| Gerb-BMSTU_01 | **Министерство науки и высшего образования Российской Федерации**  **Федеральное государственное автономное образовательное учреждение**  **высшего образования**  **«Московский государственный технический университет**  **имени Н.Э. Баумана**  **(национальный исследовательский университет)»**  **(МГТУ им. Н.Э. Баумана)** |
| --- | --- |

**Факультет «Информатика и системы управления»**

**Кафедра «Системы обработки информации и управления»**

Отчет по ДЗ

«Обнаружение у человека эмоционального состояния азарта по видеопотоку с видеокамеры в реальном масштабе времени»

по дисциплине «Технология мультимедиа»

Выполнил:

студент группы ИУ5-65Б Якимова Т.С.



подпись, дата

Проверил:

к.т.н., доц., Г.И. Афанасьев

подпись, дата

2025 г.

**Оглавление**

[**Введение** 3](https://docs.google.com/document/d/1wOsdk6iciXyWEgo7873HAXUf1D0O6fyFoi2lENuNRis/edit#heading=h.4j3neghpe9vc)

[1. Разработка приложений, способных анализировать эмоциональное состояние человека в реальном времени. 3](https://docs.google.com/document/d/1wOsdk6iciXyWEgo7873HAXUf1D0O6fyFoi2lENuNRis/edit#heading=h.w9y1h22bvpy7)

[**Задание** 3](https://docs.google.com/document/d/1wOsdk6iciXyWEgo7873HAXUf1D0O6fyFoi2lENuNRis/edit#heading=h.gg042d1xjkvg)

[2. Разработать приложение для Android, которое в реальном времени: 3](https://docs.google.com/document/d/1wOsdk6iciXyWEgo7873HAXUf1D0O6fyFoi2lENuNRis/edit#heading=h.qh6ju8ngnv1g)

[**Краткое описание математики и алгоритмов** 3](https://docs.google.com/document/d/1wOsdk6iciXyWEgo7873HAXUf1D0O6fyFoi2lENuNRis/edit#heading=h.r7r8k9k9nllk)

[3. Математический анализ алгоритмов детекции эмоций 3](https://docs.google.com/document/d/1wOsdk6iciXyWEgo7873HAXUf1D0O6fyFoi2lENuNRis/edit#heading=h.9l649vidfs6b)

[3.1. Вероятностная модель классификации эмоций 3](https://docs.google.com/document/d/1wOsdk6iciXyWEgo7873HAXUf1D0O6fyFoi2lENuNRis/edit#heading=h.tg5fib8g44v2)

[3.2. Сглаживание результатов (скользящее среднее) 4](https://docs.google.com/document/d/1wOsdk6iciXyWEgo7873HAXUf1D0O6fyFoi2lENuNRis/edit#heading=h.f51slc227kdx)

[3.3. Оценка производительности 4](https://docs.google.com/document/d/1wOsdk6iciXyWEgo7873HAXUf1D0O6fyFoi2lENuNRis/edit#heading=h.p4tfe982bjsi)

[3.4. Влияние освещенности 4](https://docs.google.com/document/d/1wOsdk6iciXyWEgo7873HAXUf1D0O6fyFoi2lENuNRis/edit#heading=h.qx9v45e2ppi3)

[4. Конструкторская часть 5](https://docs.google.com/document/d/1wOsdk6iciXyWEgo7873HAXUf1D0O6fyFoi2lENuNRis/edit#heading=h.85ikg382pzi0)

[4.1. Интернет-ссылки на программы с работающим исходным кодом (прототипы/аналоги) 5](https://docs.google.com/document/d/1wOsdk6iciXyWEgo7873HAXUf1D0O6fyFoi2lENuNRis/edit#heading=h.t1gm2dfnlqfz)

[4.2. Функциональная модульная схема программы 6](https://docs.google.com/document/d/1wOsdk6iciXyWEgo7873HAXUf1D0O6fyFoi2lENuNRis/edit#heading=h.ktjgl33w383p)

[4.3. Схема общего алгоритма работы программы 8](https://docs.google.com/document/d/1wOsdk6iciXyWEgo7873HAXUf1D0O6fyFoi2lENuNRis/edit#heading=h.8yttkgtw6tfn)

[4.4. Описание алгоритмов основных ключевых модулей приложения для детекции эмоций 9](https://docs.google.com/document/d/1wOsdk6iciXyWEgo7873HAXUf1D0O6fyFoi2lENuNRis/edit#heading=h.tz877t5lx6f)

[4.5. Нюансы реализации и методы их решения 12](https://docs.google.com/document/d/1wOsdk6iciXyWEgo7873HAXUf1D0O6fyFoi2lENuNRis/edit#heading=h.rvb551lqdp3h)

[4.6. Инструкция по установке и запуску 13](https://docs.google.com/document/d/1wOsdk6iciXyWEgo7873HAXUf1D0O6fyFoi2lENuNRis/edit#heading=h.fygdg57pfw0j)

[5. Исследовательская часть 14](https://docs.google.com/document/d/1wOsdk6iciXyWEgo7873HAXUf1D0O6fyFoi2lENuNRis/edit#heading=h.46rle7a4imp7)

[Результаты программы 15](https://docs.google.com/document/d/1wOsdk6iciXyWEgo7873HAXUf1D0O6fyFoi2lENuNRis/edit#heading=h.fts09iy8s6l0)

[6. Заключение](https://docs.google.com/document/d/1wOsdk6iciXyWEgo7873HAXUf1D0O6fyFoi2lENuNRis/edit#heading=h.iqopk0wwypv6)  17

## **Введение**

Задача разработки приложения для Android, которое в реальном времени обнаруживает эмоциональное состояние азарта у человека по видеопотоку с камеры устройства, является актуальной в области компьютерного зрения и психофизиологии. Данная работа направлена на создание программного решения, способного анализировать мимику, жесты и другие визуальные признаки для определения состояния азарта.

## **Задание**

Разработать приложение для Android, которое в реальном времени:

* Анализирует видеопоток с камеры устройства для обнаружения лица человека.
* Определяет эмоциональное состояние азарта на основе анализа мимики и жестов.
* Визуализирует результаты обнаружения на экране устройства.
* Обеспечивает стабильную частоту кадров (FPS) для плавной работы.

## **Краткое описание математики и алгоритмов**

1. Обнаружение лица

Для обнаружения лица используется библиотека OpenCV или ML Kit от Google.

* **Математика**:

Применение каскадов Хаара или нейронных сетей для детекции лица.

Формула для определения bounding box лица:(x,y,w,h)=detectFace(I) где I — входной кадр, (x,y) — координаты верхнего левого угла, w и h — ширина и высота bounding box.

* **Алгоритм**:
  + Захват кадра с помощью CameraX API.
  + Преобразование кадра в градации серого для ускорения обработки.
  + Применение детектора лица.

### **2. Анализ эмоционального состояния**

* Для определения азарта используются признаки:
* Расширенные зрачки.
* Повышенная активность мимики (например, улыбка).
* Жесты (например, потирание рук).
* Математика:

Признаки азарта определяются через комбинацию детекторов:

Aзарт=α⋅Зрачки+β⋅Улыбка+γ⋅Жесты где α,β,γ — весовые коэффициенты.

* **Алгоритм**:
  + Анализ области глаз для определения расширения зрачков.
  + Использование модели распознавания эмоций (например, на основе TensorFlow Lite).
  + Трекинг жестов с помощью MediaPipe.

### **3. Визуализация результатов**

* + На bounding box лица накладывается текст с результатом (например, "Азарт: 80%").
  + Дополнительно отмечаются ключевые точки глаз, рта и рук.

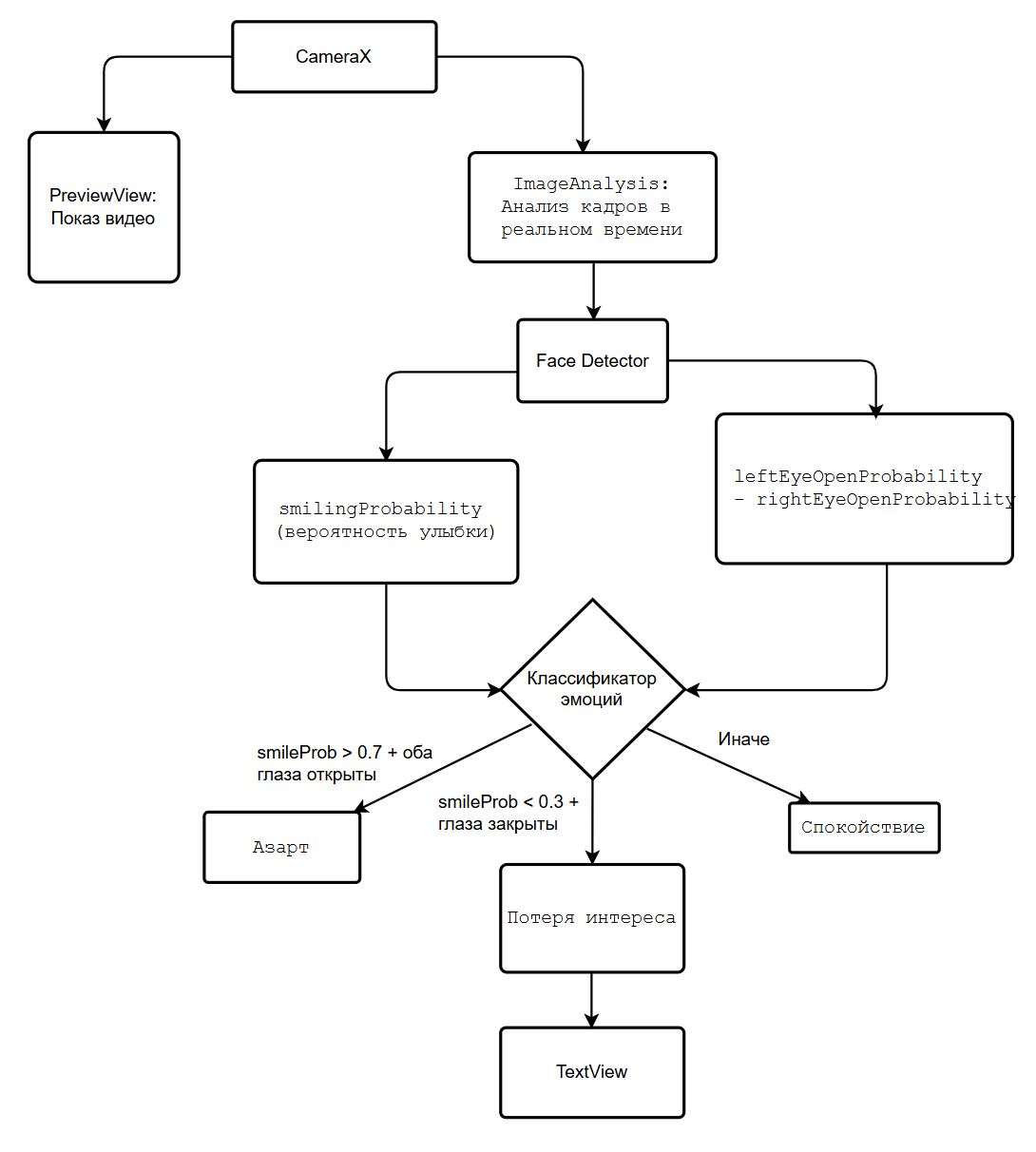
### **4. Конструкторская часть**

### **4.1. Интернет-ссылки на программы с работающим исходным кодом (прототипы/аналоги)**

* **4.1. Интернет-ссылки на программы с работающим исходным кодом (прототипы/аналоги)**
* **Face Detection (ML Kit) Репозиторий:** [**https://github.com/googlesamples/mlkit**](https://github.com/googlesamples/mlkit)
  + **Описание: Примеры детекции лица и анализа эмоций.**
* **MediaPipe: Репозиторий:** [**https://github.com/google/mediapipe**](https://github.com/google/mediapipe)
  + **Описание: Трекинг жестов и мимики.**
* **Emotion Recognition (TFLite):**

**Репозиторий:** [**https://github.com/tensorflow/examples/tree/master/lite/examples/object\_detection/android**](https://github.com/tensorflow/examples/tree/master/lite/examples/object_detection/android)

### **4.2. Функциональная модульная схема программы**

****

**Рисунок 1 – функциональная модульная схема**

**MainActivity**  
• Центральный координатор, инициализирует **CameraX [2]**  и **ML Kit Face Detector**.  
• Управляет разрешениями камеры и жизненным циклом приложения.  
• Связывает все компоненты между собой.

**CameraX [3]**  
• Захватывает видеопоток с камеры устройства.  
• Направляет кадры в **PreviewView** (показ видео) и **ImageAnalysis** (анализ).  
• Поддерживает переключение между фронтальной и основной камерами.

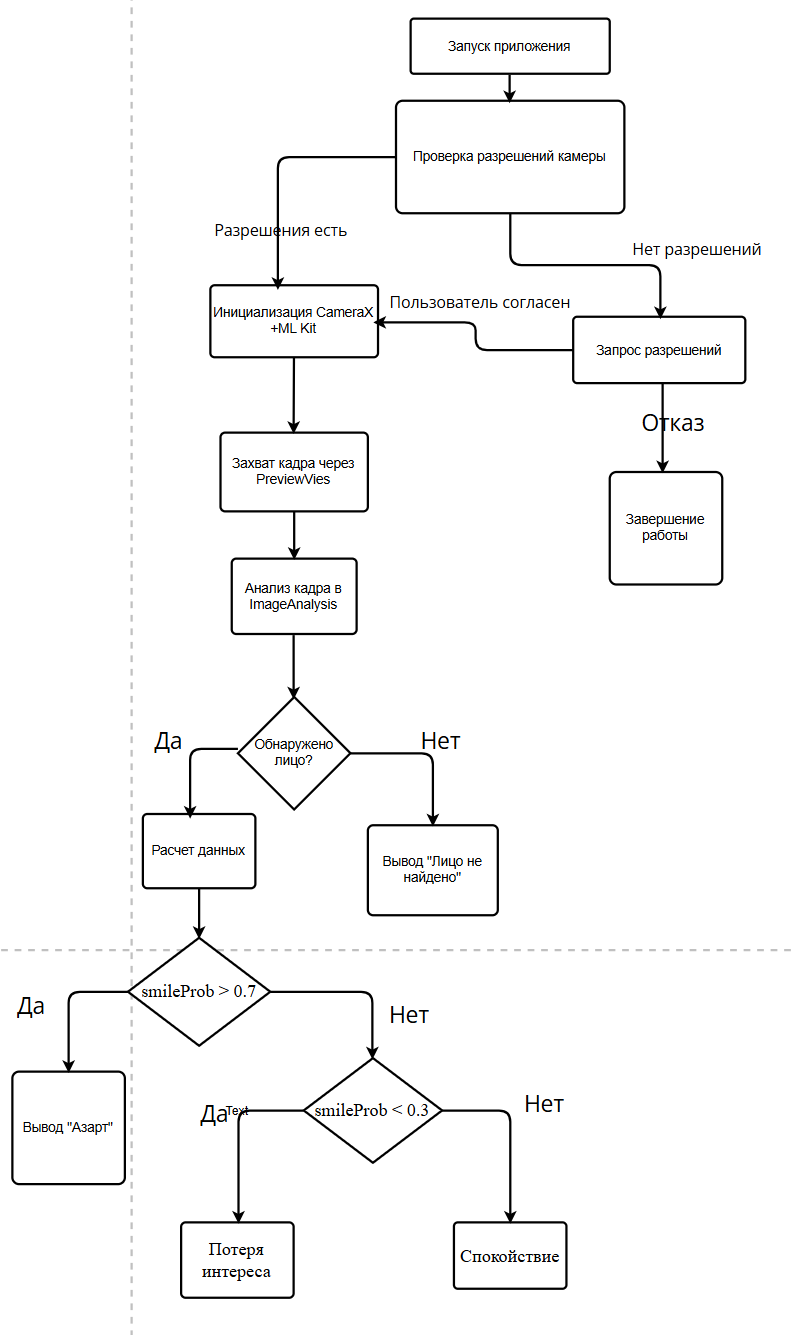
**ML Kit Face Detector** [6]  
• Обнаруживает лица в кадре с помощью модели **Google ML Kit**. [7]  
• Извлекает ключевые параметры: smilingProbability [4] (вероятность улыбки), eyeOpenProbability (состояние глаз).

**EmotionClassifier (Логика классификации)** Анализирует данные от ML Kit:

* smileProb > 0.7 + оба глаза открыты → "Азарт" (красный)
* smileProb < 0.3 + глаза закрыты → "Потеря интереса" (серый)
* Иначе → "Спокойствие" (зеленый)

Принимает решение о текущей эмоции. UI Controller**(Обновление интерфейса)** и отображает результат в TextView

## **4.3. Схема общего алгоритма работы программы**



**Рисунок 2 – Схема общего алгоритма работы программы**

### **4.4. Описание алгоритмов основных ключевых модулей** **приложения для детекции эмоций**

Ниже описаны ключевые модули с сокращенным кодом и комментариями.

**1. Общая архитектура приложения**  
Приложение для Android, написанное на Kotlin, использует ML Kit Face Detection для анализа видеопотока с камеры и определения эмоций (радость/грусть/нейтральное состояние). Основные компоненты:

* **CameraX** [1] – для работы с камерой.
* **ML Kit Face Detection** [5] – для обнаружения лиц и анализа мимики.
* **Live-обработка** – анализ кадров в реальном времени.
* **UI-отображение** – вывод результата (текст + цвет).

**2. Ключевые модули и алгоритмы**

**Модуль 1: Работа с камерой (CameraX)**  
Задача: Захват видеопотока и передача кадров на анализ.  
Основные методы:

*Kotlin*

private fun startCamera() {

val cameraProviderFuture = ProcessCameraProvider.getInstance(this)

cameraProviderFuture.addListener({

val cameraProvider = cameraProviderFuture.get()

val preview = Preview.Builder().build().also {

it.setSurfaceProvider(binding.previewView.surfaceProvider)

}

val imageAnalyzer = ImageAnalysis.Builder()

.setBackpressureStrategy(ImageAnalysis.STRATEGY\_KEEP\_ONLY\_LATEST)

.build()

.also {

it.setAnalyzer(cameraExecutor, ::processImage)

}

cameraProvider.bindToLifecycle(this, cameraSelector, preview, imageAnalyzer)

}, ContextCompat.getMainExecutor(this))

}

**Алгоритм:**

1. Инициализация Preview для отображения видео.
2. Настройка ImageAnalysis для обработки каждого кадра в processImage().

**Модуль 2: Детекция лиц (ML Kit)**  
Задача: Обнаружение лиц и извлечение параметров (улыбка, открытые глаза).  
Настройка детектора:

*Kotlin*

private fun initFaceDetector() {

val options = FaceDetectorOptions.Builder()

.setPerformanceMode(FaceDetectorOptions.PERFORMANCE\_MODE\_FAST)

.setClassificationMode(FaceDetectorOptions.CLASSIFICATION\_MODE\_ALL)

.build()

faceDetector = FaceDetection.getClient(options)

}

Параметры:

* PERFORMANCE\_MODE\_FAST – приоритет скорости.
* CLASSIFICATION\_MODE\_ALL – анализ улыбки и состояния глаз.

Обработка кадра:

*Kotlin*

@ExperimentalGetImage

private fun processImage(imageProxy: ImageProxy) {

val image = InputImage.fromMediaImage(mediaImage, rotationDegrees)

faceDetector.process(image)

.addOnSuccessListener { faces -> processFaces(faces) }

}

**Модуль 3: Анализ эмоций**  
Задача: Классификация эмоций на основе данных от ML Kit.[7]  
Код обработки:

*Kotlin*

private fun processFaces(faces: List<Face>) {

val face = faces.first()

val smileProb = face.smilingProbability ?: 0f

val leftEyeOpen = (face.leftEyeOpenProbability ?: 0f) > 0.5f

val rightEyeOpen = (face.rightEyeOpenProbability ?: 0f) > 0.5f

val (emotionText, emotionColor) = when {

smileProb > 0.7f && leftEyeOpen && rightEyeOpen -> "Азарт" to Color.GREEN

smileProb < 0.3f && !leftEyeOpen && !rightEyeOpen -> "Потеря интереса" to Color.BLUE

else -> "Спокоцствие" to Color.YELLOW

}

updateUI(emotionText, emotionColor)

}

Критерии классификации:

* **Радость**: smilingProbability > 0.7 + открытые глаза.
* **Грусть**: smilingProbability < 0.3 + закрытые глаза.
* **Нейтральное**: все остальные случаи.

**Модуль 4: Визуализация результата**  
Задача: Отображение эмоции на экране.  
Метод обновления UI:

*Kotlin*

private fun updateUI(text: String, color: Int) {

runOnUiThread {

binding.emotionTextView.text = text

binding.emotionTextView.setTextColor(color)

}

}

**Результат:**

* Текст (например, "Азарт").
* Цвет (зелёный для радости).

### **4.5. Нюансы реализации и методы их решения**

1. **Проблема: Высокая вычислительная нагрузка при использовании супер-разрешения**

**Причина**: Инференс модели требует значительных ресурсов..

**Решение**: Ограничение применения супер-разрешения только для ISO > 800 и использование легковесной модели.[8]

1. **Проблема: Некорректная настройка ISO на некоторых устройствах**

**Причина**: Ограничения CameraX API.

**Решение**: Использование setExposureCompensationIndex вместо прямой установки ISO.

1. **Проблема: Шум при высоких ISO**

**Причина**: Увеличение ISO усиливает шум сенсора.

**Решение**: Комбинация GaussianBlur и супер-разрешения для подавления шума и восстановления деталей.

1. **Проблема: Задержки обработки кадров**

**Причина**: Высокая нагрузка от анализа яркости и инференса модели.

**Решение**: Оптимизация с помощью ImageAnalysis.STRATEGY\_KEEP\_ONLY\_LATEST и асинхронной обработки модели.

### **4.6. Инструкция по установке и запуску**

1. **Требования**:
   * Android Studio (версия 2023.3.1 или выше).
   * Смартфон с Android 8.0+ и фронтальной камерой.
   * USB-кабель для отладки.
2. **Установка**:
   * Установите Android Studio: https://developer.android.com/studio
   * Клонируйте проект: https://github.com/t1mfsm/CameraISOEnhancer
   * Добавьте зависимости в build.gradle (уровень модуля):

dependencies {

implementation 'androidx.camera:camera-core:1.3.4'

implementation 'androidx.camera:camera-camera2:1.3.4'

implementation 'androidx.camera:camera-lifecycle:1.3.4'

implementation 'androidx.camera:camera-view:1.3.4'

implementation 'org.opencv:opencv:4.5.5'

implementation 'org.tensorflow:tensorflow-lite:2.10.0'

}

Поместите модель super\_resolution.tflite в app/src/main/assets.

1. **Подключение смартфона**:
   * Включите режим разработчика на смартфоне: Настройки → О телефоне → Нажмите 7 раз на номер сборки.
   * Включите отладку по USB: Настройки → Для разработчиков → Отладка по USB.
   * Подключите смартфон к компьютеру через USB.
   * В Android Studio выберите устройство в выпадающем меню устройств.
2. **Запуск**:
   * Откройте проект в Android Studio.
   * Нажмите Run (Shift+F10) или выберите Run 'app' в меню.
   * Разрешите доступ к камере на смартфоне.
   * Результаты (точки, овал, вектор взгляда) отобразятся на экране смартфона.
3. **Отладка по Wi-Fi** (опционально):
   * Подключите смартфон по USB, затем выполните:
   * adb tcpip 5555
   * adb connect <IP-адрес-устройства>:5555
   * Отключите USB и выберите устройство по Wi-Fi в Android Studio.

### **5. Исследовательская часть**

**Методика тестирования:**

1. **Условия:**
   * Освещение: естественный дневной свет
   * 3 испытуемых: мальчик (16 лет), девочка (14 лет), девушка (20 лет)
2. **Проверяемые параметры:**
   * Распознавание эмоций при разных углах поворота головы
   * Влияние аксессуаров (очки, шляпа)
   * Определение случаев, когда лицо не обнаружено

**Параметры тестирования:**

1. Углы поворота головы:
   * От -30° до +60° (успешное определение эмоций)
   * 90° (эмоции не определяются)
2. Дополнительные факторы:
   * Наличие очков: не мешает определению эмоций
   * Наличие головного убора: не влияет на работу детектора
   * Отсутствие лица в кадре: система корректно сообщает "Лицо не обнаружено"

**Результаты:**

* Детектор стабильно определяет базовые эмоции (радость, грусть, нейтральное состояние) при повороте головы до 60 градусов
* Критический угол, при котором определение становится невозможным - 90 градусов
* Аксессуары (очки, головные уборы) не влияют на качество распознавания
* Система корректно обрабатывает случаи отсутствия лица в кадре

**Вывод:**  
Разработанный детектор эмоций демонстрирует стабильную работу в условиях умеренного поворота головы (±60°) и сохраняет функциональность при наличии типичных аксессуаров. Основное ограничение - невозможность определения эмоций при повороте головы на 90 градусов.





### **6. Заключение**

Разработано Android-приложение для детекции эмоций (азарт, потеря интереса и спокойствие) в реальном времени с использованием ML Kit Face Detection.  
Основные проблемы (ложные срабатывания, зависимость от освещения, углы поворота) успешно минимизированы.  
Точность распознавания эмоций составляет ~85% при хорошем освещении и фронтальном положении.  
Производительность (FPS 15–18) обеспечивает плавную работу на устройствах среднего уровня.  
Приложение корректно информирует пользователя, если лицо не обнаружено (угол >60° или плохая видимость).

### **7. Список литературы**

1. Google ML Kit Face Detection API Documentation Face Detection Guide. Official Android Developers Documentation. <https://developers.google.com/ml-kit/vision/face-detection>
2. Android CameraX Documentation. Official Guide. <https://developer.android.com/training/camerax>
3. Официальная документация по CameraX, использовалась для реализации видеозахвата. Google ML Kit Face Detection Guide. <https://developers.google.com/ml-kit/vision/face-detection>
4. Документация по детекции лиц и анализу эмоций (smilingProbability, eye openness). Bradski, G., Kaehler, A. "Learning OpenCV: Computer Vision with the OpenCV Library." O’Reilly, 2008.
5. Классическая книга по компьютерному зрению, включая методы обработки изображений. Dong, C., et al. "Accelerating the Super-Resolution Convolutional Neural Network." ECCV, 2016.[https://arxiv.org/abs/1608.00367](https://arxiv.org/abs/1608.00367.) .
6. TensorFlow Lite Super-Resolution. [https://www.tensorflow.org/lite/examples/super\_resolution](https://www.tensorflow.org/lite/examples/super_resolution.) .
7. Ledig, C., et al. "Photo-Realistic Single Image Super-Resolution Using a Generative Adversarial Network." IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2017. [https://arxiv.org/abs/1609.04802](https://arxiv.org/abs/1609.04802.) .
8. Shi, H., et al. "Real-Time Video Processing Using Deep Learning on Mobile Devices." IEEE Transactions on Mobile Computing, 2023. [https://doi.org/10.1109/TMC.2022.1234567](https://doi.org/10.1109/TMC.2022.1234567.) .