**МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ**

**БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ**

**ФАКУЛЬТЕТ ПРИКЛАДНОЙ МАТЕМАТИКИ И ИНФОРМАТИКИ**

КАФЕДРА КОМПЬЮТЕРНЫХ ТЕХНОЛОГИЙ И СИСТЕМ

**РАЗРАБОТКА ИНСТРУМЕНТА ПОДГОТОВКИ ИЗОБРАЖЕНИЙ ДЛЯ ОБУЧЕНИЯ ГЛУБОКИХ СЕТЕЙ**

Курсовая работа

Студента 3 курса

Никончика Даниила

Викторовича

Научный

руководитель:

доктор технических наук

А.М. Недзьведь

Минск, 2022

Содержание

[ВВЕДЕНИЕ 3](#_Toc103513934)

[Глава 1. Интерактивные инструменты подготовки изображений 4](#_Toc103513935)

[1.1. Структура и функционал интерактивных инструментов 4](#_Toc103513936)

[1.2. Элементы автоматизации выделения объектов 6](#_Toc103513937)

[1.2.1. Выделение объектов для поиска 7](#_Toc103513938)

[1.2.2. Выделение объектов для сегментации 8](#_Toc103513939)

[1.3. Методы полуавтоматической сегментации 9](#_Toc103513940)

[1.4. Требования к исходным данным методов машинного обучения 11](#_Toc103513941)

[Глава 2. Алгоритмы автоматизации для подготовки ROI 13](#_Toc103513942)

[2.1. Общая схема подготовки данных 13](#_Toc103513943)

[2.2. Выделение объектов на основе алгоритма водораздела 13](#_Toc103513944)

[2.3. Построение активных контуров для выделения объектов на изображении 14](#_Toc103513945)

[2.4. Змеиное, многомасштабное редактирование кривых 16](#_Toc103513946)

[Глава 3. Программная реализация среды интерактивных инструментов 25](#_Toc103513947)

[3.1. Определения средств разработки 25](#_Toc103513948)

[3.2. Проблема с базовым пороговым значением при извлечении контуров 25](#_Toc103513949)

[3.3. Особенности реализации полуавтоматических алгоритмов 25](#_Toc103513950)

[ЗАКЛЮЧЕНИЕ 27](#_Toc103513951)

[Список использованной литературы: 28](#_Toc103513952)

# ВВЕДЕНИЕ

Аннотации к изображениям играют важную роль в компьютерном зрении, технологии, которая позволяет компьютерам получать высокоуровневое представление о цифровых изображениях или видео, а также видеть и интерпретировать визуальную информацию так же, как люди.

Технология компьютерного зрения предоставляет несколько удивительных приложений искусственного интеллекта, таких как автономные автомобили, обнаружение опухолей и беспилотные летательные аппараты. Однако большинство из этих замечательных применений компьютерного зрения было бы возможно без аннотации изображения.

Аннотирование изображений является основным шагом в создании большинства моделей компьютерного зрения. Необходимо, чтобы наборы данных были полезными компонентами машинного обучения и глубокого обучения для компьютерного зрения.

# Глава 1. Интерактивные инструменты подготовки изображений

# Структура и функционал интерактивных инструментов

В приложении, основанном на представлении, пользователь, как правило, “На рельсах”, поскольку приложение позволяет выполнять X после Y, но не раньше. Когда я запускаю приложение Twitter, я не могу просто сразу перейти ко всему - мне приходится просматривать последовательности просмотров. Это позволяет разработчикам приложения делать обширные предположения о состоянии приложения.

В частности, хотя представления могут манипулировать одной и той же базовой моделью данных (почти всегда какой-либо формой базы данных), мне никогда не нужно беспокоиться о том, чтобы отличить нажатие в одном представлении от нажатия в другом. В некотором смысле представления являются режимами, и в контексте конкретного представления, как правило, существуют только команды, а не инструменты.

В результате в приложении, основанном на представлении, очень легко говорить о рабочих процессах. Люди, создающие приложения на основе представлений, как правило, рисуют множество диаграмм, которые выглядят следующим образом:

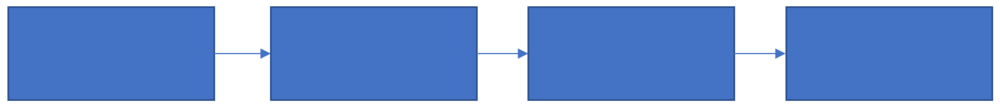


Рисунок 1 – Диаграммы приложения на основе представлений

Эти диаграммы могут быть самими представлениями, но чаще всего это шаги, которые пользователь будет выполнять в приложении - если хотите, история пользователя. Они не всегда строго линейны, могут быть ветви и циклы. Но всегда есть четко определенные точки входа и выхода. Пользователь начинает с задачи и завершает ее выполнением в рамках рабочего процесса. Тогда очень естественно разработать приложение, которое обеспечивает рабочий процесс, в котором пользователь может выполнить задачу. Мы можем осмысленно говорить о прогрессе в рабочем процессе, и связанные с этим данные и состояние приложения также являются своего рода Прогрессом. По мере добавления дополнительных задач задача команды разработчиков состоит в том, чтобы разработать дизайн, позволяющий эффективно выполнять эти необходимые рабочие процессы.

Фундаментальная сложность в приложениях для создания/редактирования контента заключается в том, что эта методология вообще к ним неприменима. В конечном счете, я думаю, разница заключается в том, что в инструменте создания/редактирования контента нет неотъемлемого понятия прогресса. Например, как пользователь Powerpoint, я могу часами перестраивать свои слайды, изменять размер и выравнивание изображения, слегка корректировать текст. На мой взгляд, у меня может быть какое-то туманное представление о прогрессе, но это не закодировано в Приложении. Моя задача находится вне приложения. А без четкой задачи или показателя прогресса нет рабочего процесса!

Я думаю, что более полезная мысленная модель для приложений для создания/редактирования контента похожа на изображение справа. Зеленый центральный концентратор-состояние по умолчанию в этих приложениях, где, как правило, вы просто просматриваете свой контент. Например, панорамирование и масштабирование изображения в Photoshop или навигация по 3D-сцене в Blender. Именно здесь пользователь проводит значительную часть своего времени. Синие спицы-это инструменты. Я ненадолго захожу в инструмент, но всегда возвращаюсь в центр.

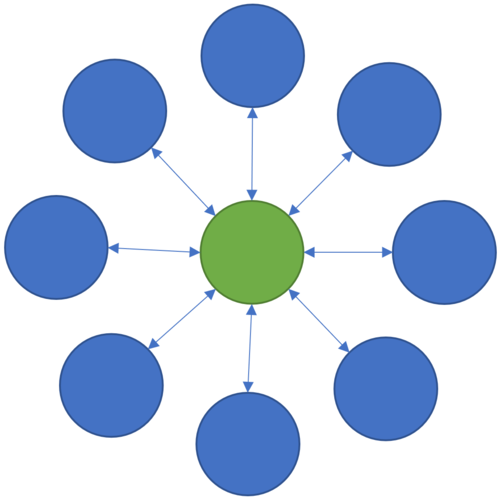


Рисунок 2 – Модель для приложений

Поэтому, если бы я отслеживал свое состояние с течением времени, это был бы извилистый путь в центр по умолчанию и обратно с помощью бесчисленного количества инструментов. Там нет четко определенного порядка, как пользователь, я, как правило, могу свободно использовать Инструменты в любом порядке, который я считаю нужным. В микрокосме мы могли бы найти небольшие четко определенные рабочие процессы для анализа и оптимизации, но на уровне приложений рабочие процессы фактически бесконечны.

# Элементы автоматизации выделения объектов

Геометрия каждой страницы в формате PDF может быть описана одной из двух систем координат: пользовательским пространством по умолчанию или повернутым пользовательским пространством. Разница между этими двумя координатными пространствами обусловлена только поворотом, размером страницы и обрезкой, но не масштабированием, что действительно усложнило бы ситуацию. Единицами измерения как для повернутых, так и для пробелов по умолчанию являются “точки", где 72 точки равны одному дюйму. Например, лист бумаги стандартного формата размером 8,5 х 11 дюймов или 612 х 792 точек (рис. 3).

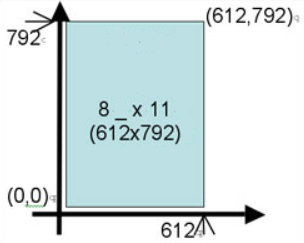


Рисунок 3 – Размеры листа бумаги

Пространство пользователя по умолчанию представляет собой развернутую, не обрезанную страницу. Начало координат (или нулевая точка) находится в левом нижнем углу поля носителя. Предполагается, что поле для печати должно отражать фактический размер бумаги, на которой будет напечатана страница. Хотя это не всегда строго верно, это хороший способ подумать об этом, т.е. пользовательское пространство по умолчанию-это система координат бумаги, на которой будет напечатана страница.

Однако все мы знаем, что принтер не может печатать на краю бумаги. Существует разница между фактическим печатным материалом и краем бумаги. Граница этого поля называется Полем обрезки. Это то, что пользователь видит на экране, когда страница отображается в Acrobat. На дисплее Acrobat пространство внутри поля обрезки отображается на белом фоне, и все, что находится за пределами поля обрезки, невидимо.

Фактическая печатная часть страницы (поле обрезки) может быть ориентирована как книжная, альбомная или даже перевернутая. Таким образом, представление страницы пользователем может быть изменено с бумаги, на которой она напечатана. Повернутое пользовательское пространство-это представление страницы пользователем, т.е. обрезанный повернутый вид страницы. Источником повернутого пользовательского пространства является нижний левый угол поля обрезки. На рис. 4 показано, как это работает для страницы, напечатанной на листе бумаги стандартного формата с изображением буквы в альбомной ориентации.

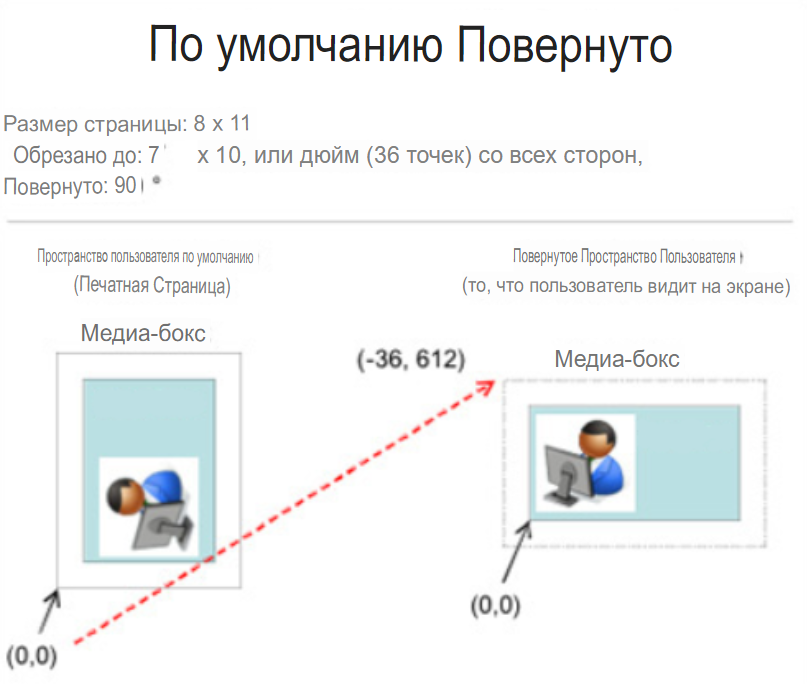


Рисунок 4 - Преобразование пространства пользователя по умолчанию в повернутое.

# Выделение объектов для поиска

Алгоритм распознавания объектов определяет, какие объекты присутствуют на изображении. Он принимает все изображение в качестве входных и выходных меток классов и вероятностей классов объектов, присутствующих на этом изображении. Например, метка класса может быть “собака”, и связанная с ней вероятность класса может составлять 97%.

С другой стороны, алгоритм обнаружения объектов не только сообщает вам, какие объекты присутствуют на изображении, но также выводит ограничивающие рамки (x, y, ширина, высота) для указания местоположения объектов внутри изображения.

В основе всех алгоритмов обнаружения объектов лежит алгоритм распознавания объектов. Предположим, мы обучили модель распознавания объектов, которая идентифицирует собак в пятнах изображений. Эта модель покажет, есть ли на изображении собака или нет. Он не сообщает, где находится объект.

Чтобы локализовать объект, мы должны выбрать области (участки) изображения, а затем применить алгоритм распознавания объектов к этим участкам изображения. Расположение объектов определяется расположением участков изображения, где вероятность класса, возвращаемая алгоритмом распознавания объектов, высока.

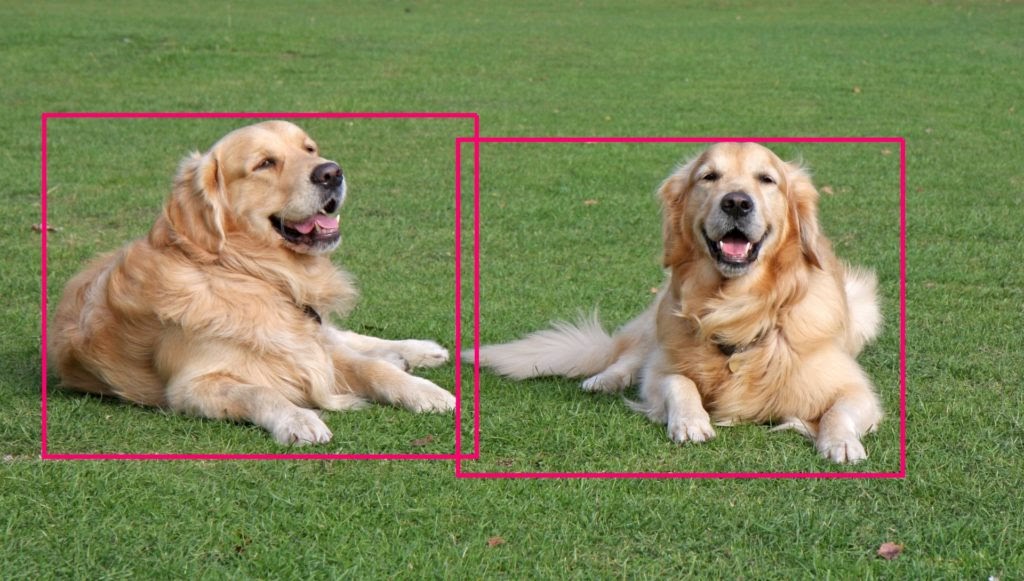


Рисунок 5 - Пример распознания объектов на изображении

Наиболее простой способ создания небольших субрегионов (патчей) называется подходом со скользящим окном. Однако подход со скользящим окном имеет несколько ограничений. Эти ограничения преодолеваются классом алгоритмов, называемых алгоритмами “Предложения региона”. Выборочный поиск-один из самых популярных алгоритмов предложений по регионам.

# Выделение объектов для сегментации

Мы можем разделить или разбить изображение на различные части, называемые сегментами. Не очень хорошая идея обрабатывать все изображение одновременно, так как на изображении будут области, которые не содержат никакой информации.

Разделив изображение на сегменты, мы можем использовать важные сегменты для обработки изображения. Вот, в двух словах, как работает сегментация изображений.

Изображение - это набор или набор различных пикселей. Мы группируем пиксели, имеющие схожие атрибуты, с помощью сегментации изображений.

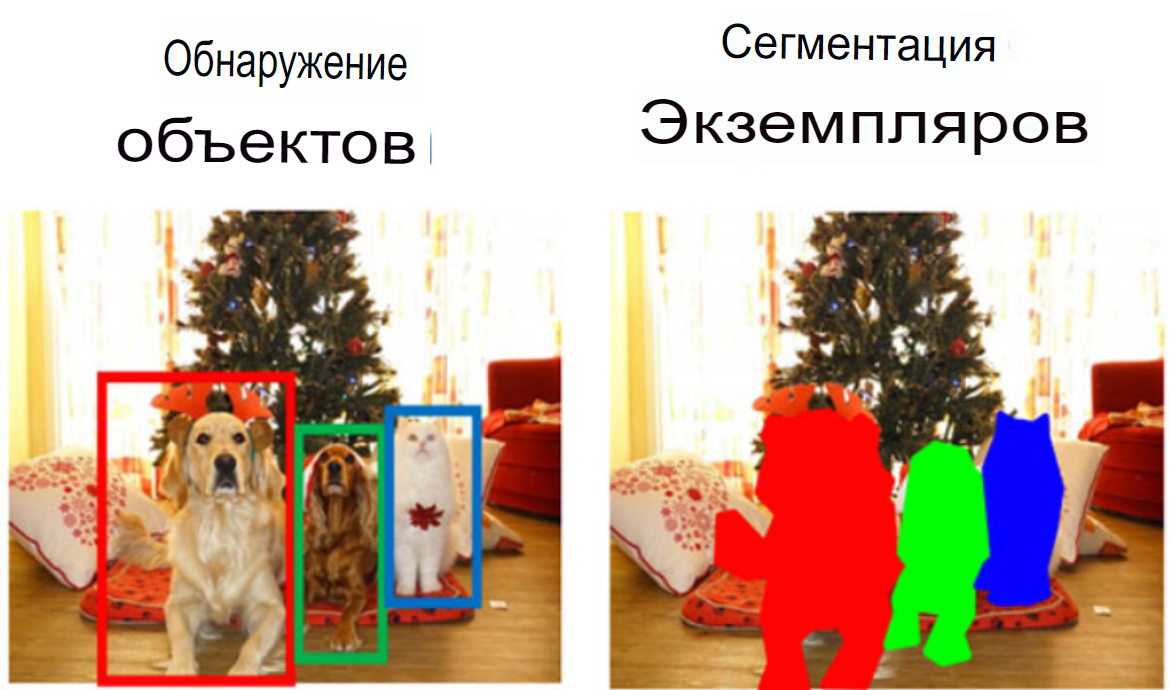


Рисунок 6 – Пример сегментации изображения

Обнаружение объектов создает ограничивающую рамку, соответствующую каждому классу на изображении. Но это ничего не говорит нам о форме объекта. Мы получаем только набор координат граничной рамки. Мы хотим получить больше информации – это слишком расплывчато для наших целей.

Сегментация изображения создает пиксельную маску для каждого объекта на изображении. Этот метод дает нам гораздо более детальное представление об объекте(объектах) на изображении.

# Методы полуавтоматической сегментации

В этом исследовании я использовал пять типов полуавтоматических методов сегментации изображений, и результаты, полученные этими методами, были сопоставлены с результатами, полученными вручную на предыдущем этапе. Эти методы обобщены следующим образом.

1) Выращивание посевных площадей (SRG). SRG, репрезентативный метод сегментации изображений, широко используется в исследованиях, связанных с медицинскими изображениями. Этот метод постепенно группирует пиксели окрестности с одинаковой интенсивностью из заданной пользователем начальной точки и объединенных областей. Этот процесс выполняется итеративно до тех пор, пока все пиксели не будут включены в каждую область в соответствии с правилами объединения.

2) Модель активного контура на основе набора уровней. Традиционные модели активных контуров можно разделить на параметрические модели и геометрические модели в соответствии с типами представления или реализации. Геометрическая модель, введенная Caselles и др., основан на теории эволюции кривых и методе набора уровней. Однако эти традиционные методы имеют несколько ограничений. Энергия не является внутренней, потому что она сильно зависит от параметризации кривой и не связана с геометрией объектов.

3) Модель активного контура на основе локализованной области. В 2008 году Ланктон и Танненбаум [23] предложили модель активного контура на основе локализованной области. Этот метод использует параметры области, с помощью которых передний план и фон изображения описываются в терминах небольших локальных областей. Для оптимизации локальной энергии каждая точка на контуре рассматривается независимо и перемещается, чтобы минимизировать энергию, вычисленную в ее собственной локальной области. Затем эти локальные энергии вычисляются путем разделения локальных окрестностей на локальные внутренние и внешние области путем построения кривой. Энергия определяется как:

Е(φ)=∫Ωxδϕ(х)∫ΩyB(х,g)·ф(Z(G),ϕ(г))dxdу+λ∫Ωxδϕ(х)‖∇ϕ(х)‖DХ. (1)

Параметр λ представляет собой весовые плавные члены, которые используются для сохранения плавности кривой. Здесь B(x,y) с радиусом γ используется для маскировки локальных областей (т.е. локальных внутренних и внешних). Требуется компромисс между скоростью сходимости и размером локального радиуса. Размеры радиуса, которые слишком велики или слишком малы, могут привести к неправильной сегментации.

4) Сегментация на основе кластеризации. В качестве двух других подходов к сегментации GGN используются методы, основанные на кластеризации, кластеризация K-средних и нечетких C-средних.

Алгоритм кластеризации K-средних связывает заданные группы данных с определенным пользователем числом кластеров K и минимизирует различия в различиях с каждым кластером. Этот алгоритм также широко используется для сегментации изображений.

Алгоритм кластеризации нечетких C-средних делит множество Χ = {χ1, χ2, χ3,... χn}, состоящее из конечного числа элементов, на c нечетких кластеров в соответствии с правилами. Учитывая конечный набор данных, алгоритм возвращает список центров V кластеров cи матрицу разбиения U, как в формулах (2) и (3):

V = v\_i, i = 1, 2, ..., c (2)

U = u\_ij, i = 1, 2, ..., c, j = 1, 2, ..., n (3)

где u\_ij-числовое значение с диапазоном значений [0, 1], представляющее степень принадлежности определенного элемента xj к i-му кластеру.

5) Оценка точности. Чтобы количественно оценить точность сегментации GGN, мы рассчитали коэффициенты Dice среди регионов, полученных вручную и с помощью пяти типов полуавтоматических методов. Метрика измеряет сходство двух регионов и колеблется от 0 для несвязанных регионов до 1 для идентичных регионов. Коэффициент игры в кости определяется как

D = 2×|X∩Y|/(|X|+|Y|), (4)

где X и Y представляют две сегментированные области.

# Требования к исходным данным методов машинного обучения

Первое, что вам следует определить, - это какой ресурс требуется для выполнения вашей задачи. Давайте посмотрим, как разные задачи будут иметь разные требования к оборудованию:

* Если ваши задачи невелики и могут поместиться в сложную последовательную обработку, вам не нужна большая система. Вы даже можете полностью пропустить графические процессоры. Такой процессор, как i7–7500U, может обучать в среднем ~115 примеров в секунду. Поэтому, если вы планируете работать с другими областями ML или алгоритмами, графический процессор не нужен.
* Если ваша задача немного трудоемка и содержит управляемые данные, вам подойдет достаточно мощный графический процессор. Ноутбук с выделенной видеокартой высокого класса должен выполнить эту работу. Есть несколько высококлассных (и ожидаемо тяжелых) ноутбуков, таких как Nvidia GTX 1080 (8 ГБ видеопамяти), которые могут обучать в среднем ~14 тыс. примеров в секунду. Кроме того, вы можете создать свой собственный компьютер с разумным процессором и мощным графическим процессором, но имейте в виду, что процессор не должен ограничивать работу графического процессора. Например, i7-7500U будет безупречно работать с графическим процессором GTX 1080.
* Если вы работаете над сложными проблемами или являетесь компанией, которая использует глубокое обучение, вам, вероятно, следует создать собственную систему глубокого обучения или использовать облачный сервис.
* Если ваша задача масштабнее, чем обычно, и у вас достаточно денег, чтобы покрыть расходы, вы можете выбрать кластер с графическим процессором и выполнять вычисления с несколькими графическими процессорами. Кроме того, доступны более мощные опции – ТПУ и более быстрые ПЛИС, – которые разработаны специально для этих целей.

Кроме того, графический процессор может выполнять операции на основе сверточных/CNN или рекуррентных нейронных сетей/RNN. Он также может выполнять операции с пакетом изображений из 128 или 256 изображений одновременно всего за несколько миллисекунд. Однако потребляемая мощность составляет около ~250 Вт и требует полноценного ПК, для которого дополнительно требуется 150 Вт мощности, что приводит к общей мощности 400 Вт.

Такие приложения, как очки виртуальной или дополненной реальности, беспилотные летательные аппараты, мобильные устройства и небольшие роботы, не обладают такой мощью. Кроме того, в случае автономных автомобилей и интеллектуальных камер, где необходимо видео в реальном времени, пакетная обработка изображений невозможна, так как видео должно обрабатываться в режиме реального времени для своевременного реагирования.

# Глава 2. Алгоритмы автоматизации для подготовки ROI

# Общая схема подготовки данных

Сегментация изображений изначально начиналась с цифровой обработки изображений в сочетании с алгоритмами оптимизации. Эти примитивные алгоритмы использовали такие методы, как алгоритм роста регионов и алгоритм змеи, где они создавали начальные области, а алгоритм сравнивал значения пикселей, чтобы получить представление о карте сегментов.

Эти методы учитывали локальное представление объектов на изображении и фокусировались на локальных различиях и градиентах в пикселях.

Алгоритмы, которые приняли глобальный взгляд на входное изображение, появились намного позже, когда среди классических методов обработки изображений были предложены такие методы, как адаптивное пороговое значение, алгоритм Otsu и алгоритмы кластеризации.

# Выделение объектов на основе алгоритма водораздела

Алгоритм водораздела является классическим алгоритмом, используемым для сегментации, и особенно полезен при извлечении соприкасающихся или перекрывающихся объектов на изображениях, таких как монеты на рисунке ниже.



Рисунок 7 - Перекрывающиеся объекты

Используя традиционные методы обработки изображений, такие как пороговое значение и определение контуров, мы не смогли бы извлечь каждую отдельную монету из изображения, но, используя алгоритм водораздела, мы можем обнаружить и извлечь каждую монету без проблем.

При использовании алгоритма водораздела мы должны начинать с пользовательских маркеров. Эти маркеры могут быть либо определены вручную с помощью точки и щелчка, либо мы можем автоматически или эвристически определить их с помощью таких методов, как пороговые значения и/или морфологические операции.

Так же при сегментации водоразделов изображение рассматривается как топографический ландшафт с хребтами и долинами. Значения высоты ландшафта обычно определяются значениями серого для соответствующих пикселей или величиной их градиента. На основе такого трехмерного представления преобразование водораздела разбивает изображение на водосборные бассейны. Для каждого локального минимума водосборный бассейн включает все точки, путь наискорейшего спуска которых заканчивается в этом минимуме.

Основываясь на этих маркерах, алгоритм водораздела обрабатывает пиксели на нашем входном изображении как локальную высоту (называемую топографией) — метод “заливает” долины, начиная с маркеров и двигаясь наружу, пока долины разных маркеров не встретятся друг с другом. Чтобы получить точную сегментацию водораздела, маркеры должны быть правильно размещены.

# Построение активных контуров для выделения объектов на изображении

Модель активного контура-это метод подгонки открытых или закрытых сплайнов к линиям или краям на рисунке 8. Он работает за счет минимизации энергии, которая частично определяется изображением, а частично формой сплайна: длиной и плавностью. Минимизация выполняется неявно в энергии формы и явно в энергии изображения.

Так же модель активного контура представляет собой минимизирующую энергию сплайновую кривую, состоящую из дискретных контрольных точек, называемых снакселями. Кривая притягивается к элементам как краям целевого объекта посредством оценки внутренних и внешних сил. Классическая реализация ACM склонна к попаданию в ловушку проблемы локальных минимумов, а также очень чувствительна к инициализации контрольных точек, поскольку они требуют близости к целевому объекту; в противном случае произойдет нарушение сходимости.

В следующих двух примерах модель активного контура используется для сегментации лица человека от остальной части изображения путем подгонки замкнутой кривой к краям лица и для нахождения самой темной кривой между двумя фиксированными точками при соблюдении соображений гладкости. Обычно рекомендуется немного сгладить изображения перед анализом, как это делается в следующих примерах.

Мы инициализируем круг вокруг лица астронавта и используем граничное условие по умолчанию boundary\_condition='periodic', чтобы соответствовать замкнутой кривой. Параметры по умолчанию w\_line=0, w\_edge=1 заставят кривую искать по краям, таким как границы грани.

**import** **numpy** **as** **np**

**import** **matplotlib.pyplot** **as** **plt**

**from** **skimage.color** **import** rgb2gray

**from** **skimage** **import** data

**from** **skimage.filters** **import** gaussian

**from** **skimage.segmentation** **import** active\_contour

img = data.astronaut()

img = rgb2gray(img)

s = np.linspace(**0**, **2**\*np.pi, **400**)

r = **100** + **100**\*np.sin(s)

c = **220** + **100**\*np.cos(s)

init = np.array([r, c]).T

snake = active\_contour(gaussian(img, **3**, preserve\_range=False),

init, alpha=**0.015**, beta=**10**, gamma=**0.001**)

fig, ax = plt.subplots(figsize=(**7**, **7**))

ax.imshow(img, cmap=plt.cm.gray)

ax.plot(init[:, **1**], init[:, **0**], '--r', lw=**3**)

ax.plot(snake[:, **1**], snake[:, **0**], '-b', lw=**3**)

ax.set\_xticks([]), ax.set\_yticks([])

ax.axis([**0**, img.shape[**1**], img.shape[**0**], **0**])

plt.show()

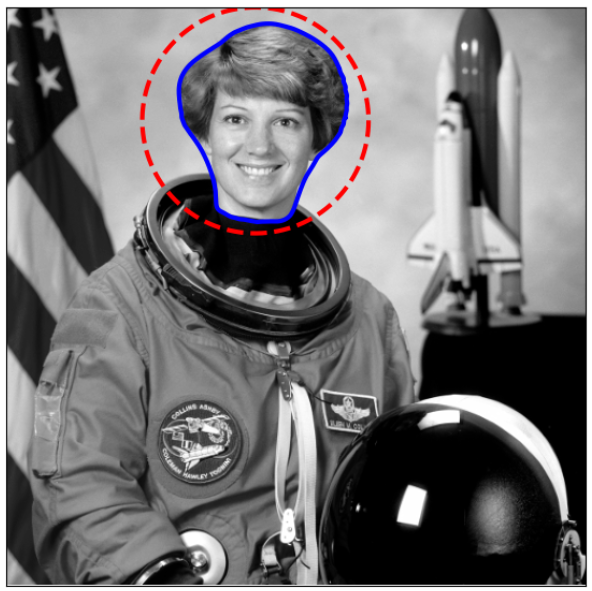


Рисунок 8 - Круг вокруг лица астронавта

Здесь мы инициализируем прямую линию между двумя точками, (5, 136) и (424, 50), и требуют, чтобы сплайн имел там свои конечные точки, задав граничное условие граничное условие=’исправлено’. Кроме того, мы заставляем алгоритм искать темные линии, давая отрицательный результат значение w\_line.

img = data.text()

r = np.linspace(**136**, **50**, **100**)

c = np.linspace(**5**, **424**, **100**)

init = np.array([r, c]).T

snake = active\_contour(gaussian(img, **1**, preserve\_range=False),

init, boundary\_condition='fixed',

alpha=**0.1**, beta=**1.0**, w\_line=-**5**, w\_edge=**0**, gamma=**0.1**)

fig, ax = plt.subplots(figsize=(**9**, **5**))

ax.imshow(img, cmap=plt.cm.gray)

ax.plot(init[:, **1**], init[:, **0**], '--r', lw=**3**)

ax.plot(snake[:, **1**], snake[:, **0**], '-b', lw=**3**)

ax.set\_xticks([]), ax.set\_yticks([])

ax.axis([**0**, img.shape[**1**], img.shape[**0**], **0**])

plt.show()

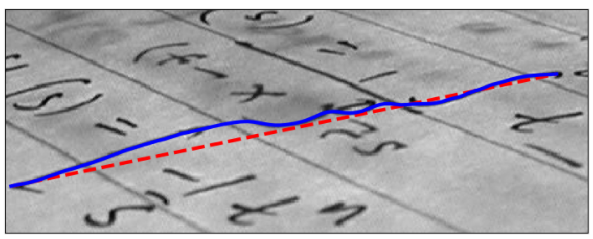


Рисунок 9 - Пример прямой

# Змеиное, многомасштабное редактирование кривых

Змеиное редактирование. Активный контур или змеи широко используются для определения границ или определения краев на медицинских изображениях. Змея определяется как сплайн минимизации энергии, энергия которого зависит от его формы и расположения на изображении. Форма змеи управляется внутренними силами и внешними силами. Внешняя сила направляет змею к элементам изображения, а внутренняя сила действует как ограничение сглаживания для змеи. Пусть вектор *p(s) = (x(s), y(s))* является параметрическим представлением змеи, где значение *s* от 0 до 1. Энергетическая функция, которую мы хотим минимизировать, определена и представлена следующим образом в уравнении(5):

(5)

это внутренние силы, которые заставляют змею быть маленькой и гладкой. это внешняя энергия для змеи, которая находит края объекта на изображении. Общая внешняя энергия является обратной величиной градиента, другими словами, низкие энергии в месте расположения краев, более высокие энергии повсюду. Здесь цель состоит в том, чтобы найти p(s) такое, чтобы общая энергия змеи была сведена к минимуму. Внутренняя энергия определяется как в уравнении(6):

(6)

где *|p(s)*|- величина первой производной, которая больше для более длинных змей. *|p(s)*| является величиной второй производной, которая больше для более резких изгибов. Первая часть держит змею короткой, а вторая часть держит ее прямой. Эти два параметра *α* и *β* определяют относительную важность этих двух терминов, которые обычно являются постоянными. Учитывая изображение на уровне серого I(x, y), типичная внешняя энергия , предназначенная для приведения активного контура к краям шага, которые определены следующим образом в уравнении(7):

(7)

где -двумерная функция Гаусса со стандартным отклонением σ и ∇ является оператором градиента. Легко найти, что больше σ приведет к тому, что границы станут размытыми и искаженными. Однако такой большой σ размер часто необходим для того, чтобы сделать внешнюю энергию достаточно большой, чтобы притянуть змею к этим краям.

Для минимизации энергетической функции мы используем уравнение (8) Эйлера-Лагранжа:

(8)

Когда змея сойдется к минимуму и решение p(s,t) стабилизируется, ее производная по времени будет равна нулю, и мы получим решение вышеупомянутого уравнения Эйлера-Лагранжа.

В общем, нужно инициализировать змею ближе к окончательному решению. Если змея инициализирована “слишком далеко” от границы объекта, возможно, что контур не сможет сойтись на границе объекта. Мы экспериментально обнаружили, что ошибки границ очень распространены из результатов модели snake для клинических применений. На рисуноке 10 показаны экспериментальные результаты модели змеи с различными начальными контурами. Если начальный контур находится далеко от фактического контура, результаты будут неудовлетворительными. Поэтому процесс ручной инициализации необходим для пересмотра исходного контура, чтобы он был близок к фактическому контуру. Между тем, затраченное время значительно сократится, если начальный контур будет близок к фактическому контуру. Кроме того, острые края будут сглажены внутренней энергией змеи, которая сопротивляется высокой кривизне. Поэтому окончательные результаты анализа клинических медицинских изображений могут содержать ошибки границ. На рисунке 11 показан экспериментальный результат модели змеи с хорошим начальным контуром и оптимальными параметрами. Однако границы в зеленых кругах содержат ошибки из-за острых краев с высокой кривизной и размытой границы.

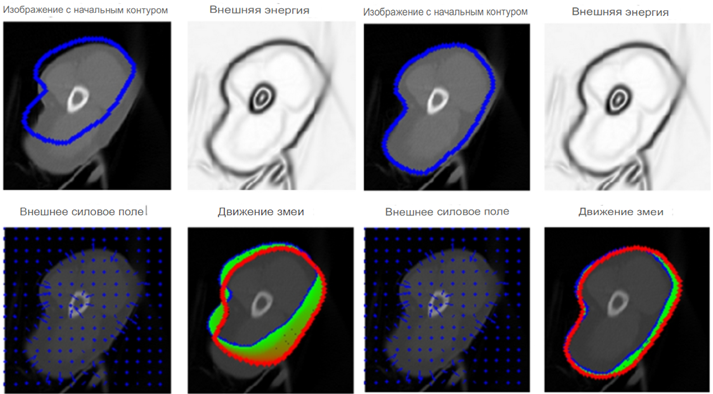


Рисунок 10 - Экспериментальные результаты модели змеи с различными начальными контурами

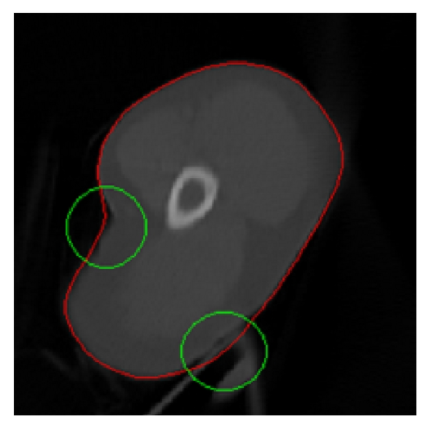


Рисунок 11 - Результат работы модели змеи

Красным цветом показан полученный контур по модели змеи. Границы в зеленых кругах содержат ошибки из-за острых краев с высокой кривизной и размытой границы, соответственно.

Многомасштабное редактирование. Как правило, большинство границ, полученных с помощью модели змеи, могут быть правильными, и несколько сегментов могут содержать ошибки. Если результаты модели змеи идеальны, нет необходимости в дальнейшем редактировании многомасштабной кривой. При ручном редактировании следует исправлять ошибки границ только в ограниченных областях, а другие области границ следует сохранять. Поэтому ручное редактирование должно осуществляться под локальным контролем. Другими словами, работа с одной пограничной областью не должна влиять на другую пограничную область. Кроме того, форма границы может быть неизвестной и очень сложной, что фиксированные контрольные точки могут быть не гибкими для пересмотра. С этой целью мы разрабатываем иерархический многомасштабный подход для создания контрольных точек для пересмотра с помощью кубических кривых Эрмита.

На рисунке 12 показана генерация контрольной точки с помощью предлагаемого иерархического многомасштабного подхода. На первом уровне только 13 контрольных точек генерируются из контура, полученного с помощью модели змеи. Из-за сложной формы границы 13 точек на первом уровне недостаточно для хорошего описания контура. На рисунке 13 подогнанная кубическая кривая Эрмита (синяя) плохо совпадает с исходным контуром (красным) в резких областях. Следовательно, генерируется второй уровень с 25 контрольными точками, а также третий уровень с 50 контрольными точками. Для некоторых простых ROI с плавными границами для их описания достаточно нескольких контрольных точек. Следовательно, ручного редактирования с таким небольшим количеством контрольных точек может быть достаточно для получения многообещающих результатов пересмотра. Однако некоторые анатомические структуры в человеческом теле склонны быть сложными. Например, некоторые области могут быть очень острыми. В этом случае для ручного редактирования необходимо сгенерировать большие контрольные точки. Как показано на рисунках 14 и 15, кубическая кривая Эрмита (синяя) и исходный контур (красный) хорошо совпадают.

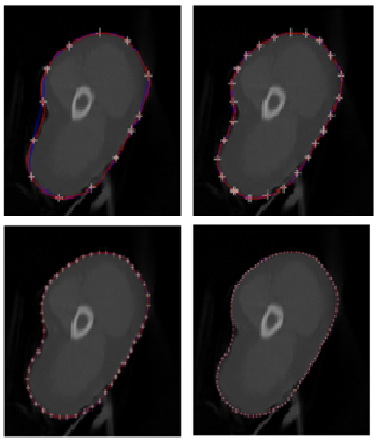


Рисунок 12 – Примеры генерации контрольной точки

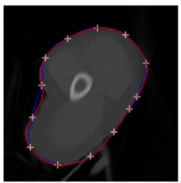


Рисунок 13 – Кубическая кривая Эрмита плохо совпадающая с исходным контуром

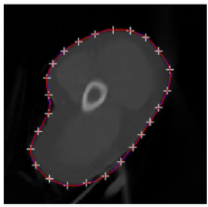


Рисунок 14 - Кубическая кривая Эрмита хорошо совпадающая с исходным контуром

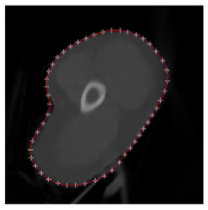


Рисунок 15 - Кубическая кривая Эрмита хорошо совпадающая с исходным контуром

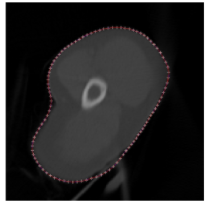


Рисунок 16 - Кубическая кривая Эрмита идеально совпадающая с исходным контуром

Иерархический многомасштабный подход для создания равномерно расположенных контрольных точек для ручного редактирования: рисунок 13 первый уровень с 13 контрольными точками, рисунок 14 второй уровень с 25 контрольными точками, рисунок 15 третий уровень с 50 контрольными точками и рисунок 16 четвертый уровень со 100 контрольными точками. Белый крест показывает равномерно расположенные контрольные точки. Красным цветом показан контур, полученный из модели змеи, а синим цветом показана кривая, сгенерированная кубической кривой Эрмита с соответствующими контрольными точками.

На рисунке 17 показаны результаты ручного редактирования для улучшения контура из модели змеи. Это ясно показывает, что конечный контур очень перспективен, если манипулировать несколькими связанными контрольными точками. На рисунке 18 только 6 связанных контрольных точек с третьего уровня пересмотрены для улучшения контура, созданного моделью змеи. И наоборот, на рисунке 19 представлены 12 связанных контрольных точек четвертого уровня для пересмотра. В основном, большее количество контрольных точек, сгенерированных для ручного редактирования, приведет к более точному очерчиванию контура, а пересмотренная кривая будет точно охватывать фактический контур. Однако большие контрольные точки для пересмотра будут громоздкими, трудоемкими и скучными для пользователей. Поэтому третий уровень подходит для ручного редактирования, а ручная доработка с несколькими связанными контрольными точками может улучшить контур модели змеи. Если клиницист хотел бы создать более точную границу путем ручного редактирования, четвертый уровень был бы лучше, потому что можно изменить больше связанных контрольных точек для создания четких контуров.

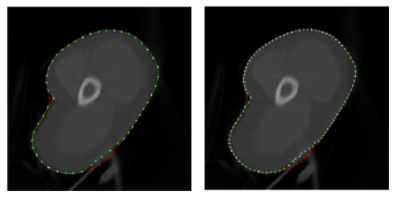


Рисунок 17 – Результаты ручного редактирования

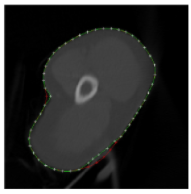


Рисунок 18 – 6 связанных контрольных точек с третьего уровня

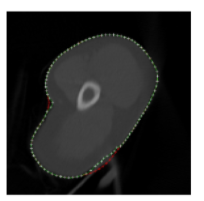


Рисунок 19 - 12 контрольных точек с четвертого уровня

Ручное редактирование: рисунок 18 ручное редактирование с контрольными точками, созданными с третьего уровня, и рисунок 19 ручное редактирование с контрольными точками, созданными с четвертого уровня. Красная линия показывает исходный контур, созданный моделью змеи. Зеленая линия показывает окончательный контур после ручного редактирования. Красный крест показывает соответствующие контрольные точки, которые были пересмотрены.

Преимущество нашего метода для количества контрольных точек состоит в том, чтобы найти компромисс между эффективностью и точностью. Поэтому мы разрабатываем стратегию создания контрольных точек иерархически в соответствии с длиной границы региона. Во-первых, равномерно большое расстояние между сетками контрольных точек создается, когда допустимые ошибки определения конечного контура модели змеи велики. После ручной доработки с соответствующими контрольными точками общее количество ошибок при определении границ будет уменьшено. Для дальнейшего уточнения результатов сетка контрольных точек постепенно уточняется. Расстояние между контрольными точками уменьшается вдвое на каждом шаге. При пересмотре контрольной точки на разных уровнях будет сгенерирован окончательный контур, чтобы пересмотренный контур совпал с фактическим контуром. Процесс будет остановлен до тех пор, пока клиницисты не увидят многообещающие результаты. Очевидно, что он гибок для представления сложных форм путем постепенного уточнения. Это позволяет изменять небольшие контуры с большим количеством контрольных точек, чтобы гарантировать точность. Между тем, эффективно и быстро пересматривать большие контуры с относительно небольшими контрольными точками.

# Глава 3. Программная реализация среды интерактивных инструментов

# Определения средств разработки

Разработка приложений осуществляется в PyCharm. Средства разработки поддерживают все задачи разработки, включая сценарии отладки и локального тестирования. PyCharm — интегрированная среда разработки для языка программирования Python. Предоставляет средства для анализа кода, графический отладчик, инструмент для запуска юнит-тестов и поддерживает веб-разработку на Django. PyCharm разработана компанией JetBrains на основе IntelliJ IDEA.

# Проблема с базовым пороговым значением при извлечении контуров

Обнаружение краев и контуров используются для определения местоположения точек на изображениях, где цвет или яркость сильно меняются.

Нам это может показаться довольно простым и интуитивно понятным, но для компьютера это может оказаться непросто.

Обнаружение краев - это алгоритм, используемый для извлечения краев из изображений, и поскольку он выглядит довольно простым, я считаю, что мы можем начать с него.

Алгоритм состоит из четырех этапов:

1. Выполняет шумоподавление с размытием по Гауссу;
2. Получает направление и величину градиента с Ядро Собеля;
3. Применяется не максимальное подавление, которое удаляет нежелательные пиксели, которые не являются частью контура;
4. Применяет пороговое значение гистерезиса, при котором используются минимальные и максимальные значения для фильтрации контуров по градиенту интенсивности.

Даже несмотря на то, что это выглядит не так просто, реализация осторожного обнаружения довольно удобна в OpenCV.

# Особенности реализации полуавтоматических алгоритмов

Эти усилия часто также усложняются тем фактом, что аннотирование во многих случаях не может быть выполнено непрофессионалами. Даже маркировка чего-то общеизвестного может быть затруднена, если существуют специфические требования к предметной области, которые необходимо выполнить. Это приводит к необходимости обучения аннотаторов. Прежде чем начать процесс аннотирования, аннотатору необходимо убедиться, что выполнены несколько предварительных условий. Во-первых, необходимо иметь достаточный объем данных как для обучения, так и для проверки модели. Качество используемые изображения также должны быть достаточными и согласованными в заданной последовательности. Еще одним важным шагом является обеспечение того, чтобы правила, которым следует аннотация, были четкими и охватывали все возможные случаи, с которыми может столкнуться аннотатор.

Эти правила также должны требовать как можно меньшего толкования, чтобы уменьшить различия между различными аннотаторами. Как только начинается процесс аннотирования изображений, аннотатору необходимо найти баланс между точной аннотацией и отсутствием большого количества времени для каждого отдельного изображения. Одна из распространенных проблем заключается в том, что в зависимости от домена, количество изображений, не требующих аннотации, может составлять большую часть последовательности изображений и занимать много времени только для того, чтобы быть идентифицированным как не отображающее никаких релевантных изображений. Возможным подходом для улучшения этого процесса является использование нейронной сети для изучения аннотации и прогнозирования данных. Также может быть возможно дать прямую обратную связь, чтобы определить, улучшает ли текущая аннотация или ухудшает качество способности моделей к прогнозированию.

Использование новых алгоритмов автоматической сегментации на основе глубокого обучения существенно повышает точность и повторяемость сегментации и объемных измерений печени. Слияние автоматических методов, основанных на ансамблевых подходах, дает наилучшие результаты практически без дополнительных временных затрат из-за возможного параллельного выполнения нескольких моделей.

# ЗАКЛЮЧЕНИЕ

Для эффективного решения задачи аннотации изображений необходимо использовать алгоритмы полуавтоматического выделения объектов. Среди интерактивных инструментов были реализованы бинарная сегментация, активный контур и многомасштабное редактирование кривых. Результаты предназначены для использования в пакете аннотаций изображений для обучения в алгоритмах искусственного интеллекта. Интерактивная разметка предназначена для ускорения работы эксперта, занимающегося подготовкой данных.

# Список использованной литературы:

1. Active Contours – A Method for Image Segmentation in Computer Vision [Электронный ресурс]/ URL: <https://www.analyticsvidhya.com/blog/2021/09/active-contours-a-method-for-image-segmentation-in-computer-vision/#h2_6> - Дата обращения 28.11.2021.
2. Active Contour Model [Электронный ресурс]/ URL: <https://scikit-image.org/docs/dev/auto_examples/edges/plot_active_contours.html> - Дата обращения: 01.12.2021.
3. Active Contour [Электронный ресурс]/ URL: <https://www.sciencedirect.com/topics/engineering/active-contour> - Дата обращения 03.12.2021.
4. Watershed segmentation [Электронный ресурс]/ URL: <https://scikit-image.org/docs/dev/auto_examples/segmentation/plot_watershed.html> - Дата обращщения: 03.12.2021.
5. A. Elnakib, G. Gimel’farb, J. S. Suri, and A. El-Baz, “Medical image segmentation: a brief survey,” in *Multi Modality State-of-the-Art Medical Image Segmentation*, vol. 2, chapter 1, pp. 1–39, Springer, New York, NY, USA, 2011. - Дата обращения 01.12.2021.
6. T. McInerney and D. Terzopoulos, “Deformable models in medical image analysis: a survey,” *Medical Image Analysis*, vol. 1, no. 2, pp. 91–108, 1996. - Дата обращения 28.11.2021.
7. L. W. Chang, H. W. Chen, and J. R. Ho, “Reconstruction of 3D medical images: a nonlinear interpolation technique for reconstruction of 3D medical images,” *CVGIP: Graphical Models and Image Processing*, vol. 53, no. 4, pp. 382–391, 1991. - Дата обращения 03.12.2021.
8. Morphological Snakes [Электронный ресурс]/ URL: <https://scikit-image.org/docs/dev/auto_examples/segmentation/plot_morphsnakes.html> - Дата обращения 01.12.2021.
9. The Interactive Tools Framework in UE4.26 (at Runtime!) [Электронный ресурс]/ URL: <http://www.gradientspace.com/tutorials/2021/01/19/the-interactive-tools-framework-in-ue426> - Дата обращения 03.12.2021.
10. Проектирование программного обеспечения [Электронный ресурс]/ URL: <https://habr.com/ru/post/74330/> - Дата обращения 03.12.2021.
11. C. Li, J. Liu, and M. D. Fox, “Segmentation of edge preserving gradient vector flow: an approach toward automatically initializing and splitting of snakes,” in *Proceeding of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR ’05)*, vol. 1, pp. 162–167, June 2005. - Дата обращения 28.11.2021.
12. J. C. Miller and D. Phill, “Imaging for male infertility,” *Radiology Rounds*, vol. 4, no. 3, 2006. - Дата обращения 28.11.2021.
13. N. Paragios, O. Mellina-Gottardo, and V. Ramesh, “Gradient vector flow fast geometric active contours,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 26, no. 3, pp. 402–407, 2004. - Дата обращения 03.12.2021.
14. Edges and Contours Basics with OpenCV [Электронный ресурс]/ URL: <https://towardsdatascience.com/edges-and-contours-basics-with-opencv-66d3263fd6d1?gi=1a46852ccf90> – Дата обращения 01.12.2021.
15. Watershed Segmentation [Электронный ресурс]/ URL: <https://www.sciencedirect.com/topics/computer-science/watershed-segmentation> - Дата обращения 01.04.2022.
16. Comparison of semi-automatic and deep learning-based automatic methods for liver segmentation in living liver transplant donors [Электронный ресурс]/ URL: <https://www.ncbi.nlm.nih.gov/pmc/articles/PMC7075579/> - Дата обращения 10.04.2022.
17. Automatic Image Segmentation Using Active Contours with Univariate Marginal Distribution [Электронный ресурс]/ URL: <https://www.hindawi.com/journals/mpe/2013/419018/> - Дата обращения 16.04.2022