Автоматизация и моделирование в проектировании и управлении. 2022. № 4 (18). С. 18-28.
ISSN 2658-3488 print, ISSN 2658-6436 online
Automation and modeling in design and management. 2022. № 4 (18). P. 18-28.

Научная статья
Статья в открытом доступе
УДК 004.93'14
DOI 10.30987/2658-6436-2022-4-18-28

ФОРМИРОВАНИЕ СИНТЕТИЧЕСКИХ ДАННЫХ ДЛЯ ОБУЧЕНИЯ СИСТЕМЫ КОМПЬЮТЕРНОГО ЗРЕНИЯ

Денис Александрович Копылов [1], Егор Сергеевич Агешин [2], Ольга Владиславовна Хомутская [3]

[1, 2, 3] Московский Авиационный институт (Национальный исследовательский университет), г. Москва, Россия

[1] den210799@gmail.com, http://orcid.org/0000-0002-0138-3950
[2] ageshin.e@mail.ru, http://orcid.org/0000-0002-3939-5690
[3] khomutskayaov@gmail.com, http://orcid.org/0000-0003-1814-6334

Аннотация. *Приведен метод формирования синтетических данных для обучения нейронной сети (далее – нейросеть) распознавать существующие объекты. Данный метод призван упростить процесс составления начального набора данных и его изменения для дальнейшего использования в компьютерном зрении. В качестве образца объекта для распознавания используется напечатанный с помощью аддитивных технологий редуктор авиационного двигателя. Трехмерные модели загружались в трехмерный редактора Houdini, где с помощью подпрограммы (далее – скрипт) на Python сохранялась коллекция скриншотов детали на разном фоне. Полученный набор данных использовался для обучения трех нейронных сетей на сайте Roboflow, а полученные результаты анализировались для возможности дальнейшего использования данного метода. В статье подробно показан процесс создания скриншотов и результат распознавания напечатанной детали с помощью трех нейронных сетей.*

Ключевые слова: распознавание объекта, компьютерное зрение, двигателестроение, трехмерный редактор, Houdini, Python, нейронные сети, машиностроение, производство

Для цитирования: Копылов Д.А., Агешин Е.С., Хомутская О.В. Формирование синтетеических данных для обучения системы компьютерного зрения // Автоматизация и моделирование в проектировании и управлении. 2022. №4 (18). С. 18-28. doi: 10.30987/2658-6436-2022-4-18-28.

Original article

Open Access Article

FORMING SYNTHETIC DATA FOR TRAINING A COMPUTER VISION SYSTEM

Denis A. Kopylov[1], Egor S. Ageshin[2], Olga V. Khomutskaya[3]
[1,2,3] Moscow Aviation Institute, Moscow, Russia
[1]den210799@gmail. com, [2]ageshin.e@mail.ru, [3]khomutskayaov@gmail.com

Abstract. *The article presents a method for generating synthetic data for training a neural network (hereinafter referred to as a neural network) to recognize existing objects. This method is designed to simplify the process of compiling the initial data set and modifying it for further application in the computer vision. An aircraft engine gearbox printed using additive technologies is used as a sample object for recognition. Three-dimensional models are loaded into Houdini three-dimensional editor, where a screenshot collection of the part on different backgrounds is saved using a sub-programme (hereinafter referred to as script) in Python. The received data set is applied to train three neural networks on the Roboflow website, and the results obtained are analysed for the possibility of using this method further. The article shows in detail the process of creating screenshots and the result of recognizing a printed part using three neural networks.*

Keywords: object recognition, computer vision, engine building, 3D editor, Houdini, Python, neural networks, mechanical engineering, manufacturing.

For citation: Kopylov D.A. Ageshin E.S., Khomutskaya O.V. Forming synthetic data for training a computer vision system. Automation and modeling in design and management, 2022, no. 4 (18). pp. 18-28. doi: 10.30987/2658-6436-2022-4-18-28.

Введение

Современные датасеты (dataset – набор данных) изображений представляют собой сотни тысяч изображений с десятками классов, а данные для таких наборов отбираются и размечаются в течение длительного периода времени, что может стоить больших финансовых затрат. В ряде областей исследователи начинают применять синтетические исходные данные для обучения нейросетей, чтобы повысить качество получаемых прогнозов на действительных изображениях. Исследователи из Медицинской школы Гарварда совместно с учеными из MIT создали изображения подтипов почечно-клеточного рака [1]. Такие изображения должны избавить ученых не только в необходимости обработки большого количества сложной медицинской информации из клиник, но и сделать исходные данные более прозрачными. Синтетические изображения создавались с помощью GAN-сетей на основе подтвержденных случаях почечно-клеточного рака. В результате удалось получить большое количество похожих изображений, которые в дальнейшем могут быть использованы для диагностики заболевания (рис.1).

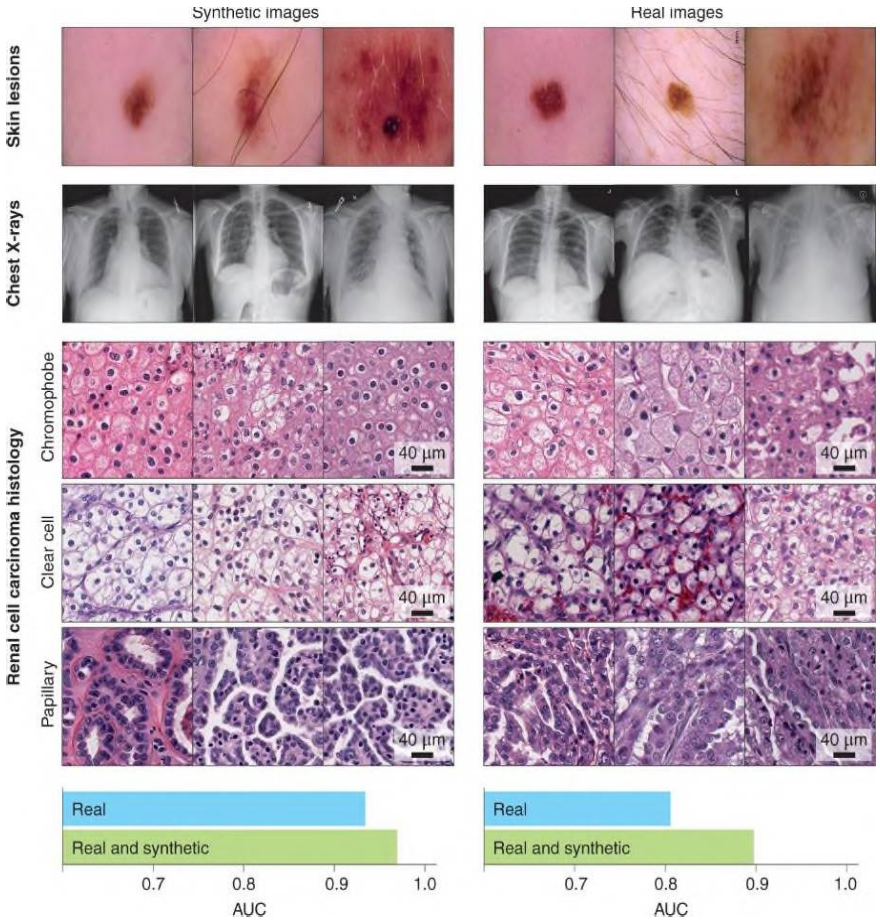

Рис. 1. Сравнение настоящих и искусственных изображения для диагностики рака
Fig. 1. Comparison of Real and Artificial Images for Cancer Diagnosis

Другим примером может служить применение созданных изображений для обучения нейросети выявлять повреждения и износ конструкций на промышленных объектах [2],[3]. Инженеры компании PHYGITALISM с помощью редактора Blender3D создали генератор моделей ржавых и поврежденных труб, и на основании полученных изображений обучили нейронную сеть выявлять дефекты трубопроводов через камеру квадрокоптера на настоящих промышленных объектах (рис.2).

Кроме медицинских проблем и задач дефектоскопии создание искусственных данных активно применяется при разработке программного обеспечения для автономных машин и такси, распознавания человеческих поз [4]. Поскольку обработать информацию о ДТП с участием людей крайне затруднительно, а проводить имитации с пешеходами и водителями даже на специальных автодромах опасно для здоровья испытуемых, на помощь может прийти создание необходимых данных даже на основании объектов в компьютерных играх.

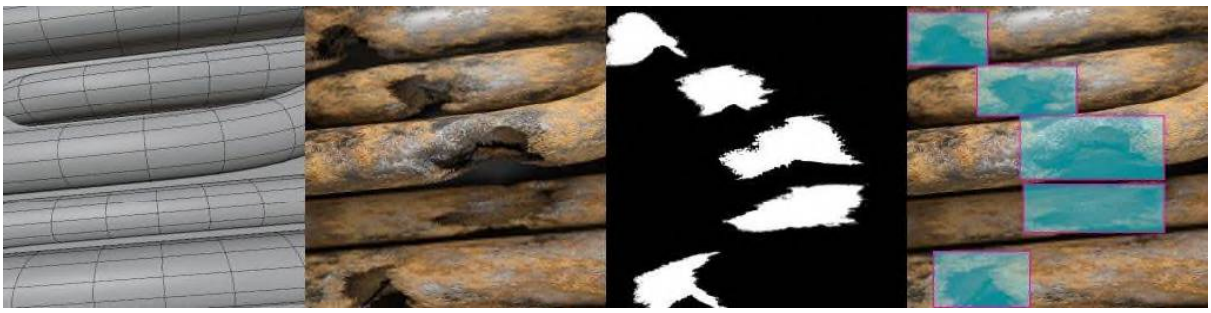

Рис. 2. Процесс моделирования поврежденных труб в трехмерном редактор для дальнейшего формирования исходных данных для нейросети
Fig. 2. The process of modeling damaged pipes in a three-dimensional editor for further generation of initial data for a neural network

Такой подход реализовали исследователи из TUM (ФРГ) и Университета Модены и Реджо-Эмилии (Италия) [5]. Инженеры создали синтетических набор данных MOTSynth на основании игры Grand Theft Auto V (GTA 5) и применили его для слежения за пешеходами в системе автопилота машины. Несмотря на то, что исходные данные, созданные только из синтетических изображений, не отвечали ожидаемым результатам исследования, набор из смешанных данных показал хорошие показатели.

Опыт безопасного для человека и оборудования сбора данных применяется и в промышленности, которая использует робототехнические комплексы на производстве. Роботы-манипуляторы в большинстве случаев запрограммированы на выполнение монотонных действий и не могут распознать неправильно расположенные объекты или нахождение людей в рабочей зоне. Чтобы робот смог выполнять поставленные задачи его оборудуют системой распознавания, которая была обучена анализировать объекты вокруг. Способность верно распознавать расположение объекта и его ориентацию повышает эффективность и безопасность выполнения технологических процессов [6].

В области машиностроения получение полноценного набора данных, еще и в реальных условиях, представляет собой практически нерешаемую задачу из-за труднодоступности или отсутствия законченного объекта.

В связи с чем имеет смысл прибегнуть к методам синтеза данных, т.к. такой подход может иметь существенные преимущества:

– неограниченный объем данных;
– создание датасетов для труднодоступных объектов;
– создание анонимных данных;
– создание разметки данных на этапе синтеза.

Стоит также отметить и недостатки метода:

– не всегда синтезированные данные обладают особенностями реального объекта, которые могут оказать существенное влияние на результаты применения алгоритмов машинного обучения;

– синтез наборов данных происходит при помощи различных алгоритмов, которые в свою очередь создаются человеком;

– автоматизация разметки так же влечет за собой сложности и дополнительные ошибки.

Подготовка к работе
Подход к созданию изображений

Современные программы для работы с трехмерной графикой обладают широкими возможностями для синтеза реалистичных фото и видео. Это можно наблюдать по визуальным эффектам, которые применяют в киноиндустрии, компьютерных играх.

Такое возможно благодаря современным алгоритмам трассировки лучей, которые лежат в основе рендер двигателей. Рендер – процесс преобразования трехмерного объекта и простых геометрических фигур в реалистичную картинку. Создание фотореалистичной картинки представляет собой отдельную задачу и в данной статье рассматриваться не будет.

В данной работе используется метод генерации изображений в трехмерных редакторах, который также применялся в [2],[7].

Выбор трехмерного редактора

В данной работе процесс обработки трехмерных моделей происходил в пакете трехмерной графики Houdini. Ключевой особенностью данной среды является наличие визуального программирования, что позволяет автоматизировать процесс работы с графикой. Более того, редактор имеет возможность писать скрипты на языках C++ и Python. Второй важной особенностью является возможность работы без графического интерфейса, что удобно при последующей установке данного пакета на сервер. Последней особенностью, важной для задачи синтеза изображений, является встроенный визуализатор (рендера), который создает изображение высокого качества.

Единственным недостатком данного визуализатора является работа только на вычислительных мощностях центрального процессора (CPU), в то время как современные аналоги имеют возможность работать на графическом процессоре (GPU). Такая особенность значительно замедляет скорость выполнения задач.

Совокупность перечисленных факторов с хорошо описанной и проработанной документацией позволяет использовать данный пакет для синтеза изображений в настоящем проекте.

Основы работы с Houdini

Основная работа в Houdini осуществляется с помощью контекстов (context). Всего в Houdini представлено 8 контекстов: ch, img, mat, obj, out, shop, stage и tasks. В данной проекте достаточно ограничиться только тремя:

– geo – данный контекст находится в более общем контексте Obj и позволяет работать с геометрией;

– out – контекст, отвечающий за рендер и постобработку;

– mat – контекст работы с материалами.

Работа в редакторе происходит при помощи узлов (nodes) и связей между ними (wires). При помощи линии связи данные передаются на вход узла и далее во все дочерние подузлы, в которых происходит модификация или обработка данных.

Такой подход позволяет получить следующие преимущества:

– Легкое понимание работы, так как сцена фактически представляет собой усложненную блок-схему;

– Возможность создания пользовательских блоков, к которым нужно только подключить необходимые объекты;

– Возможность процедурного подхода, так как достаточно только следить за передаваемой между узлами информации.

Отдельно необходимо отметить глубокую интеграцию языка Python в данный 3*D*-редактор. Все узлы, представленные в Houdini чаще всего представлены либо Python-скриптом, либо C-подобным внутренним языком, которые реализуют математические операции с данными. Такой подход позволяет представить узлы в необходимом виде для текущей задачи.

Более подробно с работой в Houdini можно ознакомиться на официальном сайте Houdini в разделе документации [8].

Процесс создания изображений

Процесс создания изображения можно представить в виде следующих этапов, представленных на рис. 3.

Для создания изображений необходимо подготовить сцену внутри Houdini. Данный этап состоит из следующих шагов:

1. Создание алгоритма перемещения камеры вокруг точки. В точку затем будет помещен объект для создания датасета;

2. Создание инструмента «наложение Bounding Box» на 3*D* объект и последующий перевод 3-мерных координат в координаты на 2-мерном пространстве камеры;

3. Создание Python скрипта для получения координат Bounding Box на синтезированном изображении и запись этих координат в специальном формате для разметки данных;

4. Создание структуры передачи данных между созданными элементами в Houdini.

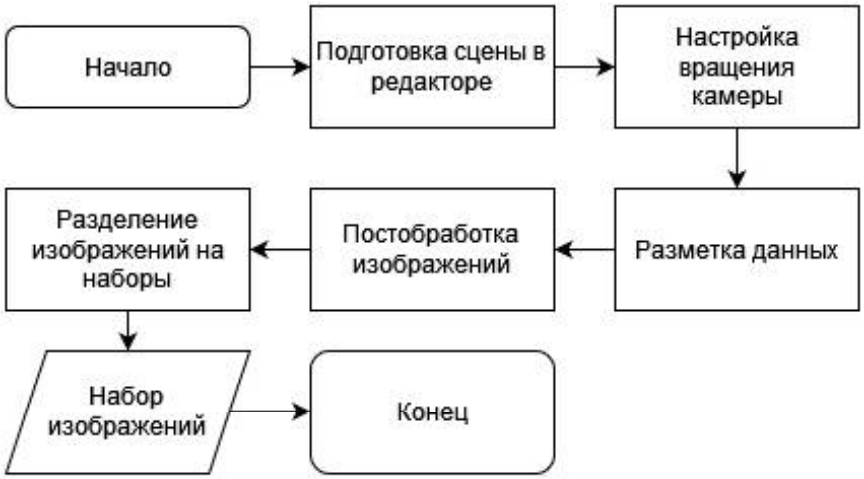

Рис. 3. Этапы составления набора данных для обучения нейросети
Fig. 3. Stages of compiling a dataset for training a neural network

Начальные настройки редактора

После подготовки инструментов необходимый объект подгружается для создания датасета и начинается процесс создания рендера. На входе рендера в каждом кадре (их количество в данном случае равно размеру датасета для объекта) выбирается случайная точка на расстоянии величины радиуса от объекта, в которую затем помещается камера, и визуализатор осуществляет «запекание» объекта в кадр. Кадр сохраняется вместе с информацией о разметке.

Вращение камеры

Для создания алгоритма вращения камеры вокруг детали необходимо создать сферу вокруг центра координат, внутренний радиус которой в 1,5 раза больше максимальной высоты объекта, а внешний радиус – в 2,5 раза больше. Таким образом, из любых точек сферы будет обеспечен приемлемый обзор на объект. Данные размеры получены в ходе работы и испытаний разных геометрических параметров для разных 3*D*-объектов.

После создание сферы необходимо удалить все полигоны, оставляя лишь точки, в которые будет помещаться виртуальная камера. Количество точек предполагается сопоставимым с количеством объектов готовящегося датасета. Для достижения большего разнообразия ракурсов рекомендуется использовать количество точек в 2-3 раза превышающее количества объектов планируемого набора данных.

Так как камера представляет собой точку с направляющим вектором, то у каждой точки дополнительно создана система из трёх векторов, один из которых направлен к центру сферы. По этому вектору выставляется направление камеры. Дополнительной сложностью является метод управления геометрией в Houdini: для того, чтобы повернуть объекты нужно передавать углы Ейлера, поэтому для каждой точки дополнительно рассчитаны углы поворота.

Разметка данных

Следующим этапом является разметка. В рамках данной работы выбор пал на разметку с помощью Bounding Box – область на изображении, ограничивающая объекты на изображении, которая имеет принадлежность к классам. Далее данные о координатах и метки класса используется при обучении нейросети.

Для создания автономной системы разметки необходимо осуществить следующие этапы:

– Создание трехмерной Bounding Box вокруг объемной модели;
– Преобразовать полученные координаты плоскость камеры.

Одной из сложностей в работе была в трансформации объемных координат в координаты плоскости камеры, которая решается применением специальных настроек в встроенном модуле Houdini. Так же был создан отдельный Python скрипт, который после обработки изображения сохраняет координаты Bounding Box в текстовый файл (рис. 4).

```
0 0.5709134615384616 0.36658653846153844 0.18028846153846154 0.31490384615384615
```

Рис. 4. Пример координат Bounding Box объекта
Fig. 4. Example of Bounding Box object coordinates

Первое число является меткой класса, следующие 4 числа представляют собой нормированные координаты Bounding Box:

– X – первая координата центра Bounding Box;
– Y – вторая координата центра Bounding Box;
– W – ширина Bounding Box относительно центра;
– H – высота Bounding Box относительно центра.

Нормирование происходит путем деления координаты на размер изображения. Например, для X координаты центра выражение:

$$X\_center\_norm = \frac{X\_center}{image\_width} \quad (1)$$

Таким образом алгоритм получения координат Bounding Box для каждого изображения выглядит следующим образом:

1. Расположение камеры в случайной точке заданной области;
2. Определение углов поворота камеры для направления в центр области;
3. Поворот камеры;
4. Размещение объекта 3D Bounding Box;
5. Преобразование полученных координат на прошлом этапе в координаты системы камеры;
6. Запись нормированных координат в текстовый файл.

Для автоматизации описанных процессов, все этапы были заключены в собственные узлы (рис. 5).

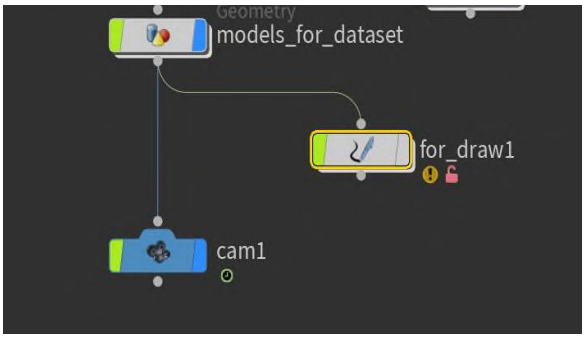

Рис. 5. Структура узлов для синтеза изображений
Fig. 5. Structure of nodes for image synthesis

Узел «models_for_dataset» отвечает за этапы создания области расположения камеры и рассчитывает углы поворота, так же в неё загружается объект исследования. Узел «for_draw» отвечает за получение координат ограничивающего Bounding Box.

Постобработка полученных изображений

Для постобработки полученных данных использовался сайт для разметки данных RoboFlow, на котором есть возможность осуществлять быструю аугментацию изображений.

Аугментация – это методика создания дополнительных обучающих данных из уже имеющихся. Для изображения данная методика представляет собой процесс применения разных трансформаций:

– наложение разных фильтров;
– размытие;
– вращение;
– вырез небольших квадратов на изображение.

Все эти методы помогают расширить датасет, а также избавиться от ложных признаков, сформированных в процессе обучения. Например, для распознавания автомобилей цвет автомобиля не важен, но, если в наборе данных для обучения доминирует черные машины, в процессе обучения у нейросети может сформироваться признак, который все черные объекты будет относить к автомобилям.

Постобработка полученных изображений

В качестве образца использовалась модель коробки отбора мощности от воздушно-реактивного двигателя Orenda (Avro Canada Orenda) [9], находящаяся в открытом доступе на ресурсе grabcad.com. Были отобраны CAD-модели наиболее сложной геометрической формы и переименованы эксперимента. Для данной работы была выбрана деталь «Гайка» под номером MAI1.2100.035.000 (рис.6).

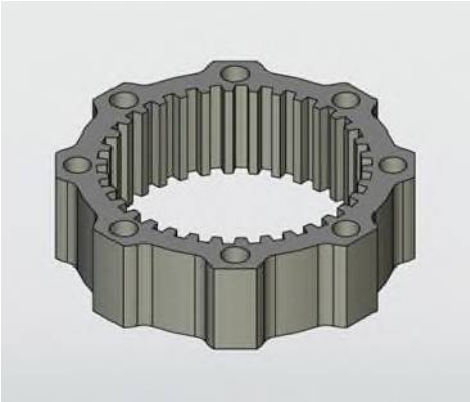

Рис. 6. Деталь MAI1.2100.035.000 («Гайка»)
Fig. 6. Detail MAI1.2100.035.000 («Nut»)

Для последующей проверки на реальных объектах деталь была напечатана с помощью аддитивных технологий и был подготовлен набор фотографий для процесса валидации (рис. 7). Фотографии распечатанной детали сделаны с помощью камеры мобильного телефона Xiaomi MI8. Параметры Контрастность, Насыщенность, Резкость установлены на «Норма», качество фотографии установлено на «Высокое», размер кадра 4:3.

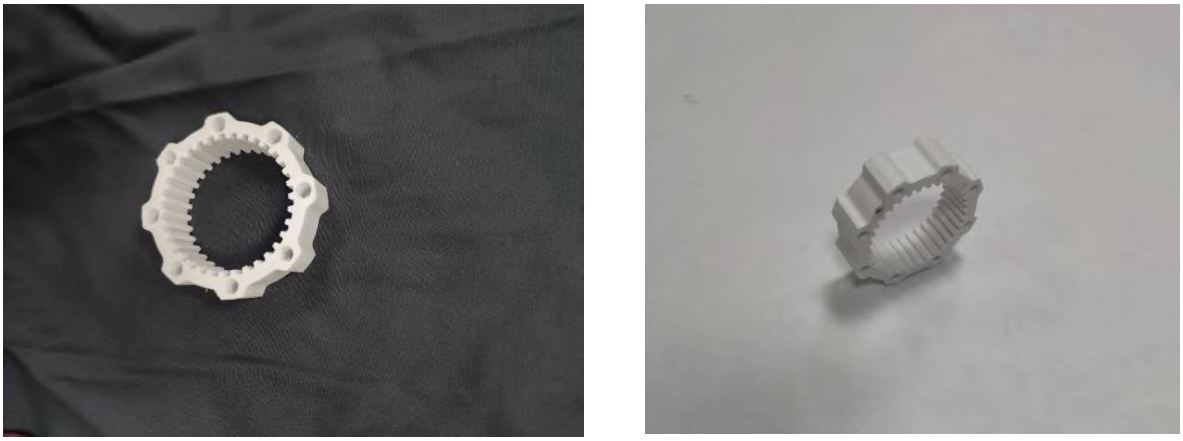

Рис. 7. Фотографии напечатанной детали на темном (слева) и светлом (справа) фоне
Fig. 7. Photos of the printed part on a dark (left) and light (right) background

Окружение

В качестве окружения использовался фон на основе произвольного изображения. Далее выбиралась точка центра модели объекта на сцене и относительно нее запускался Python-скрипт, описанный ранее. Такой подход позволяет сымитировать реальное окружение.

Для второго набора данных использовался только статичный фон из карты освещения (hdr).

Карта освещения использовалась в обоих случаях, т.к. она позволяет имитировать естественное освещение и тени, которые добавляют реалистичность изображению и отсутствие которых может оказывать негативный эффект на выходе работы нейросети (рис.8).

На фото слева рис.8 показан кадр, на котором модель объекта находится только в окружении карты освящения. На фото справа рис.8 можно увидеть объекты находящиеся на столе в окружении, похожем на комнату. Также на фото можно увидеть результаты применения аугментации.

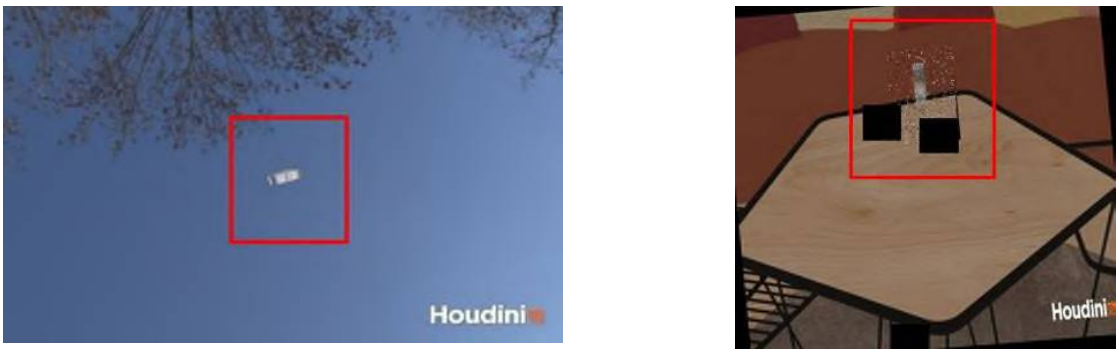

Рис. 8. Примеры использования карт освещения. Гайка обведена в красную рамку для наглядности. На применяемых изображениях красные рамки отсутствуют
Fig. 8. Examples of using lighting maps. The nut is circled in a red frame for clarity. There are no red borders on the applied images

Используемые модели нейросетей

В данной статье использовались следующие модели нейронного обучения:
– YOLO 5 версии (далее YOLO);
– RetinaNet (retinanet_R_50_FPN_3x) (далее RetinaNet);
– Faster R-CNN (faster_rcnn_R_50_FPN_3x) (далее f-CNN).

Модели YOLO представляют собой семейство моделей распознавания объектов, которые стали популярными благодаря высокому качеству предсказаний.

Процесс обработки изображения в YOLO отличается от работы других нейросетей. Поступающее на вход изображение делиться сеткой на области, которые напрямую предсказывают объект и ограничивают его рамкой. В результате получается большое количество ограниченных областей, среди которых формируется окончательный прогноз с помощью метода Кэнни (подавление немаксимумов, non-maximum suppression) (рис.9).

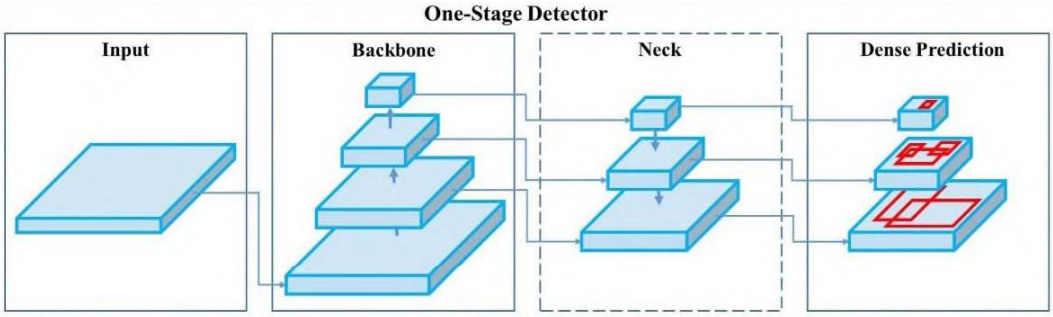

Рис. 9. Архитектура YOLO. Источник: Object detection algorithm – YOLOv5 Architecture [11]
Fig. 9. YOLO architecture. Source: Object detection algorithm – YOLOv5 Architecture [11]

Модель RetinaNet (рис.10) была разработана Facebook в 2018 году. Особенностью данной нейронной сети является предложение о потере фокуса и применение одноэтапной сети обнаружения цели. Такой подход позволил создать модель, сочетающую в себе скорость работы одноуровневых архитектур и увеличить точность, превосходя двухуровневые детекторы (как, например, R-CNN).

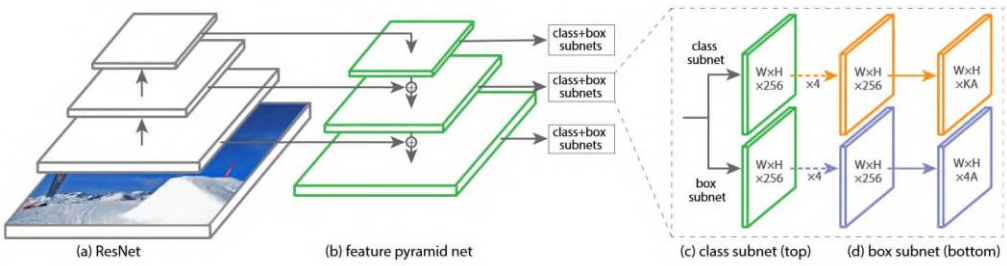

Рис. 10. Архитектура модели RetinaNet. Источник: [12]
Fig. 10. Architecture of the RetinaNet model. Source: [12]

Модель faster R-CNN (рис.11) представляет собой алгоритм обнаружения объектов без использования выборочного поиска, что позволяет сети исследовать предсказания по частям изображения с помощью отдельной сети. Предсказанные области затем изменяются с использованием слоя пула RoI, который затем используется для классификации изображения в пределах предлагаемой области и прогнозирования значений смещения для ограничивающих рамок.

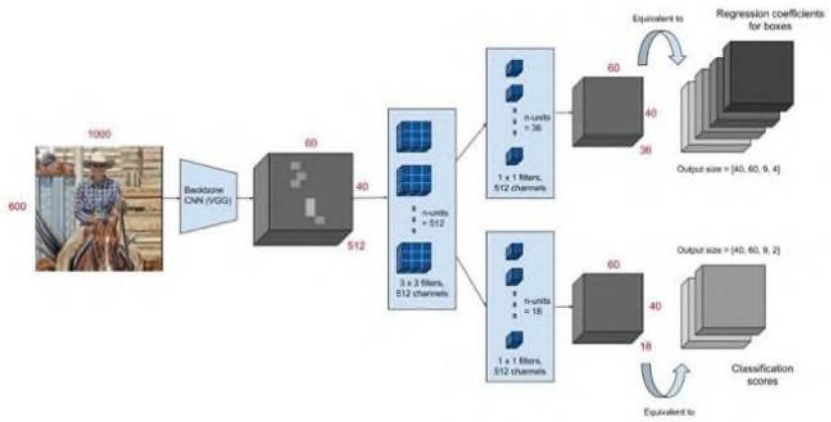

Рис. 11. Архитектура модели faster R-CNN. Источник: Faster R-CNN for object detection [13]
Fig. 11. Architecture of the faster R-CNN model. Source: Faster R-CNN for object detection [13]

Для проверки были выбраны предобученные веса с самыми большими значениями для каждой модели. Структура сетей не изменялась, не были применены методы заморозки или dropout.

Результаты применения моделей

Для обучения были синтезированы 100 фотографий. После применения аугментации их количество увеличилось до 300. Для обучения изображения были разбиты на обучающую и валидационную выборки в соотношении 80:20. Результат обучения тестировался на 55 фотографиях натурного объекта.

Результаты работы моделей на тестовой выборке при обучении на 300 итерациях представлены в таблицах ниже:

Таблица 1

Результаты для случая с пустым окружением

Table 1

Results for the case with an empty environment

Модель	Кол-во верных срабатываний	Кол-во ложных и множественных срабатываний
YOLO 5	29	1
RetinaNet	17	3
fasterCNN	14	2

Таблица 2

Результаты для случая с имитацией комнаты

Table 2

Results for the simulation room case

Модель	Кол-во верных срабатываний	Кол-во ложных и множественных срабатываний
YOLO 5	29	1
RetinaNet	21	3
fasterCNN	14	4

В рамках данной работы не стояла задача получения наилучшей модели распознавания, поэтому значения точности и стандартных метрик моделей не приводятся.

Столбец «Кол-во верных срабатываний» показывает на скольких изображениях удалось определить объект. Столбец «Кол-во ложных срабатываний» показывает количество изображении, на которых место объекта обозначено неверно (рис.12). Из табл.1 и табл.2 видно, что наилучший результат показала модель YOLO5, т.к. из 30 изображений удалось распознать практически все.

Рис. 12. Примеры распознавания гайки на фото. Слева показан правильное распознавание, справа - ошибка выделения объекта (засчитывается, как ложное распознавание)
Fig. 12. Examples of nut recognition in the photo. The correct recognition is shown on the left, the object selection error is shown on the right (counted as false recognition)

Также из таблиц можно увидеть отсутствие существенных изменений в результатах между двумя наборами данных при стандартных настройках нейросетей. Это говорит о том, что применение окружения не даёт ощутимых преимуществ, а процесс создания подобных кадров занимает в 3-4 раза больше времени.

Таким образом, в ходе проделанной работы удалось создать алгоритмы автоматизации процесса создания и разметки датасетов, а также видно, что полученные изображения могут быть использованы для обучения моделей нейросетей определению объектов.

Дальнейшие исследования

Как уже говорилось ранее, в ходе работы многие параметры были не затронуты, а их варьирование может оказать существенное влияние на конечный результат. Эти параметры станут объектом дальнейшего исследования. К ним можно отнести:

– реалистичность синтезируемого изображения;
– материалы 3*D*-объектов;
– параметры освещения;
– параметры камеры;
– и другие параметры трехмерной среды.

Не менее важным остается и постобработка полученных кадров виды и количество аугментаций, размеры Bounding Box. Больших исследований требует и количество генерируемых изображений. Также стоит рассмотреть возможности замены Bounding Box на сегментацию изображения как вариант разметки.

В ходе дополнительных исследований на стадии написания статьи было обнаружено, что добавление к обучающему набору ч/б изображений с контуром исследуемого объекта увеличивает количество верных срабатываний до 52 из 55 изображений.

Отдельным пунктом дальнейшего изучения стоит настройка самой нейросети, применение различных приемов контроля обучения. Использование других весов или обучение с «чистого листа» так же может значительно улучшить результата. Стоит отметить, что увеличение количества классов для распознавания так же может сказаться положительно.

Также необходимо довести автоматизацию до конца и отказаться от использования интернет-ресурсов (Roboflow) и создать систему, которая сможет по полученным 3*D*-моделям осуществлять дальнейшие шаги для обучения либо уже готовых моделей, либо создание новых.

СПИСОК ИСТОЧНИКОВ

1. Richard J. Chen, Ming Y. Lu, Tiffany Y. Chen, Drew F. K. Williamson, Faisal Mahmood, «Synthetic data in machine learning for medicine and healthcare» Nature Biomedical Engineering (5), 2021. p. 493-497.
2. В. Кондаратцев, А. Крючков, Р. Чумак, «Машинное зрение в промышленной дефектоскопии» PHYGI-TALISM. Москва. 2020.
3. Aleksei Boikov, Vladimir Payor, Roman Savelev, Alexandr Kolesnikov, «Synthetic Data Generation for Steel Defect Detection and Classification Using Deep Learning» Symmetry. 2021. № 13.

References:

1. Chen R.J., Lu M.Y., Chen T.Y., Y.Ch., Williamson D.F.K., Mahmood F. Synthetic Data in Machine Lear-ning for Medicine and Healthcare. Nature Biomedical Engineering. 2021;(5):493-497.
2. Kondaratsev V., Kryuchkov A., Chumak R. Machine Vision in Industrial Flaw Detection. Moscow: Phygitalism; 2020.
3. Boikov A., Payor V., Savelev R., Kolesnikov A. Synthetic Data Generation for Steel Defect Detection and Classification Using Deep Learning. Symmetry; 2021:13.

4. Erfanian Ebadi, Salehe; Jhang, You-Cyuan; Zook, Alex; Dhakad, Saurav; Crespi, Adam; Parisi, Pete; Borkman, Steven; Hogins, Jonathan; Ganguly, Sujoy, «PeopleSansPeople: A Synthetic Data Generator for Human-Centric Computer Vision» Unity technologies, 2021.
5. Matteo Fabbri, Guillem Brasó, «MOTSynth: How Can Synthetic Data Help Pedestrian Detection and Tracking?» Modena. 2021.
6. Damien Trentesaux, Theodor Borangiu, Paulo Leitão, Jose-Fernando Jimenez, Jairo R. Montoya-Torres, « SOHOMA: International Workshop on Service Orientation in Holonic and Multi-Agent Manufacturing,» в Service Oriented, Holonic and Multi-Agent Manufacturing Systems for Industry of the Future. Bogota. Colombia. 2021.

7. Pablo Martinez-Gonzalez, Sergiu Oprea, John Alejandro Castro-Vargas, Alberto Garcia-Garcia, Sergio Orts-Escolano, Jose Garcia-Rodriguez, Markus Vincze, «International Joint Conference on Neural Networks (IJCNN),» в UnrealROX+: An Improved Tool for Acquiring Synthetic Data from Virtual 3D Environments. Shenzhen. China. 2021.
8. «Houdini help» [В Интернете]. Available: https:// www.sidefx.com/docs/houdini/index.html.
9. D. Rutland, «Orenda 9/10 jet engine power take off gearbox». 2021. [В Интернете]. Available: grabcad.com.

10. S. Gutta, «Object Detection Algorithm – YOLO v5 Architecture. Object Detection Algorithm – YOLO v5 Architecture» [В Интернете]. Available: https://medium. com/ analytics-vidhya/object-detection-algorithm-yolo-v5-architecture-89e0a35472ef.
11. A.A.f. Python, «How RetinaNet works» [В Интер-нете]. Available: https://developers.arcgis. com/python/ guide/how-retinanet-works/.
12. S. Ananth, «Faster R-CNN for object detection» [В Интернете]. Available: https://towardsdatascience.com / faster – r – cnn – for – object – detection – a - technical-summary-474c5b857b46.

Информация об авторах:

Денис Александрович Копылов
студент-магистр Института №3 «Системы управления, информатика и электроэнергетика» Московского Авиационного института (Национального исследовательского университета), ORCID: 0000-0002-0138- 3950.

Егор Сергеевич Агешин
аспирант Института №2 «Авиационные, ракетные двигатели и энергетические установки» Московского Авиационного института (Национального исследовательского университета), ORCID: 0000-0002-3939-5690.

Ольга Владиславовна Хомутская
кандидат технических наук, доцент кафедры 307 Института №3 «Системы управления, информатика и электроэнергетика» Московского Авиационного института (Национального исследовательского университета), ORCID: 0000-0003-1814-6334.

4. Ebadi S.E., Jhang Y.C., Zook A., Dhakad S., Crespi A., Parisi P., Borkman S., Hogins J., Sujoy G. PeopleSansPeople: A Synthetic Data Generator for Human-Centric Computer Vision, Unity Technologies [Internet]. 2021. Available from: https://github.com/ Unity-Tech-nologies.
5. Fabbri M., Brasó G. MOTSynth: How Can Synthetic Data Help Pedestrian Detection and Tracking? Modena (Italy): University of Modena and Reggio Emilia; 2021.
6. Trentesaux D, Borangiu Th, Leitão P, Jimenez J-F, Montoya-Torres JR. Service Oriented, Holonic and Multi-Agent Manufacturing Control. In: Proceedings of the 1st Latin-American International Workshop on Service-Oriented, Holonic and Multi-Agent Manufacturing Systems for Industry of the Future. SOHOMA (Latin America): Bogota, Colombia: 2021.
7. Martinez-Gonzalez P, Oprea S, Castro-Vargas JA, Garcia-Garcia A, Orts-Escolano S, Garcia-Rodriguez J, Vincze M. UnrealROX+: An Improved Tool for Acquiring Synthetic Data from Virtual 3D. In: Proceedings of Environments International Joint Conference on Neural Networks (IJCNN); Shenzhen (China): 2021.
8. Houdini Help [Internet]. Available from: https:// www.sidefx.com/docs/houdini/index.html
9. Rutland D. Orenda 9/10 Jet Engine Power Take off Gearbox [Internet]. 2021. Available from: https:// grabcad.com / library/orenda-9-10-jet-engine-po-wer- take-off-gearbox-1.
10. Gutta S. Object Detection Algorithm – YOLO v5 Architecture. History and architecture of YOLOv5. [Internet]. Available from: https://medium.com/analytics-vidhya/ object-detection-algorithm-yolo-v5-architecture-89e0a35472ef.

11. API for Python. How RetinaNet works [Internet]. Available from: https://developers.arcgis.com/python /guide/ how-retinanet-works/.
12. Ananth S. Faster R-CNN for Object Detection [Internet]. Available from: https://towardsdatascience. com / faster-r-cnn-for-object-detection-a-technical-sum-mary-474c5b857b46.

Information about authors:

Denis Alexandrovich Kopylov
master's student of Institute No. 3 «Control Systems, Computer Science and Power Engineering» of Moscow Aviation Institute (National Research University), ORCID iD: 0000-0002-0138-3950.

Yegor Sergeevich Ageshin
post-graduate student of Institute No. 2 «Aircraft, rocket engines and power plants» of Moscow Aviation Institute (National Research University), ORCID iD: 0000-0002-3939-5690.

Olga Vladislavovna Khomutskaya
Candidate of Technical Sciences, Associate Professor of Department 307 of Institute No. 3 «Control Systems, Computer Science and Power Engineering» of Moscow Aviation Institute (National Research University), ORCID iD: 0000-0003-1814-6334.

Вклад авторов: все авторы сделали эквивалентный вклад в подготовку публикации.
Contribution of the authors: the authors contributed equally to this article.

Авторы заявляют об отсутствии конфликта интересов.
The authors declare no conflicts of interests.

Статья поступила в редакцию 21.06.2022; одобрена после рецензирования 02.08.2022; принята к публикации 05.08.2022.
The article was submitted 21.06.2022; approved after reviewing 02.08.2022; accepted for publication 05.08.2022.

Рецензент – Подвесовский А.Г., кандидат технических наук, доцент, Брянский государственный технический университет.
Reviewer – Podvesovskij A.G., Candidate of Technical Sciences, Associate Professor, Bryansk State Technical University.