Слайд про поставленные задачи.

1.Подготовка изображения предназначена для обработки изображения, поступающего на вход, с целью выделения нужного объекта на изображении и его коррекция или коррекция уже готового скана листа. Сначала проверяется, что на изображении – скан или фотография, содержащая лист с нотной нотацией. Далее, в зависимости от того, что на изображении, проводится коррекция изображения.

2. Поиск и обнаружение нотных станов на изображении. Данная подпрограмма предназначена для того, чтобы на обработанном ранее изображении найти нотные станы, для дальнейшего поиска нот на нотных станах.

3. поиск и объектов, похожих на ноту. Блоб – группа связанных пикселей или регионов в цифровом изображении, которые имеют общие характеристики и свойства. Данная подпрограмма используется для поиска блобов на изображении, которые похожи на ноты на нотном листе.

4. Определение высоты каждой ноты. Данная подпрограмма предназначена для определения высоты каждой ноты. (или же просто эта часть дает имя каждой ноте епта)

5. Данная подпрограмма создает MIDI-файл на основе переданных нот.

MIDI (Musical Instrument Digital Interface) – стандартный протокол, используемый для передачи музыкальной информации между электронными инструментами, компьютерами и другими устройствами, связанными с музыкой.

Формат MIDI представляет собой набор числовых данных, которые описывают различные аспекты музыки, такие как нажатие и отпускание клавиш, длительность звуков, силу звучания и другие параметры. MIDI-данные могут быть записаны, сохранены и воспроизведены с помощью поддерживающего MIDI-устройства или программного обеспечения.

Файлы, содержащие MIDI-данные, могут быть воспроизведены на компьютере, синтезаторе, цифровом фортепиано, секвенсоре или другом устройстве. Это позволяет музыкантам записывать, обмениваться и исполнять музыкальные произведения с использованием различных инструментов и звуковых эффектов

2й слайд сам расскажу.

3й.

*Для выполнения этих задач компьютерное зрения использует различные методы и алгоритмы обработки изображения и анализа данных.*

*1.Предобработка изображений. Включает операции такие как изменение размера изображений, фильтрация, сглаживание, улучшение контрастности и яркости, а также удаление шума и искажений, чтобы подготовить изображение к дальнейшему анализу.*

*2.Извлечение признаков. Операции, направленные на извлечение информативных признаков из изображений, таких как границы, углы, текстуры, цвета и формы. Это может включать алгоритмы детектирования краев, дескрипторы особых точек или градиентные методы.*

*3.Классификация и распознавание. Процесс определения и идентификации объектов или образов на изображении. Это может включать обучение моделей машинного обучения, таких как нейронные сети или классификаторы, для автоматического распознавания и классификации объектов.*

*4.Сегментация изображений. Разделение изображения на отдельные сегменты или регионы, обозначающие объекты или области интереса. Это может быть полезно для обнаружения и извлечения объектов из фона или для анализа структуры и контуров объектов.*

*5.Трекинг движущихся объектов: Отслеживание и анализ перемещения объектов на последовательных кадрах видео. Это позволяет отслеживать движущиеся объекты и анализировать их поведение или изменения во времени.*

*6.3D компьютерное зрение. Включает в себя методы и техники для восстановления трехмерной информации из двухмерных изображений или видео. Это может быть полезно для анализа глубины, размеров и пространственной структуры объектов.*

Компьютерное зрение находит широкое применение во многих областях, таких как робототехника, медицина, автоматическое управление, безопасность, анализ изображений и видео, дополненная виртуальная реальность и многие другие. Оно играет ключевую роль в развитии автономных систем, распознавании лиц, *машинном зрении*, медицинской диагностике и многих других приложениях, где обработка и анализ визуальных данных являются критически важными задачами.

7й слайд. На данном слайде мы видим 2 изображения. Слева изображен скан музыкального произведения, справа изображено фото, на котором находится это же музыкальное произведение, распечатанное на листе бумаги.

8й слайд. На этом слайде мы видим обработанные сканы листов. Первое изображение было бинаризировано (то есть, оно состоит всего из 2х цветов – черного (номер 0) и белого (255). Бинаризация пороговая. То есть, сначала изображение цветового формата bgr переводится в оттенки серого. Далее, каждый пиксель оценивается, если его значение ( от 0 до 255) меньше порога, то пиксель становится черным, а если больше порога – белым.

второе изображение (фото с нотным листом) обрабатывается, на нем ищется прямоугольный контур и преобразовании перспективы для получения скана листа. Затем изображение преобразуется в оттенки серого и подвергается пороговому преобразованию для улучшения контраста и яркости текста на листе.

*1.Преобразование изображения RGB в оттенки серого с помощью функции* ***cv2.cvtColor*** *и флага* ***cv2.COLOR\_BGR2GRAY****. Результат сохраняется в переменной* ***gray****.*

*2.Применение размытия с использованием фильтра Гаусса с помощью функции* ***cv2.GaussianBlur****. Размытое изображение сохраняется в переменной* ***blur****.*

*3.Обнаружение краев на размытом изображении с помощью оператора Canny с помощью функции* ***cv2.Canny****. Результат сохраняется в переменной* ***edged****.*

*4.Создание ядра для морфологической операции закрытия с помощью функции* ***cv2.getStructuringElement****.*

*5.Применение морфологической операции закрытия к изображению краев с помощью функции* ***cv2.morphologyEx****. Результат сохраняется в переменной* ***closed****.*

*6.Поиск контуров на изображении с помощью функции* ***cv2.findContours****. Контуры сохраняются в переменной* ***contours****.*

*7.Сортировка контуров по площади в убывающем порядке с помощью функции* ***sorted****.*

*8.Применение алгоритма Douglas-Peucker для упрощения контуров изображения. Ищется контур с четырьмя углами.*

*9.Извлечение координат углов контура и сохранение их в переменных* ***top\_left****,* ***bottom\_right****,* ***top\_right****,* ***bottom\_left****.*

*10.Вычисление максимальной ширины и высоты изображения на основе координат углов.*

*11.Создание массива* ***arr****, содержащего координаты исходного изображения в прямоугольной форме.*

*12.Создание массива* ***rectangle****, содержащего координаты прямоугольника (листа) на исходном изображении.*

*13. Получение матрицы преобразования перспективы с помощью функции* ***cv2.getPerspectiveTransform****.*

*14.Применение преобразования перспективы к исходному изображению с помощью функции* ***cv2.warpPerspective****. Результат сохраняется в переменной* ***dst****.*

*15.Рисование контуров на исходном изображении с помощью функции* ***cv2.drawContours****.*

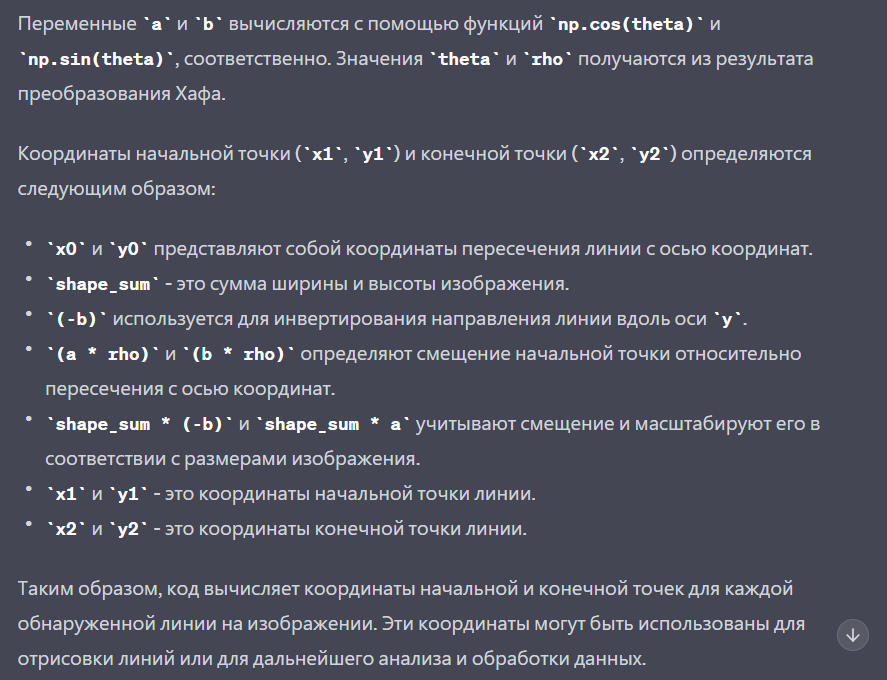
*16.Преобразование результата в оттенки серого с помощью функции* ***cv2.cvtColor****.*

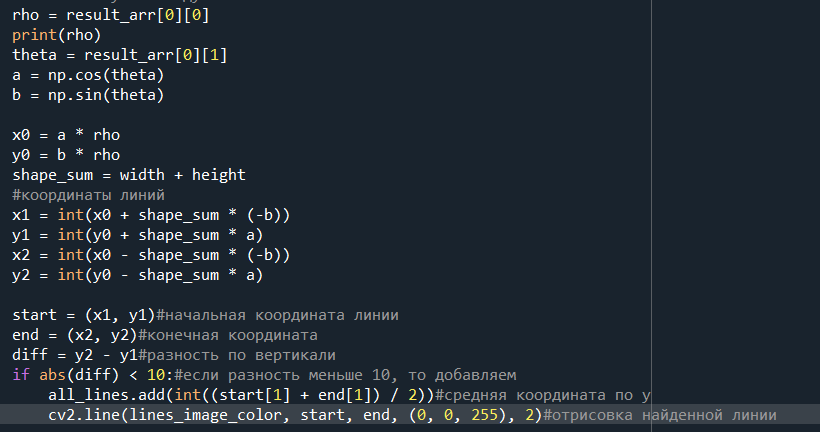
*17.Пороговое преобразование с использованием флагов* ***cv2.THRESH\_BINARY*** *и* ***cv2.THRESH\_OTSU*** *с помощью функции* ***cv2.threshold****. Результат сохраняется в переменной* ***result****.*

*18.Возвращение результата.*

Далее на обоих изображениях происходит поиск линий, после нахождения линий на изображениях ищутся начальные и конечные положения нотных станов (линии – красные, нотные станы – зеленые).

Вначале происходит поиск линий на изображении.

*Перед поиском линий сначала идет поиск контуров. Операция выделения контуров могут быть использованы для поиска областей на изображении.* *Для поиска линий используется преобразование хафа. Преобразование хафа – техника в компьютерном зрении, которая используется для обнаружения геометрических форм, таких как линии или окружности, на изображении. Результат работы функции представляет собой список лтний, каждая из которых задана парой значений (rho, theta), где rho – расстояние от начала координат до линии, а theta – угол, определяющий наклон линии. После получения значений угла и расстояния производится преобразование параметрического представления линии в координаты начальной и конечной точек на изображении.* **

**

Далее, после найденных линий, происходит группировка линий. Обнаруж