УДК 004.92:004.032.26
DOI: 10.15827/0236-235X.139.329-339
Дата подачи статьи: 27.07.22, после доработки: 05.08.22
2022. Т. 35. № 3. С. 329–339

Структуризация сущностей естественного текста с использованием нейронных сетей для генерации трехмерных сцен

Б.А. Козар [1], *инженер, bogdan.kozar.itis@gmail.com*
В.В. Кугуракова [1], *к.т.н., доцент, vlada.kugurakova@gmail.com*
Г.Ф. Сахибгареева [1], *ассистент, gulnara.sahibgareeva42@gmail.com*

[1] *Казанский (Приволжский) федеральный университет, г. Казань, 420008, Россия*

Предметом настоящего исследования является автоматизация с помощью нейронных сетей процесса сборки трехмерной сцены, которая может быть использована как для генерации по текстовому описанию трехмерных сцен или локаций в компьютерных играх, так и для подготовки секвенций трехмерных синтетических данных. Эта тематика актуальна для разработки трехмерной графики, в том числе для интерактивных проектов – игр, тренажеров, AR/VR-приложений.

В результате анализа и сравнения результатов, полученных в ряде известных выполненных проектов, выделены те технологии и программные библиотеки, которые позволяют эффективно достичь поставленной цели – предоставить быструю сборку трехмерных сцен, наполненных объектами, согласно текстовому описанию. Благодаря синтезу лучших решений удалось создать оптимальную концепцию, которая позволяет добиться быстрого и качественного результата при верно сформулированных правилах по выстраиванию геометрических отношений между объектами сцены. Сформированы перечень требований к проектируемому инструменту и его архитектура. Входными данными для применения этого инструмента является текст на естественном языке, выходными – сцена с объектами, соответствующими использованному описанию.

Основным достигнутым результатом стал готовый программный инструмент для Unreal Engine, разработанный на основе нейронной сети nlp-ue4 и набора библиотек tensorflow, nltk, pandas, gensim, h5py. Готовность инструмента оценена как прототипное решение, которое можно интегрировать в этап черновой сборки интерактивных проектов с трехмерной графикой.

Для объективной оценки эффективности созданного инструмента проведены эксперименты, которые доказали, что его применение даже в текущей версии значительно сокращает время разработки, а также не требует наличия у пользователя навыков программирования или создания трехмерной графики.

Обсуждены также перспективы развития исследований.

Ключевые слова: *компьютерная графика, трехмерное моделирование, визуализация, процедурная генерация, дизайн уровней, автоматизация, NLP, нейронные сети, Unreal Engine, компьютерные игры.*

Ручная сборка игровых проектов трудоемка и занимает много времени, а очевидный риск того, что результат труда не станет итоговым, в свою очередь, порождает исправления и переделки вплоть до реконструкции этой сборки с самого начала. Как следствие, в разы увеличивается вероятность того, что проект может выйти за рамки установленного бюджета. В таком случае на помощь приходят нейронные сети, которые позволяют получать результаты в различных областях, например, при решении конкретных задач по визуализации объектов на основе входных изображений или текста [1–3], и дают возможность сократить время на поиск (или генерацию) определенных моделей, упомянутых в тексте.

Геймдизайн включает в себя не только создание сценария с вариантами развития игры, описание игровых механик, персонажей, но и детальную разработку сцен, где происходит действие игры, и тогда генерация бесконечного множества вариаций таких локаций, основанных на одном и том же текстовом описании, становится серьезной основой для работы геймдизайнера. Подобное увеличение выборки вариантов локаций также увеличивает возможность нахождения наиболее уникального и лучшего результата, что сокращает время не только на визуальное представление, но и на сборку подобной сцены вручную, поскольку все трехмерные модели уже будут находиться на своих местах на игровой сцене. Существую-

щие подходы в области генерации трехмерных сцен [4] сегодня никак практически не используются и созданы только ради искусства. Предложенный алгоритм позволяет легко интегрировать генерацию сцен в собственные проекты, например, для визуализации при сценарном прототипировании [5].

Обзор связанных работ

Идея создания визуального контента не нова и уже имеет множество воплощений. Рассмотрим самые яркие из них.

Метод Joint Embeddings (совместных вкраплений) в инструменте ShapeNet (https://shapenet.org/) позволяет добиваться генерации цветных реалистичных воксельных моделей с разнообразными текстурами и деталями (рис. 1).

Рис. 1. Генерация воксельной графики по текстовому описанию

Fig. 1. Voxel graphics generation by text description

Нейросеть Dream Fields (https://www.dreamfields.com/) с использованием модели CLIP (https://openai.com/blog/clip/) от Google AI способна генерировать уникальные трехмерные объекты с различной стилизацией (см. http://www.swsys.ru/uploaded/image/2022-3/2022-3-dop/34.jpg). Другое название нейросети Dream Fields – DALL-E (https://rudalle.ru/) для трехмерной графики, и в 2022 году результат ее эффектных генераций произвел фурор среди специалистов.

Благодаря CLIP можно усложнить геометрию существующих низкополигональных моделей по текстовому запросу (рис. 2), как это реализовано в Text2Mesh [6].

Важными аспектами трехмерной графики являются освещение и тени. Если трехмерная модель получена на основе ряда фотографий (см. http://www.swsys.ru/uploaded/image/2022-3/2022-3-dop/35.jpg), например, с применением NeRF [7], то использование некоторых методов позволяет решить и задачу автоматического изменения освещения, как, например, это реализовано в NeROIC [3].

Более сложной является задача генерации трехмерного объекта по ограниченному набору его изображений. Например, существует решение [8], способное генерировать модель всего лишь по двумерной проекции, то есть по изображению, которое иллюстрирует силуэт объекта (см. http://www.swsys.ru/uploaded/image/2022-3/2022-3-dop/36.jpg).

Рис. 2. Стилизация низкополигональной модели человека под Железного человека

Fig. 2. A low-poly human model styled on Iron Man

Еще сложнее восстановить данные об объекте только по одной фотографии. Так, например, в методе BARC (Breed-Augmented Regression using Classification – регрессии с классификацией на основе дополнительных данных о классифицируемых объектах) [9] реализована функция определения породы и позы собаки по одному изображению (см. http://www.swsys.ru/uploaded/image/2022-3/2022-3-dop/37.jpg).

С задачей создания примитивных объектов с учетом их описания и связей между ними справляется алгоритм Part2Word, тоже использующий метод Joint embeddings. Его особенность заключается в том, что все используемые модели состоят из сборочных частей (рис. 3) и при входном значении текста алгоритм собирает данную модель на основе описания, что делает каждый запрос уникальным [10].

Помимо модульной генерации отдельных частей фурнитуры в одну полноценную модель, существует готовое решение генерации трехмерной модели с разной перспективой на основе фотографий, представленных как входные данные. Такой алгоритм выстраивает сферические сверточные сети и использует их как инструмент представления трехмерных фигур [11].

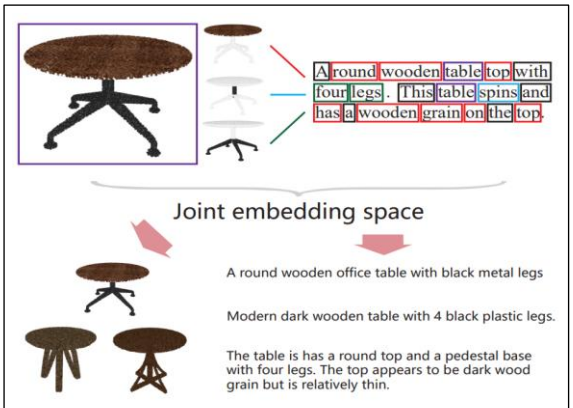


Рис. 3. Обработка текста с помощью алгоритма Part2Word

Fig. 3. Text processing using the Part2Word algorithm

Технология, подобная описанной, используется для приложений с 3D-графикой, где данные и модели могут присутствовать в произвольной ориентации. Такое изображение кодируется с учетом свойств 3D-формы наблюдаемого объекта, что в результате оказывается эквивалентным его трехмерному вращению (рис. 4).

Одним из важных факторов в генерации трехмерной модели является представление формы как деформации и комбинации обучаемых элементарных трехмерных структур, которые являются примитивами, полученными в результате обучения набору форм (рис. 5). Так реализована DeepVoxels [12] – инвариантная к точке обзора, постоянная и однородная 3D-сетка вокселей. Подобное представление примитивов приводит к улучшению и созданию более точных трехмерных форм [13].

Для изучения элементарных структур существуют два подхода: обучение деформации патчей и точечному переводу.

Оба подхода могут быть распространены на абстрактные структуры более высоких измерений для улучшения результатов. Главными задачами, стоящими при использовании данного метода, являются реконструкция объектов с помощью ShapeNet, а также оценка соответствий между сканами, сделанными вручную. В этом случае алгоритм использует набор облаков точек, по которому происходит обучение основам примитивов и более сложных моделей для последующей генерации в уже самостоятельный объект. Таким способом сначала происходит обучение набора облаков точек на конкретном примере, после чего подобные элементарные структуры используются для реконструкции объектов подобного или близкого по классификации типа из первоначальных данных (рис. 6).

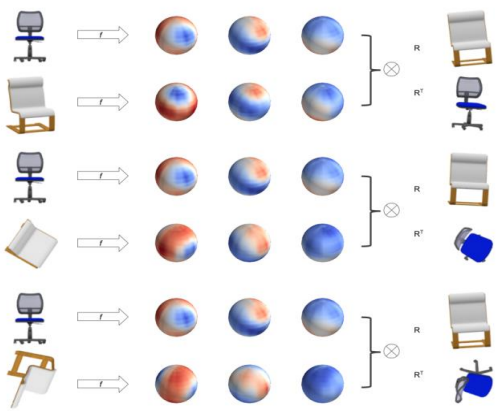


Рис. 4. Визуализация трехмерного вращения относительно позы

Fig. 4. Visualization of 3D rotation relative to a pose

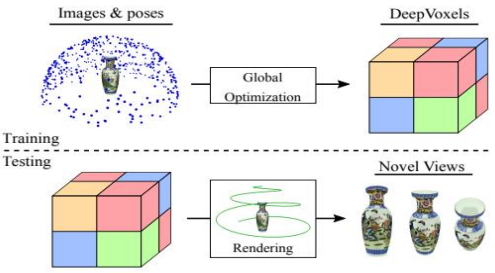


Рис. 5. Обучение DeepVoxels (сверху), синтез нового объекта (снизу)

Fig. 5. DeepVoxels training (top), new object synthesis (bottom)

Существует растущая область исследования, ориентированная на применение методов глубокого обучения к приложениям трехмерной геометрии и компьютерной графики. В компьютерном зрении структура данных очень проста: изображения состоят из плотных пикселей, которые представлены в виде массивов. В трехмерном пространстве подобный массив визуализируют с помощью вокселей, облаков точек, сеток, наборов многоракурсных изображений и т.д. Каждое из этих входных представлений также имеет свой собственный набор недостатков. Например, воксели имеют очень низкое разрешение, несмотря на высокую стоимость их вычисления. Облака точек не дают точное определение поверхностей или их нормалей, тем самым топология объекта не может быть однозначно получена из подобных

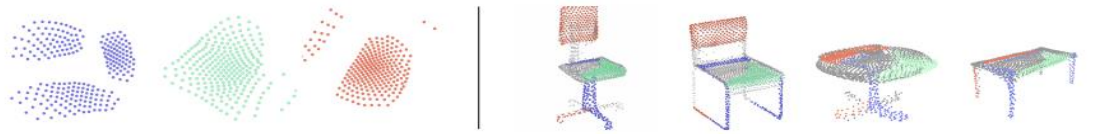


Рис. 6. Элементарные структуры стула (слева), 3D-реконструкции (справа)

Fig. 6. Elementary structures of a chair (left), a 3D reconstruction (right)

данных. Нейронная генеративная модель для сеток [14] PolyGen, которая оценивает совместно грани и вершины модели для непосредственного создания сеток, позволяет решить озвученные проблемы в подходах к реконструкции объектов. При этом используется уникальный подход к созданию модели – трехмерный объект представляется как строго упорядоченная последовательность вершин и граней, а не изображение в определенном ракурсе. Такой строгий порядок позволяет применять этот подход к моделированию последовательностей для создания трехмерных сеток.

Названное решение соединяет комплексные подходы в один и преобразует результат в трехмерную модель. Используя изображение объекта, воксели и сетку, можно создать вершины будущей модели, после чего все тем же способом определить грани и поверхности. На финальную модель накладываются все данные, и в результате с небольшой погрешностью получается сгенерированный объект (рис. 7).

Рис. 7. Генерация моделей с помощью PolyGen

Fig. 7. Generating models with PolyGen

Итак, большое количество успешных впечатляющих подходов, в том числе на основе обучения нейронных сетей, для реконструкции трехмерных моделей по ограниченной информации порождает новые идеи создания похожих алгоритмов для генерации компьютерной графики, например, для интерактивных проектов: компьютерных, включая серьезные, игр, тренажеров и симуляторов, AR/VR-приложений.

Генерация трехмерной сцены по текстовому запросу

Чтобы автоматически получать вариации трехмерных сцен на основании их текстового описания, авторы данного исследования предлагают использовать промежуточную структуризацию сущностей и их связей. Прежде чем приступить к самой генерации, нужно построить модели сущностей и связанных с ними ситуаций, которые могут быть обработаны с помощью нейронных сетей.

Весь алгоритм выделения этих сущностей и связей состоит из решения по отдельности совокупности довольно сложных подзадач (рис. 8):

– обнаружение в тексте каких-либо упоминаний о физических объектах;

– точная идентификация этих физических объектов;

– сопоставление физических объектов с записанным датасетом трехмерных моделей на основе названия объекта;

– оценка атрибутов каждого объекта (размер, пространственные ограничения, размещение объектов, цвет, текстура и др.) в соответствующих относительных положениях/ориентациях друг к другу (под, над, на, рядом, возле, около, недалеко и т.п.);

– генерация вариантов описываемого текстом размещения объектов на трехмерной сцене.

Входной текст, описывающий сцену, подается на обработку, результатом которой явля-

ется абстрактное представление шаблона сцены (рис. 9), фиксирующее объекты и отношения между ними.

Шаблон используется для создания конкретной сцены, визуализирующей входное описание, путем извлечения и размещения нужных моделей. Для такой генерации трехмерной сцены по текстовому запросу предлагается использовать комплекс перечисленных далее инструментов.

GoogleNews-vector-negative (https://code.google.com/archive/p/word2vec/) – датасет, содержащий 300-мерные векторы для трех миллионов слов и фраз, эти фразы были получены с использованием метода распределения входных данных и определения композиционности в них.

nlp-ue4 – нейронная сеть, способная работать совместно с игровым движком Unreal Engine и использовать формат для визуального программирования. Эта сеть упакована в виде плагина для удобного использования ее в сторонних классах Blueprint. Предшественницей этой нейронной сети была бесплатная и находящаяся в открытом доступе технология Microsoft LUIS (https://www.luis.ai/) для обработки текстов на естественном языке, которая взята за основу для реализации алгоритмов nlp-ue4. Важной особенностью nlp-ue4 является наличие функций по выделению сущностей (объектов) в предложениях и определению связей между ними.

TensorFlow Graphics (https://www.tensorflow.org/graphics) – инструмент, предназначенный для работы с 3D ML, имеется также расширение функционала в виде TensorBoard (https://www.tensorflow.org/tensorboard), который добавляет непосредственно отображение

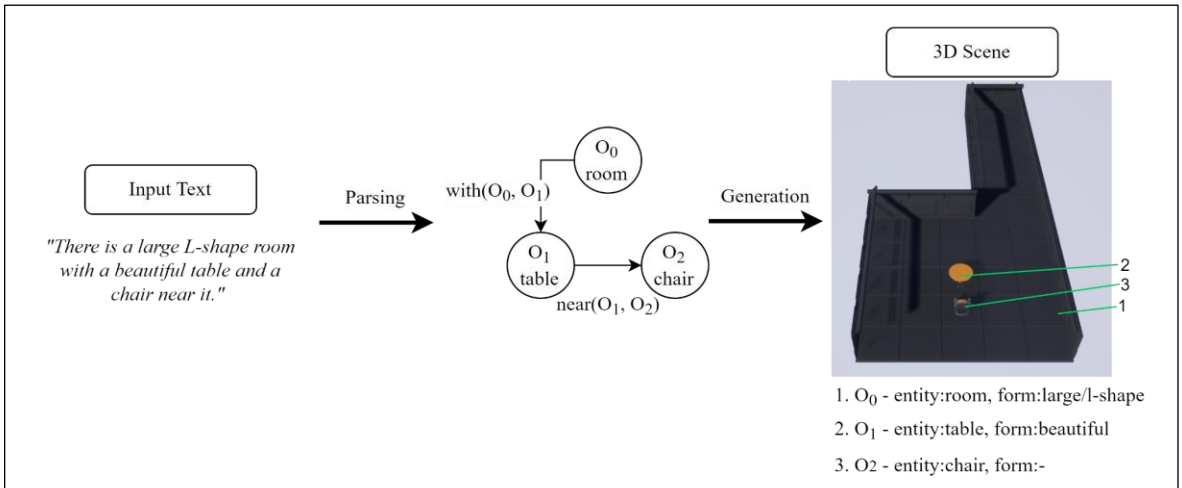


Рис. 8. Иллюстрация архитектуры алгоритма

Fig. 8. An algorithm architecture illustration

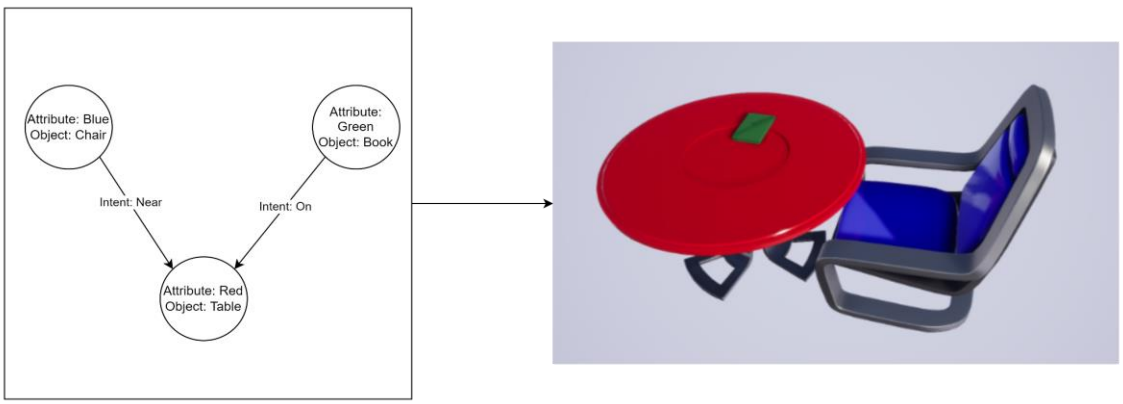


Рис. 9. Шаблон сцены

Fig. 9. A stage template

трехмерных моделей для визуализации данных.

NLTK (https://www.nltk.org/) – эффективная платформа для задач обработки текстов на естественном языке. Представляет собой простые интерфейсы для более чем 50 корпусов и лексических ресурсов, таких как WordNet (https://wordnet.princeton.edu/). Также в NLTK присутствуют наборы библиотек обработки текста для классификации, токенизации, выделения корней, тегов, синтаксического и семантического анализа, оболочки для промышленных библиотек NLP и прочие функции для обширной работы с текстом. Эта библиотека содержит в себе множество полезных и необходимых функций для обработки входных данных.

Pandas (https://pandas.pydata.org/) – быстрый, гибкий, мощный и простой в использовании инструмент для анализа и обработки данных. Библиотека является высокоуровневой, поскольку за ее основу взята низкоуровневая библиотека NumPy (https://numpy.org/), написанная на языке C, что делает Pandas более производительной, чем аналогичные инструменты. На сегодняшний день эта библиотека является наиболее гибкой, быстро развивающейся и продвинутой в области обработки текста на естественном языке.

Gensim (https://github.com/RaRe-Technologies/gensim) – библиотека обработки текстовых данных на естественном языке (подобна предыдущим), которая, помимо всего прочего, способна работать с векторными моделями данных из текста, а также создавать тематические модели на основе входных данных описания. Подобные операции называют тематическим моделированием – метод извлечения определенных тем в обрабатываемом тексте. В рассматриваемом случае извлечение тем необходимо для структурирования и описания основных тем и вычленения на этой основе определенных объектов с последующей классификацией и созданием связей между ними.

H5PY (https://docs.h5py.org/en/stable/) – интерфейс для двоичного формата HDF5, необходимый для записи и извлечения модели, полученной в результате обучения. HDF5 позволяет хранить огромные объемы числовых данных и легко манипулировать ими, используя Pandas. Тысячи наборов данных могут храниться в одном файле, классифицироваться и маркироваться с помощью имеющегося функционала инструмента.

Перечисленные технологии покрывают ту часть работы, которая касается обучения и обработки данных. Оставшиеся функции, непосредственно связанные с конечной интерпретацией данных и дальнейшей генерацией и наполнением трехмерной сцены, реализуют алгоритм, разработанный без использования дополнительных технологий и библиотек.

На рисунке 10 показано взаимодействие отдельных инструментов. Представленное комплексное решение исправляет множество недоработок, модернизируя структуризацию получаемого в ходе обработки текста результата в формате связей между найденными сущностями.

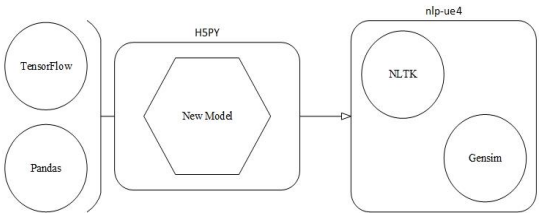


Рис. 10. Схема работы инструментов

Fig. 10. The scheme of the tools

Для построения массива данных об объектах необходимо ввести отдельные структуры:

– описание объекта $E_j = <N, a_1, \ldots, a_m{}^j>$, в котором находятся название N и массив атрибутов a_i, где m^j – количество выявленных атрибутов сущности E_j;

– связи между объектами $R_{i,j} = <p_{ij}, E_i, E_j>$, где p_{ij} – параметр, отвечающий за расположение объекта E_i по отношению к центральному объекту E_j (уже находящемуся на сцене, по отношению к которому уточняется расположение объекта E_i).

В результате получается разреженная матрица R, где содержатся все связи между объектами, участвующие в действиях друг с другом: $R = \{R_{i,j}, i, j \in (0, K]\}$, K – количество выявленных сущностей сцены.

Сущности E_t и E_q, содержащие в качестве центра одинаковый объект E_j с $p_{tj} = p_{qj}$, размещаются вместе – на одной поверхности центрального объекта (рис. 11).

Весь процесс проверки свободного места осуществляется с помощью обхода ячеек целочисленного массива данных. Подобный метод включает в себя сохранение клеток, уже занятых сущностями, и распределение на следующие подходящие варианты.

Выбор точки появления определенной сущности вычисляется по радиусу случайных значений, которые варьируются от $[X - \frac{Scale}{4};$

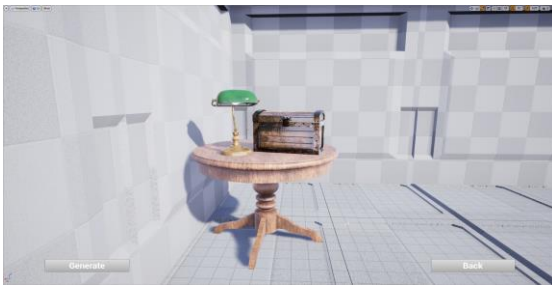


Рис. 11. Сгенерированные связи между объектами на столе

Fig. 11. Generated links between the objects on the table

$Y - \frac{Scale}{4}]$ до $[X + \frac{Scale}{4}; Y + \frac{Scale}{4}]$, где X, Y – координаты центра выбранной подматрицы; *Scale* – параметр, отвечающий за размер одной клетки. Таким образом, расстановка объектов происходит в центральной части подматрицы со случайной погрешностью в заданном интервале. При вычислении расположения сущности относительно уже находящейся в клетке объекта сначала выбирается случайная сторона из четырех областей в клетке, затем выбранная область делится еще на три части и снова случайным образом выбирается одна из них (рис. 12).

После завершения процесса распределения местоположения объектов, задействованных в описании текста, происходит визуализация готовой сцены с последующей возможностью перераспределения всех сущностей в другом порядке, включая форму рассматриваемой комнаты. Последовательность процессов внутри инструмента такова:

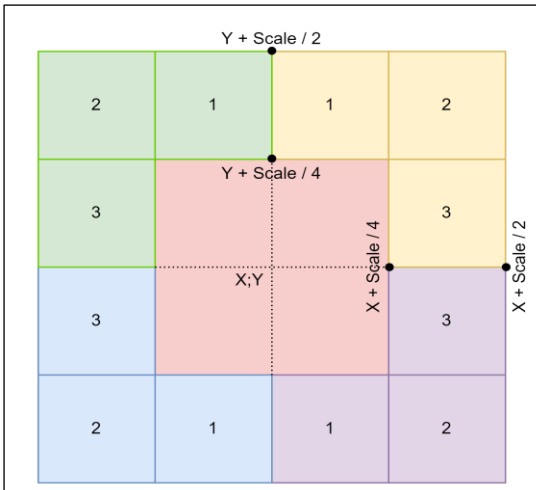


Рис. 12. Алгоритм нахождения точки появления объекта в клетке

Fig. 12. An algorithm for finding the point of object appearance in a cell

– первичное обучение модели на основе табличных данных;

– ввод текстовых данных описания сцены;

– обработка полученного сообщения нейронной сетью на основе обученной модели;

– вывод результата в формате сущностей и связей между ними;

– обработка и структуризация объектов;

– сбор атрибутов для каждой полученной сущности;

– визуализация трехмерной сцены.

В результате из исходного текстового описания сцены получается массив, состоящий из структур данных, где каждый элемент содержит связь между объектами (Intent) и структуру сущности (Entity), состоящую из названия объекта и массива атрибутов.

Для генерации сцены используется этот массив с обходом каждого элемента в строке. При упоминании, например, стола с атрибутом small рядом со стеной алгоритм выставит маленький стол рядом со стеной. Подобным образом происходит и размещение объектов около друг друга, друг на друге или в другой конфигурации. Для этого добавляется дополнительный параметр во внешней таблице, где хранятся модели сущностей и их размеры, в виде высоты объекта и погрешности для выставления более мелких предметов на нем (рис. 9).

Обсуждение результатов

На данный момент алгоритм способен принимать текст любого объема, но существует ограничение на максимальное количество сущностей и связей между ними, указанных в одном предложении (не более 10).

Алгоритм способен генерировать сцену в размере от 4 до 30 клеток в зависимости от описания сцены и ограничивается параметрами закрытого помещения.

Алгоритм способен генерировать по запросу около пяти вариаций сцен в секунду.

Для иллюстрации работы инструмента проведены эксперименты, в результате которых были сгенерированы сцены по текстовому описанию. В ходе экспериментов были введены описания локации, включающие в себя объекты, находящиеся в ней, и связи между ними. Также был запущен процесс генерации 10 трех-

мерных сцен. Параллельно с этим воспроизводились создание такой же комнаты вручную и расстановка всех объектов, в том числе фиксировались границы этой локации в виде стен и пола.

Эксперимент А. *There is a small room. A table stands near the wall. A small chair near this table.* (Это маленькая комната. Стол находится рядом со стеной. Маленький стул стоит рядом со столом.)

Результатом стала трехмерная сцена, выбранная из сгенерированных 10 сцен (субъективно лучшая, в целом все полученные сцены хороши) (рис. 13а).

В процессе обработки приведенного текста получены следующие сущности, их атрибуты и связи:

Room (small) – Table–NEAR–Wall
Table – Chair–NEAR–Table
Chair (small) – нет.

Общее затраченное время на создание 10 вариаций сцен – 2 секунды. Общее затраченное время на сборку сцены вручную – 6 минут.

Эксперимент Б. *A small room. A shelf somewhere. A large book on a shelf. A table near the wall. A chair stands near the table. A lamp lies on the table and a small book lies on a table. A large chest stands near the wall.* (Маленькая комната. Полка находится где-то. Большая книга на полке. Стол стоит рядом со стеной. Стул находится рядом с этим столом. Лампа и маленькая книга лежат на столе. Большой сундук находится рядом со стеной.)

Результат: трехмерная сцена (рис. 13б) – генерация одной из 10 сцен.

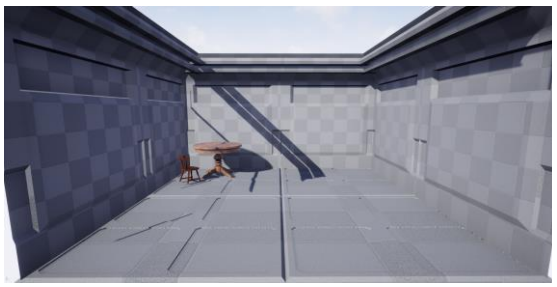
В процессе обработки текста получены следующие сущности, их атрибуты и связи:

Room (small) – Shelf–RANDOM–Room
Shelf – Book (L) –ON–Shelf
Book (large) – Table–NEAR–Wall
Table – Chair–NEAR–Table
Chair – Lamp–ON–Table
Lamp – Book (S) –ON–Table
Book (small) – Chest–NEAR–Wall
Chest – нет.

Общее затраченное время на создание 10 вариаций сцен – 2 секунды. Общее затраченное время на сборку сцены вручную – 30 минут.

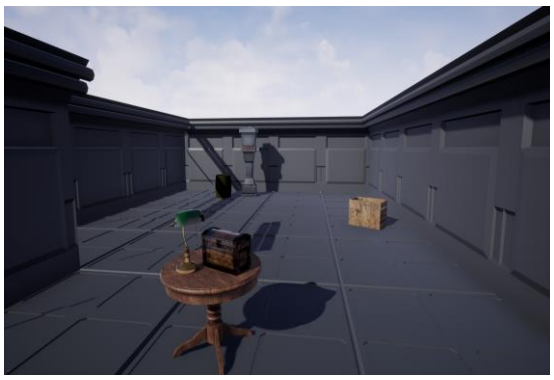
Эксперимент В. *There is a large l-shape room. A table stands somewhere. A small chest lies on the table and a large lamp on the table. Somewhere, a plank stands alone. A pillar stands near one of the wall and a barrel near the pillar.* (Это большая L-образная комната. Стол находится где-то. Маленький сундук лежит на столе вместе с большой лампой. Где-то находится одинокая доска. Столб находится рядом со стеной, и бочка стоит рядом со столбом.)

Полученный результат: трехмерная сцена (рис. 13в) – генерация одной из 10 сцен.



а)

б)



в)

Рис. 13. Пример генерации одной из 10 сцен: а) эксперимент А, б) эксперимент Б, в) эксперимент В

Fig. 13. An example of generating one of 10 scenes: a) A experiment, б) Б experiment, в) B experiment

В процессе обработки текста получены следующие сущности, их атрибуты и связи:

Room (large/l-shape) – Table–RANDOM–Room
Table – Chest–ON–Table
Chest (small) – Lamp–ON–Table

Lamp (large)	–	Plank–RANDOM–Room
Plank	–	Pillar–NEAR–Wall
Pillar	–	Barrel–NEAR–Plank
Barrel	–	нет.

Общее затраченное время на создание 10 вариаций сцен – 2 секунды. Общее затраченное время на сборку сцены вручную – 30 минут.

Заключение

Использование разработанного инструмента для получения трехмерных сцен на основе исходного текстового описания значительно сокращает время разработки в отличие от ручной сборки данных. Инструмент предоставляет огромное множество вариантов сборки сцен, что упрощает работу при необходимости создания альтернативных вариантов сцен на основе одного и того же текстового описания.

Важно отметить, что для работы алгоритма не требуются высокие мощности компьютера, поскольку он использует уже обученную модель на заранее введенных данных. При работе с инструментом от пользователей не требуются глубокие технические знания в области трехмерного моделирования и программирования.

Инструмент может быть кардинально улучшен увеличением максимального количества сущностей в одном предложении, а расширение сущностей и атрибутов во внутреннем датасете может значительно увеличить функциональность и гибкость инструмента в решении пользовательских задач.

Работа выполнена за счет средств Программы стратегического академического лидерства Казанского (Приволжского) федерального университета («ПРИОРИТЕТ-2030»).

Литература

1. Chen K., Choy Ch., Savva M., Chang A., Funkhouser Th., Savarese S. Text2Shape: Generating shapes from natural language by learning joint embeddings. In: Computer Vision – ACCV, 2018, pp. 100–116. DOI: 10.1007/978-3-030-20893-6_7.

2. Oscar M., Roi B., Richard L., Sagie B., Rana H. Text2Mesh: Text-driven neural stylization for meshes. IEEE Computer Society. Proc. CVPR, 2021, pp. 13492–13502.

3. Zhengfei K., Kyle O., Menglei Ch., Zeng H., Panos A., Sergey T. NeROIC: Neural rendering of objects from online image collections. ArXiv, 2022, vol. abs/2201.02533, pp. 1–17. URL: https://www.semanticscholar.org/reader/1372924688f627eacbc1e8c97d33399c1d648309 (дата обращения: 10.07.2022).

4. Chang A., Monroe W., Savva M., Potts C., Manning C.D. Text to 3D Scene generation with rich lexical grounding. Proc. LIII Annual Meeting ACL, 2015, vol. 1, pp. 53–62. DOI: 10.3115/v1/P15-1006.

5. Кугуракова В.В., Сахибгареева Г.Ф., Нгуен А.З., Астафьев А.М. Пространственная ориентация объектов на основе обработки текстов на естественном языке для генерации раскадровок // Электронные библиотеки. 2020. Т. 23. № 6. С. 1213–1238. DOI: 10.26907/1562-5419-2020-23-6-1213-1238.

6. Michel O., Bar-On R., Liu R., Benaim S., Hanocka R. Text2Mesh: Text-Driven Neural Stylization for Meshes. ArXiv, 2021, vol. abs/2112.03221, pp. 1–15. URL: https://www.semanticscholar.org/reader/d15b27edf3630728cdb40f49946365d9011641cf (дата обращения: 10.07.2022).

7. Mildenhall B., Srinivasan P.P., Tancik M., Barron J.T., Ramamoorthi R., Ng R. NeRF: Representing scenes as neural radiance fields for view synthesis. In: Computer Vision – ECCV, 2020, vol. 12346 LNCS, pp. 405–421. DOI: 10.1007/978-3-030-58452-8_24.

8. Gadelha M., Maji S., Wang R. 3D Shape induction from 2D views of multiple objects. Proc. Int. Conf. 3DV, 2017, pp. 402–411. DOI: 10.1109/3DV.2017.00053.

9. Rueegg N., Zuffi S., Schindler K., Black M.J. BARC: Learning to regress 3D dog shape from images by exploiting breed information. ArXiv, 2022, pp. 1–9. DOI: 10.48550/arXiv.2203.15536. URL: https://barc.is.tue.mpg.de/media/upload/00931_notchecked_c.pdf (дата обращения: 10.07.2022).

10. Tang C., Yang X., Wu B., Han Z., Chang Y. Part2Word: Learning joint embedding of point clouds and text by matching parts to words. ArXiv, 2021, pp. 1–9. URL: https://www.semanticscholar.org/reader/726c93dad32d5010b5ffbb9ecbba7a97b714881e (дата обращения: 10.07.2022).

11. Esteves C., Sud A., Luo Z., Daniilidis K., Makadia A. Cross-domain 3D equivariant image embeddings. ICML, 2018, pp. 1–15.

12. Sitzmann V., Thies J., Heide F., Niebner M., Wetzstein G., Zollhofer M. DeepVoxels: Learning persistent 3D feature embeddings. Proc. IEEE/CVF CVPR, 2019, pp. 2432–2441. DOI: 10.1109/CVPR.2019.00254.

13. Deprelle T., Groueix T., Fisher M., Kim V.G., Russell B.C., Aubry M. Learning elementary structures for 3D shape generation and matching. In: Advances in Neural Information Processing Systems, 2019, pp. 7433–7443.

14. McGough M. Generating 3D models with PolyGen and PyTorch. Towards Data Science, 2020. URL: https://towardsdatascience.com/generating-3d-models-with-polygen-and-pytorch-4895f3f61a2e (дата обращения: 10.07.2022).

Software & Systems
DOI: 10.15827/0236-235X.139.329-339

Received 27.07.22, Revised 05.08.22
2022, vol. 35, no. 3, pp. 329–339

Structuring natural text entities using neural networks for generating 3D-scenes

B.A. Kozar [1], *Engineer, bogdan.kozar.itis@gmail.com*
V.V. Kugurakova [1], *Ph.D. (Engineering), Associate Professor, vlada.kugurakova@gmail.com*
G.F. Sakhibgareeva [1], *Assistant, gulnara.sahibgareeva42@gmail.com*

[1] *Kazan Federal University, Kazan, 420008, Russian Federation*

Abstract. The subject of this study is the automation using neural networks of the process of assembling three-dimensional scene, which can be used both to generate three-dimensional scenes or locations in computer games from a textual description, and to prepare sequences of three-dimensional synthetic data. This topic is relevant for developing three-dimensional graphics including interactive projects – games, simulators, AR/VR applications.

After analyzing and comparing the results obtained in a number of well-known completed projects, the authors determine technologies and software libraries, which allow effectively achieving the desired goal - to provide fast assembly of three-dimensional scenes filled with objects according to the text description. Thanks to synthesis of the best solutions, it was possible to create an optimal concept that allows achieving quick and qualitative result with the right rules of building geometrical relations between scene objects. There is a formed list of requirements to the designed tool and its architecture. Input data for using this tool is a text in natural language; output data is a scene with objects corresponding to the description used.

The main result achieved is a finished software tool for Unreal Engine developed on the basis of the nlp-ue4 neural network and the set of tensorflow, nltk, pandas, gensim, h5py libraries. The readiness of the tool is evaluated as a prototype solution, which can be integrated into the drafting stage of interactive projects with three-dimensional graphics.

To evaluate the created tool effectiveness objectively, the authors have conducted the experiments that proved that its use even in the current version significantly reduces development time and does not require a user to have skills in programming or creating three-dimensional graphics.

There is also a discussion about the research development prospects.

Keywords: computer graphics, 3D modeling, visualization, procedural generation, level design, automation, NLP, neural networks, Unreal Engine, computer games.

Acknowledgements. *This paper has been supported by the Kazan Federal University Strategic Academic Leadership Program ("PRIORITY-2030").*

References

1. Chen K., Choy Ch., Savva M., Chang A., Funkhouser Th., Savarese S. Text2Shape: Generating shapes from natural language by learning joint embeddings. In: *Computer Vision – ACCV*, 2018, pp. 100–116. DOI: 10.1007/978-3-030-20893-6_7.

2. Oscar M., Roi B., Richard L., Sagie B., Rana H. Text2Mesh: Text-driven neural stylization for meshes. *IEEE Computer Society. Proc. CVPR*, 2021, pp. 13492–13502.

3. Zhengfei K., Kyle O., Menglei Ch., Zeng H., Panos A., Sergey T. NeROIC: Neural rendering of objects from online image collections. *ArXiv*, 2022, vol. abs/2201.02533, pp. 1–17. Available at: https://www.semanticscholar.org/reader/1372924688f627eacbc1e8c97d33399c1d648309 (accessed July 10, 2022).

4. Chang A., Monroe W., Savva M., Potts C., Manning C.D. Text to 3D Scene generation with rich lexical grounding. *Proc. LIII Annual Meeting ACL*, 2015, vol. 1, pp. 53–62. DOI: 10.3115/v1/P15-1006.

5. Kugurakova V.V., Sakhibgareeva G.F., Nguen A.Z., Astafiev A.M. Spatial orientation of objects based on processing of a natural language text for storyboard generation. *Russian Digital Libraries J.*, 2020, vol. 23, no. 6, pp. 1213–1238. DOI: 10.26907/1562-5419-2020-23-6-1213-1238.

6. Michel O., Bar-On R., Liu R., Benaim S., Hanocka R. Text2Mesh: Text-Driven Neural Stylization for Meshes. *ArXiv*, 2021, vol. abs/2112.03221, pp. 1–15. Available at: https://www.semanticscholar.org/reader/d15b27edf3630728cdb40f49946365d9011641cf (accessed July 10, 2022).

7. Mildenhall B., Srinivasan P.P., Tancik M., Barron J.T., Ramamoorthi R., Ng R. NeRF: Representing scenes as neural radiance fields for view synthesis. In: *Computer Vision – ECCV*, 2020, vol. 12346 LNCS, pp. 405–421. DOI: 10.1007/978-3-030-58452-8_24.

8. Gadelha M., Maji S., Wang R. 3D Shape induction from 2D views of multiple objects. *Proc. Int. Conf. 3DV*, 2017, pp. 402–411. DOI: 10.1109/3DV.2017.00053.

9. Rueegg N., Zuffi S., Schindler K., Black M.J. BARC: Learning to regress 3D dog shape from images by exploiting breed information. *ArXiv*, 2022, pp. 1–9. DOI: 10.48550/arXiv.2203.15536. Available at: https://barc.is.tue.mpg.de/media/upload/00931_notchecked_c.pdf (accessed July 10, 2022).

10. Tang C., Yang X., Wu B., Han Z., Chang Y. Part2Word: learning joint embedding of point clouds and text by matching parts to words. *ArXiv*, 2021, pp. 1–9. Available at: https://www.semanticscholar.org/reader/726c93dad32d5010b5ffbb9ecbba7a97b714881e (accessed July 10, 2022).

11. Esteves C., Sud A., Luo Z., Daniilidis K., Makadia A. Cross-domain 3D equivariant image embeddings. *ICML*, 2018, pp. 1–15.

12. Sitzmann V., Thies J., Heide F., Niebner M., Wetzstein G., Zollhofer M. DeepVoxels: Learning persistent 3D feature embeddings. *Proc. IEEE/CVF CVPR*, 2019, pp. 2432–2441. DOI: 10.1109/CVPR.2019.00254.

13. Deprelle T., Groueix T., Fisher M., Kim V.G., Russell B.C., Aubry M. Learning elementary structures for 3D shape generation and matching. In: *Advances in Neural Information Processing Systems*, 2019, pp. 7433–7443.

14. McGough M. Generating 3D models with PolyGen and PyTorch. *Towards Data Science*, 2020. Available at: https://towardsdatascience.com/generating-3d-models-with-polygen-and-pytorch-4895f3f61a2e (accessed July 07, 2022).

Для цитирования

Козар Б.А., Кугуракова В.В., Сахибгареева Г.Ф. Структуризация сущностей естественного текста с использованием нейронных сетей для генерации трехмерных сцен // Программные продукты и системы. 2022. Т. 35. № 3. С. 329–339. DOI: 10.15827/0236-235X.139.329-339.

For citation

Kozar B.A., Kugurakova V.V., Sakhibgareeva G.F. Structuring natural text entities using neural networks for generating 3D-scenes. *Software & Systems*, 2022, vol. 35, no. 3, pp. 329–339 (in Russ.). DOI: 10.15827/0236-235X.139.329-339.