополненной реальности (рис. 12).

Рис. 12. Добавление синтезированных 3D моделей мебели и людей в существующую базу изображений DISCOMAN [8]. Изображения не выглядят

реалистичными вследствие низкой реалистичности DISCOMAN и отсутствия необходимой для аугментации информации об освещении.

5. Примеры применения

Разработанная система генерации синтетических данных была апробирована на задачах распознавания редких дорожных знаков и автомобилей.

Для автомобилей для экспериментальной оценки использовался набор данных KITTI [15]. Были составлены обучающая и проверочная выборки по 200 кадров. В качестве исходных данных для генерации использовались 50 различных 3D-моделей автомобилей, процедурные текстуры для моделирования грязи, ржавчины и проективного наложения изображений на 3D-модели, для освещения использовался набор HDR-панорам. Позиции автомобилей выбирались на траекториях (рис. 7), построенных как кусочно-линейные функции для всех 200 изображений (плоскость дороги была реконструирована, исходя из известных параметров камеры).

Для каждого изображения из обучающей выборки было проведено встраивание 2-5 случайных моделей автомобилей объектов 20 раз с использованием разработанной системы генерации синтетических данных. В результате был получен расширенный обучающий набор из 4000 изображений. Также был сгенерирован набор данных со случайными горизонтальными отражениями (вероятность отражения 0,5) исходных изображений, чтобы понять преимущества нашего подхода по сравнению с простыми методами увеличения размера обучающих данных. Были построены три нейросетевые модели (на основе Faster-RCNN [38] м ResNet-50 FPN) – исходные изображения, исходные изображения + отраженные изображения, сгенерированные синтетические данные.

Для оценки качества детектора использовалась метрика mAP. Результирующие значения метрики mAP, демонстрирующие увеличение качества для синтетических данных, полученных с помощью разработанного подхода, показаны в таблице 1.

Таблица 1. Значения метрики mAP для набора данных по автомобилям.

Набор данных	Метрика
Исходные изображения	38.67 %
Исходные + отраженные	40.30 %
Синтетические (наши)	**43.26 %**

В задаче распознавания дорожных знаков [3] использование генерации синтетических данных с помощью предлагаемой системы позволило поднять точность распознавания редких дорожных знаков от 0% в базовом наборе

данных RTSD (Russian traffic sign dataset)[39] до 69.7% без использования нейросетевой постобработки методом CycleGAN и до 71% с её использованием.

6. Выводы

В данной работе предложена система генерации синтетических данных с использованием фотореалистичного рендеринга для обучения нейросетевых моделей в задачах компьютерного зрения. В отличие от существующих решений, предложенная система была протестирована в нескольких сценариях для генерации разных наборов данных, что позволяет нам обобщить полученные результаты.

Предложенное решение отличается гибкостью и может интегрировано с разными подходами генерации 3D-сцен, что было показано на примерах, использующих подходы на основе дополненной реальности и полной генерации синтетических сцен.

Формализованный подход к процессу случайной выборки генерируемой сцены и процесс создания исходных ресурсов (3D-моделей и материалов), тесно связанный с конкретным приложением, позволяет генерировать наборы данных с желаемым распределением свойств и обеспечивает воспроизводимость результатов генерации. Для одного и того же набора входных параметров, представляемых как точки в многомерном пространстве, разработанное решение всегда генерирует одну и ту же сцену и, соответственно, одно и тоже рассчитанное изображение.

Кластеризация многомерных векторов случайных чисел, описывающих генерируемые сцены, позволяет автоматических оптимизировать процесс рендеринга за счет уменьшения числа операций загрузки ресурсов (текстур, 3D-моделей).

Рандомизация большинства параметров моделей материалов (а не только цвета) в комбинации с процедурным текстурированием позволяет достичь высокой вариативности результирующих изображений для небольшого количества исходных 3D-моделей и материалов, подготовленных вручную художниками.

Список литературы

[1] Karpathy, Andrej, et al. *Large-scale video classification with convolutional neural networks* // Proceedings of the IEEE conference on Computer Vision and Pattern Recognition. 2014.

[2] Wu, Zuxuan, et al. *Deep learning for video classification and captioning* // Frontiers of multimedia research. pp. 3-29. 2017.

[3] Фаизов Б.В., Шахуро В.И., Санжаров В.В., Конушин А.С. *Классификация редких дорожных знаков* // Компьютерная Оптика, Т. 44, №2, 2020.

[4] Movshovitz-Attias, Y., Kanade, T., and Sheikh, Y. *How useful is photo-realistic rendering for visual learning* // European Conference on Computer Vision., pp. 202–217. 2016.

[5] Zhang, Y., Song, S., Yumer, E., Savva, M., Lee, J.Y., Jin, H. and Funkhouser, T. *Physically-based rendering for indoor scene understanding using convolutional neural networks* // Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 5287–5295. 2017.

[6] Song, S., Yu, F., Zeng, A., Chang, A.X., Savva, M. and Funkhouser, T. *Semantic Scene Completion from a Single Depth Image* // arXiv*:1611.08974*. 2016.

[7] Jakob W. *Mitsuba Renderer //* URL: http://www.mitsuba-renderer.org (accessed 01.09.2020)

[8] Kirsanov, P., et al. *DISCOMAN: Dataset of Indoor Scenes for Odometry, Mapping And Navigation //* arXiv preprint arXiv:1909.12146. 2019.

[9] Li, Z., and Snavely, N. *Cgintrinsics: Better intrinsic image decomposition through physically-based rendering* // Proceedings of the European Conference on Computer Vision (ECCV), pp. 371–387. 2018.

[10] McCormac, J., Handa, A., Leutenegger, S., and Davison, A. J. *SceneNet RGB-D: Can 5M Synthetic Images Beat Generic ImageNet Pre-training on Indoor Segmentation //* Proceedings of the IEEE International Conference on Computer Vision, pp. 2678-2687. 2017.

[11] Alhaija, Hassan Abu, et al. *Augmented reality meets computer vision: Efficient data generation for urban driving scenes* // International Journal of Computer Vision 126.9: 961-972. 2018.

[12] Dosovitskiy, Alexey, et al. *Flownet: Learning optical flow with convolutional networks* // Proceedings of the IEEE international conference on computer vision. 2015.

[13] Varol, Gul, et al. *Learning from synthetic humans* // Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. 2017.

[14] Chen, Wenzheng, et al. *Synthesizing training images for boosting human 3d pose estimation* // 2016 Fourth International Conference on 3D Vision (3DV). IEEE, 2016.

[15] Geiger, A., Lenz, P., Stiller, C. and Urtasun, R. *Vision meets robotics: The kitti dataset* // The International Journal of Robotics Research, Vol. 32, No. 11, pp.1231-1237. 2013.

[16] Tremblay, J. et al. *Training deep networks with synthetic data: Bridging the reality gap by domain randomization* // arXiv preprint arXiv:1804.06516, 2018.

[17] Tsirikoglou, A., Kronander, J., Wrenninge, M. and Unger, J. *Procedural Modeling and Physically Based Rendering for Synthetic Data Generation in Automotive Applications* // arXiv preprint *arXiv:1710.06270. 2017.*

[18] Risi, S., and Togelius, J. *Increasing Generality in Machine Learning through Procedural Content Generation //* arXiv preprint *arXiv:1911.13071*. 2019.

[19] Chelliah, B. J. et al. *3D Character Generation using PCGML //* International Journal of Innovative Technology and Exploring Engineering (IJITEE)ISSN: 2278-3075, Volume-8 Issue-6S, 2019.

[20] Spick, R.J., Cowling, P. and Walker, J.A. *Procedural Generation using Spatial GANs for Region-Specific Learning of Elevation Data // IEEE Conference on Games (CoG)*, pp. 1-8. 2019.

[21] Fremont, Daniel J., et al. *Scenic: a language for scenario specification and scene generation* // Proceedings of the 40th ACM SIGPLAN Conference on Programming Language Design and Implementation. 2019

[22] Armeni, Iro, et al. *3D Scene Graph: A Structure for Unified Semantics, 3D Space, and Camera* // Proceedings of the IEEE International Conference on Computer Vision. 2019.

[23] Jiang, Chenfanfu, et al. *Configurable 3d scene synthesis and 2d image rendering with per-pixel ground truth using stochastic grammars* // International Journal of Computer Vision 126.9: 920-941. 2018.

[24] Hodaň, T., Vineet, V., Gal, R., Shalev, E., Hanzelka, J., Connell, T., Urbina, P., Sinha, S.N. and Guenter, B. *Photorealistic image synthesis for object instance detection //* arXiv preprint *arXiv:1902.03334. 2019.*

[25] Shital Shah and Debadeepta Dey and Chris Lovett and Ashish Kapoor. *AirSim: High-Fidelity Visual and Physical Simulation for Autonomous Vehicles.* ArXiv:1705.05065. 2017.

[26] Denninger, M. et al. *BlenderProc //* arXiv preprint *arXiv:1911.01911. 2019.*

[27] Straub, J. et al. *The replica dataset: A digital replica of indoor spaces //* arXiv preprint *arXiv:1906.05797. 2019.*

[28] Frolov, V., Sanzharov, V., Trofimov, M., Pavlov, D. and Galaktionov, V. *Hydra Renderer. Open source GPU based rendering system. 2018. //* URL: https://github.com/Ray-Tracing-Systems/HydraAPI (accesed 01.09.2020)

[29] Sanzharov, V. and Frolov, V. *Level of Detail for Precomputed Procedural Textures* // Programming and Computer Software, V. 45, No. 4, pp. 187-195. 2019.

[30] *Natron* // open source compositing software. URL: https://natrongithub.github.io/ (accesed 01.09.2020)

[31] Silberman, Nathan, et al. *Indoor segmentation and support inference from rgbd images //* European conference on computer vision. Springer, Berlin, Heidelberg, 2012.

[32] Koenig, R., Knecht, K. *Comparing two evolutionary algorithm-based methods for layout generation: Dense packing versus subdivision // AI EDAM*. Vol 28, No. 3, pp. 285-299. 2014.

[33] Martin, Jess. *Procedural House Generation: A method for dynamically generating floor plans //* University of North Carolina, Chapel Hill. 2016. URL:

https://pdfs.semanticscholar.org/7afd/472787d5f1c4898b1599d6a6f2c1ae713ccb.pdf (accesed 01.09.2020)

[34] Qi, S., Zhu, Y., Huang, S., Jiang, C., and Zhu, S. C. Human-centric Indoor Scene Synthesis Using Stochastic Grammar. arXiv preprint *arXiv:1808.08473v1*.

[35] Wang, K., Savva, M., Chang, A.X. and Ritchie, D. *Deep convolutional priors for indoor scene synthesis //* ACM Transactions on Graphics (TOG). Vol. 37, No. 4, pp. 1-14. 2016.

[36] Krauth, Werner. 2015. *Advanced Monte Carlo algorithms. // course lect.* pp 11-12. 2015. URL: http://www.lps.ens.fr/~krauth/images/5/50/BadHonnef_2.pdf (accesed 01.09.2020)

[37] Nandor Simanyi. *Proof of the Boltzmann-Sinai Ergodic Hypothesis for Typical Hard Disk Systems //* arXiv:math/0008241v4. 2003.

[38] Ren, S., He, K., Girshick, R., and Sun, J. F*aster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks //* Advances in neural information processing systems. pp. 91-99. 2015.

[39] Shakhuro, V., and Konushin A. *Russian traffic sign images dataset* // Computer optics 40.2: 294-300. 2016.

Оглавление