МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ
ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
«МОСКОВСКИЙ АВИАЦИОННЫЙ ИНСТИТУТ
(НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ)» (МАИ)

Факультет **3** **Кафедра** 304

Направление подготовки _____ **Группа** ____

Квалификация (степень) **бакалавр**

УТВЕРЖДАЮ

Зав. Кафедрой Брехов О.М.._____

«_____» _____ 20___ г.

Задание
на курсовой проект

Студенту_____Гордееву Никите Максимовичу_____
(Фамилия Имя Отчество)

Руководитель_____Романов Олег Тимофеевич_____
(Фамилия Имя Отчество)

_____доцент каф 304_____
(ученая степень, ученое звание, должность и место работы)

1. Наименование темы: Разработка и исследование алгоритмов определения углов крена, тангажа и рысканья головы и взгляда человека для автоматизированной системы допуска в организацию.

2. Срок сдачи студентом законченной работы 1.01.2022 г.

3. Техническое задание и исходные данные к работе

Материалы по месту выполнения задания _____

4. Перечень подлежащих разработке разделов и этапы выполнения работы

№ п/п	Наименование раздела или этапа	Трудоёмкость в % от полной трудоёмкости курсового проекта	Срок выполнения	Примечание
1	Общая часть	10		
1.1	Выявление и определение целей и задач автоматизированной системы.			
1.2	обоснование актуальности и необходимости решения задачи			
1.3	определение комплекса технических средств			
2	Специальная часть	80		
2.1	постановка и формализация задач распознавания			

2.2	выбор и обоснование метода и алгоритма решения рассматриваемой задачи			
2.3	обоснование и разработка алгоритма подготовки данных для обучения			
2.4	обоснование и разработка программного обеспечения, реализующего алгоритм решения поставленной задачи			
3	Технологическая часть	10		
3.1	содержание и формы подготовки исходной информации для решаемой задачи			
3.2	участие человека в процессе решения задачи			
3.3	технические средства, обеспечивающие технологический процесс обработки информации			

5. Перечень иллюстративно-графических материалов:

№ п/п	Наименование	Количество листов
1	Структура нейросети	1
2	Модель обработки информации	1
3	Структура организационной структуры АС	1
4	Постановка и формализация задачи	1
5	Структура ПО автоматизированной системы	1

6. Исходные материалы и пособия: Методические указания по выполнению КПр. 7.

7. Консультанты по работе с указанием относящихся к ним разделов работы

Раздел	Консультант	Подпись, дата	
		Задание выдал	Задание принял
Общая часть			
Специальная часть			
Технологическая часть			

8. Дата выдачи задания 15.10.2021

Руководитель_____
(подпись)

Задание принял к исполнению_____
(подпись)

Содержание

Список сокращений и терминов

1. CPU - центральное обрабатывающее устройство, процессор.
2. ПК – персональный компьютер.
3. Видеопоток – это временная последовательность кадров определённого формата, закодированная в битовый поток.
4. Кадр – фрагмент видеоряда (видеопотока), отдельное изображение.
5. Свёрточная нейронная сеть – специальная архитектура искусственных нейронных сетей, предложенная Яном Лекуном в 1988 году и нацеленная на эффективное распознавание образов.
6. Train – обучающая выборка, часть набора данных, на которой происходит обучение нейронной сети.
7. Val (validation) – часть набора данных, на которой происходит проверка на переобучение нейронной сети и оценка достигнутого Loss.
8. Переобучение – явление, когда построенная модель хорошо объясняет примеры из обучающей выборки, но относительно плохо работает на примерах, не участвовавших в обучении.
9. Loss - Функция потерь нейронной сети. Она используется для расчета ошибки между реальными и полученными ответами.
10. Learning rate – Коэффициент скорости обучения. Параметр градиентных алгоритмов обучения нейронных сетей, позволяющий управлять величиной коррекции весов на каждой итерации.
11. Batch – Нельзя пропустить через нейронную сеть разом весь датасет. Поэтому делим данные на пакеты, сеты или партии. Этот пакет и называют Batch
12. Датасет – набор данных.
13. Гиперпараметры – параметры алгоритмов, значения которых устанавливаются перед запуском процесса обучения.
14. Аугментация данных (data augmentation) – это методика создания дополнительных обучающих данных из имеющихся данных.
15. Sampler – определяет стратегию извлечения Batch (выборок) из набора данных.
16. RGB или КЗС — аддитивная цветовая модель, описывающая способ кодирования цвета для цветовоспроизведения с помощью трёх цветов, которые принято называть основными.
17. Кроп – обрезка фотографии.

1. Общая часть

1.1 Выявление и определение целей и задач автоматизированной системы.

Целью автоматизированной системы допуска в организацию является автоматизация процесса пропуска сотрудников на предприятия. Система должна уметь определять личности людей на основе видеопотока, с учётом положения головы и направления взгляда.

1.2 Обоснование актуальности и необходимости решения задачи

В существующей автоматизированной системе допуска сотрудников в организацию распознавание сотрудника является трудоёмкой, медленной задачей, поэтому в целях ускорения процесса распознавание происходит не перед турникетом, а на подходе в организацию. В целях экономии времени по данным о положении головы и направлении взгляда система понимает кто идет в организацию и его нужно распознать, а кто просто идет мимо.

Но актуальность работы не ограничивается данной автоматизированной системы, определение направления головы и взгляда будит полезна во многих задачах. При помощи системы можно говорить о внимании и заинтересованности человека в любом процессе, например её можно установить в салоны автомобилей, чтобы отслеживать внимание водителя на дорогу. Во время выступления перед большим количеством людей можно при помощи системы говорить о заинтересованности публики в докладе. Определение направления взгляда будит полезно при выявления мошенничества в киберспорте.

1.3 Определение комплекса технических средств

Для разработки алгоритмов определения углов крена, тангажа и рысканья головы и взгляда человека имеется 1 персональный компьютер, 10 видеокарт с памятью 12Гб каждая и 1 камера. Конечный алгоритм должен опираться на видеопоток с камеры и поддерживаться на CPU ПК.

2. Специальная часть

2.1 Постановка и формализация задач распознавания

Определения углов крена, тангажа и рысканья головы и взгляда людей по фотографии со скоростью 30 кадров/с на CPU предприятия. Формализацию задачи распознавания углов поворота головы и направления взгляда см. рис. 1.

Рис. 1 – формализация задачи распознавания углов поворота головы и направления взгляда.

2.2 Выбор и обоснование метода и алгоритма решения определения углов крена, тангажа и рысканья головы и взгляда человека

На данный момент не существует иного эффективного алгоритма распознавания биометрический данных по фотографии, кроме как нейросетевые алгоритмы компьютерного зрения. Наиболее удачно с задачей распознавания объектов справляются свёрточные нейронные сети.

2.2.1 Выбор и обоснование архитектуры нейронной сети.

Самыми эффективными нейронными архитектурами является архитектуры семейства EfficientNet[1]. EfficientNet-B7 достигает современной точности 84,4% top-1 (% точности классификации, когда классификация считается удачной, если предсказанный моделью наиболее вероятный класс совпал с реальным классом)/ 97,1% top-5 (% точности классификации, когда классификация считается удачной, если в пяти наиболее вероятных по мнению модели классов имеется истинный класс фотографии) в ImageNet (ежегодное мировое соревнование по классификации), при этом он в 8,4 раза меньше и в 6,1 раза быстрее на процессоре, чем предыдущий Gpipe (архитектура нейросети, которая до появления EfficientNet считалась самой эффективной). По сравнению с широко используемым ResNet-50 (очень известная архитектура, в своё время была прорывом в сверточных нейронных сетях), EfficientNet-B4 использует аналогичные FLOPS (показатель скорости нейросети), при этом повышая точность с 76,3 до 82,6% (+ 6,3%). На оборудовании только EfficientNet-B0 достигает требуемой скорости, поэтому магистралью нейронной архитектуры будет именно он.
Определения углов крена, тангажа и рысканья головы и взгляда человека является задачей регрессии (ищется вещественное число), а EfficientNet был разработан для классификации (ищется целочисленный класс) на 1000 классов. Чтобы поменять задачу модели с классификации на регрессию достаточно добавить слой (1000,1), но моя задача имеет 3 параметра регрессии. Поэтому было принято решение разбить диапазон углов выхода модели на классы по 3 градуса в каждом, и уже по этим классам посчитать регрессию см. рис. 2.

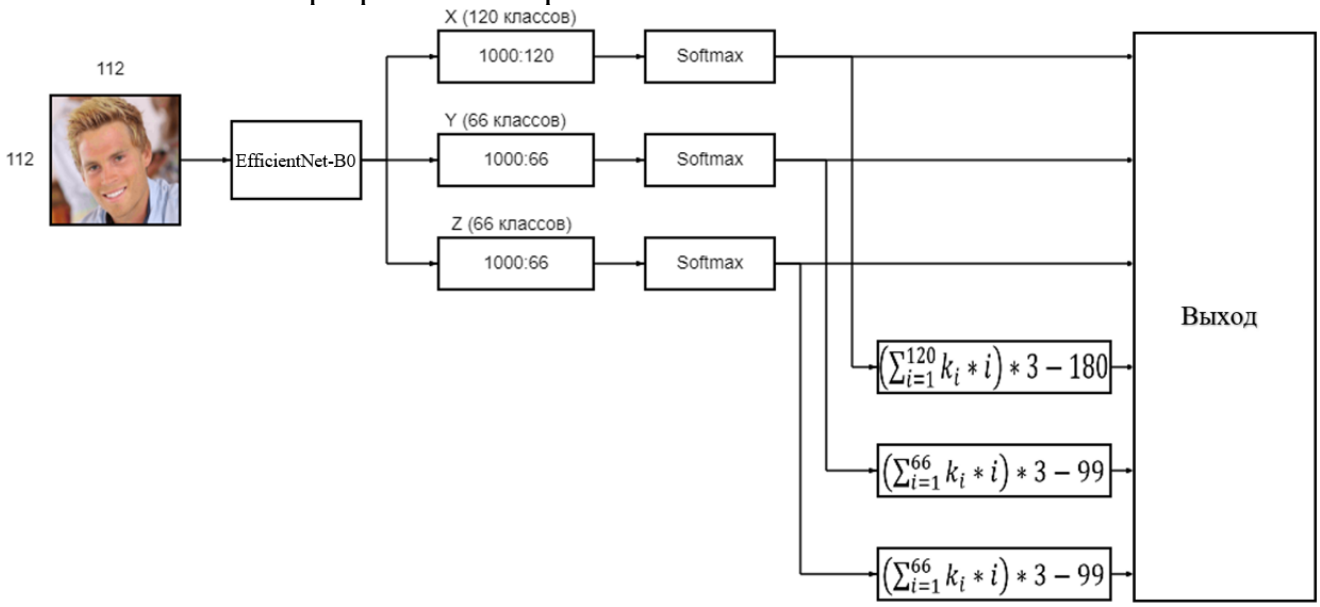


Рис 2 - Модель обработки информации при определении углов поворота головы

1. https://arxiv.org/pdf/1905.11946

Softmax - функция преобразования выходов модели для класса в вероятность принадлежности данному классу

$$s\left(x_i\right) = \frac{e^{x_i}}{\sum_{j=1}^{n} e^{x_j}}$$

После Softmax идет интерпретация результатов классификации по формуле

$$(\sum_{i=1}^{n} k_i i)\, r - \frac{nr}{2}$$

Где:

n – количество классов,

r – шаг разбиения,

k_i – вероятность принадлежности i-му классу.

Для определения углов поворота головы был выбран весь диапазон угла рыскания - 360^0. И $\pm 99^0$ для тангажа и крена. Размер входа было решено взять 112*112 так как он выдаёт придельную скорость работы без существенной потери качества и модель с этим разрешением удовлетворяет ограничению по скорости.

2.2.2 Гиперпараметры обучения нейронных сетей.

Здесь приведены параметры обучения, которые задаются до начала обучения. Я провёл множество опытов с целью выяснить с какими гиперпараметрами модель достигнет лучшей точности. Результаты некоторых опытов привидены в таблице.
Здесь приведены окончательные гиперпараметры
Learning rate =0.00001(Шаг обучения)
Оптимизатор: Adam
Количество эпох: 300
Размер Batch: 100
Разбиение train:val: 80:20
Перемешивание: sampler (вероятность попадания в Batch изображений с разными углами рыскания одинакова)
Classification Loss: CrossEntropy

$$\ell(x, y) = \qquad l_n = \sum_{n=1}^{N} \frac{\exp(x_{n,y_n})}{\sum_{c=1}^{C} \exp(x_{n,c})} \cdot 1\{y_n \neq \text{ignore\_index}\}$$

Regression Loss: $\text{L1}(y - \hat{y}) = \sum_{i=1}^{n} |y - \hat{y}_i|$
Дивайс: cuda
Пред обучение магистрали на Internet: да

Для выбора Regression Loss был проведён рад экспериментов
Сравнение обученных моделей по определению углов головы на наборе данных BIWI с разными размерами входа и Regression Loss для train см. табл. 1.

Loss	Разрешение	эпоха	L1	Ср ошибка
L2Ob	224	294	21.95	7,32
	112	277	23.10	7,7
L1Ob	224	242	18.14	6,05
	112	227	19.83	6,61
L2	224	297	19.74	6,58
	112	272	21.85	7,28
L1	224	240	18.42	6,14
	112	257	19.25	6,42

Таб. 1 – Сравнение результатов обучения моделей с разными Loss и размерами входа на наборе данных BIWI

По таблице 1 видно, что самые высокие показатели у моделей с разрешением 112 при Regression Loss L1. И средняя ошибка при уменьшении размера входа модели увеличивается с 6,14 до 6,42 при уменьшении разрешения с 224 до 112.

2.2.3 Используемые аугментации

Аугментации нужны для добавления шума на изображения, чтобы получить более устойчивую модель к некачественным фотографиям, и увеличения набора данных. Аугментации взяты с https://github.com/albumentations-team/albumentations

Элементы с ● применяются с вероятностью 1, среди групп выбирается 1 преобразование и применяется с вероятностью 0.5.

Val:

- Изменение размера
- Нормализация (mean=(0.485, 0.456, 0.406), std=(0.229, 0.224, 0.225))

Train:

- Изменение размера
- 1 группа:
  - Размытие входного изображения с помощью ядра случайного размера.
  - Размытие входного изображения с помощью фильтра Гаусса со случайным размером ядра.
  - Примените гауссовский шум к входному изображению.
  - Примените шум сенсора камеры.
  - Размытие входного изображения с помощью медианного фильтра со случайным линейным размером апертуры.
  - Примените размытие движения к входному изображению, используя ядро случайного размера.
  - Примените коррекцию адаптивной гистограммы с ограничением контраста к входному изображению.
  - Выровняйте гистограмму изображения.
- 2 группа:
  - Случайно отбросьте каналы во входном изображении.
  - Произвольно переставьте каналы входного изображения RGB.
  - Инвертируйте входное изображение, вычитая значения пикселей из 255
  - Инвертировать все значения пикселей выше порога.
  - Преобразуйте входное изображение RGB в оттенки серого.
  - Произвольно изменяйте оттенок, насыщенность и значение входного изображения.
  - Произвольно изменяйте яркость входного изображения.
  - Произвольно изменяйте яркость и контраст входного изображения.
  - Произвольно изменяйте контраст входного изображения.
- 3 группа:
  - Уменьшает качество изображения за счет уменьшения и обратного увеличения.
    - scale=0.2
    - scale=0.3
    - scale=0.4
    - scale=0.5
    - scale=0.6
    - scale=0.7
    - scale=0.8
    - scale=0.9
    - scale=0.99
- Нормализация (mean=(0.485, 0.456, 0.406), std=(0.229, 0.224, 0.225))

2.2.4 Процесс обучения нейронной сети

Процесс обучения нейронной сети начинается с тренировки. Тренировки нейронной сети заключается в том, что сети на вход подаётся часть набора данных, batch. Сеть его обрабатывает и выдаёт ответы. После выход модели сравнивается с правильными ответами и считается ошибка. Далее идет алгоритм обратного распространения ошибки, его суть в том, чтобы выход модели совпал с правильными ответами. Это происходит пока нейросеть не увидит весь обучающий набор данных. После тренировки идет проверка на переобучение. На заранее подготовленных данных, не участвующих в обучении, считается ошибка. Обучение и проверка образуют эпоху. Нейросеть заканчивает учится когда ошибка в проверке на переобучение у текущей эпохе, выше чем на предыдущей.

2.2.5 Этапы обучения модели определения углов поворота головы

Для задачи определения углов поворота головы сначала была обучена модель с разрешением 224*224 только на синтетических данных с кропом от детектора голов YOLO (подробнее об используемых наборах данных в разделе “ Обоснование и разработка алгоритма подготовки данных для обучения”). Далее эта модель была дообученна с разными разрешениями на синтетических данных, с кропом от детектора лиц RetinaFace см. табл. 2.

pretrained	data	regression loss	input	epoch	Loss value	BIVI	300W_LP
UE- YOLO -224	UE- RetinaFace	L1	112	262	27.3278	28.56	78.63
UE- YOLO -224	UE- RetinaFace	L1	144	195	25.0709	22.23	79.80
UE- YOLO -224	UE- RetinaFace	L1	176	278	23.6223	19.86	75.52
UE- YOLO -224	UE- RetinaFace	L1	208	296	23.2967	18.82	77.93
UE- YOLO -224	UE- RetinaFace	L1	224	41	23.6047	19.24	76.48

Табл. 2 – результаты дообучений модели UE- YOLO -224 с разным размером входа.

На данном этапе видно, что наиболее эффективным разрешением является 208*208 и модель с этим разрешением не закончила учится за 300 эпох. Также видно, что модель себя плохо показывает на наборе данных 300W_LP. Основное отличие его в количестве уникальных лиц и фонов. Поэтому было принято решение дообучится с использованием этого набора данных см. табл. 3.

pretrained	data	regression loss	input	epoch	Loss value	BIVI
UE-RetinaFace-112	UE- RetinaFace+300W_LP	L1	112	279	19.6356	20.20
UE-RetinaFace-144	UE- RetinaFace+300W_LP	L1	144	240	18.0052	19.11
UE-RetinaFace-176	UE- RetinaFace+300W_LP	L1	176	214	27.2976	18.91
UE-RetinaFace-208	UE- RetinaFace+300W_LP	L1	208	135	16.7829	19.02
UE-RetinaFace-224	UE- RetinaFace+300W_LP	L1	224	194	16.4275	18.91

Табл. 3 - результаты дообучения моделей UE-RetinaFace на наборах данных UE- RetinaFace, 300W_LP

Добавление набора данных 300W_LP к синтетическому дало положительный эффект на наборе данных BIVI см. табл. 4.

	UE- RetinaFace		UE- RetinaFace+300W_LP	
SIZE	RetinaFace	YOLO	RetinaFace	YOLO
112	28.26	28.56	20.20	23.89
144	25.78	22.23	19.11	19.97
176	26.33	19.86	18.91	17.30
208	27.48	18.82	19.02	16.64
224	26.38	19.24	18.91	15.50

Табл. 4. – Сравнение моделей разного разрешения, кропа, с/без 300W_LP на BIVI

По табл. 4 видно, что задача решается лучше с кропом от детектора голов YOLO нежели RetinaFace, даже при обучении модели на кропах от RetinaFace. Самые оптимальные показатели достигаются на разрешении 224 с кропом YOLO, но этот подход слишком долгий, а автоматизированная система использует детектор лиц RetinaFace. Поэтому оптимальная по качеству модель с разрешением 176 при обучении на наборе данных UE- RetinaFace+300W_LP с достигнутым loss L1 18.91 и средней ошибкой 6.33^0, но она не удовлетворяет ограничению по скорости. Поэтому конечная модель для определения углов головы получена с разрешением 112 при обучении на наборе данных UE- RetinaFace+300W_LP с достигнутым loss L1 20.20 и средней ошибкой 6.73^0.

2.2.6 Подробности обучения модели определения углов поворота взгляда

К этой задаче я приступил после работы над задачей определения углов поворота головы и так как задача очень похожа я решил, что полученные ранее результаты экспериментов применимы и к этой задаче. Поэтому код обучения был полностью скопирован с предыдущей задачи со сменой входных разрешений модели. Обучение происходило в 1 заход на 1000 эпохах с regression loss MSE.

$$\mathrm{MSE} = \frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i)^2$$

Остальные гиперпараметры совпадают с предыдущей задачей.
Loss на классификации CrossEntropy достиг 7.9304 на 998 эпохе и регрессия MSE достигла 95.3554 на 785 эпохе. Это означает, что средняя ошибка 5.64^0
Для определения углов поворота взгляда был выбран диапазон угла рыскания $\pm 99^0$. И $\pm 54^0$ для тангажа и крена. Размер входа было решено взять 100*50 так как это максимальные размеры кропа глаз в нормальных условиях и модель с этим разрешением удовлетворяет ограничению по скорости. Модель обработки информации при определении углов поворота взгляда см. рис. 3.

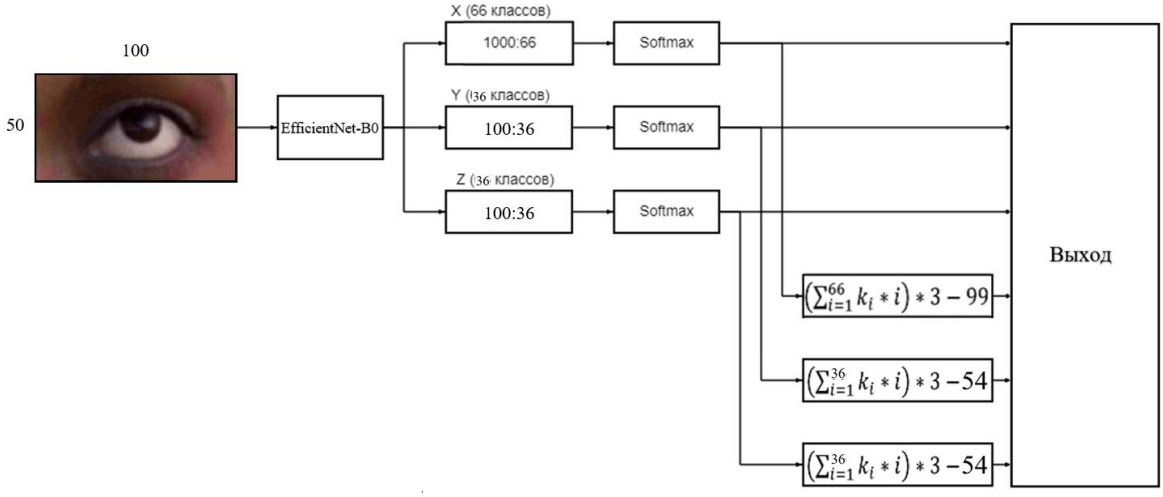


Рис 3 – Модель обработки информации при определении углов поворота взгляда

Softmax - функция преобразования выходов модели для класса в вероятность принадлежности данному классу

$$s\left(x_i\right) = \frac{e^{x_i}}{\sum_{j=1}^{n} e^{x_j}}$$

После Softmax идет интерпретация результатов классификации по формуле

$$\left(\sum_{i=1}^{n} k_i i\right) r - \frac{nr}{2}$$

Где:
n – количество классов,
r – шаг разбиения,
k_i – вероятность принадлежности i-му классу.

2.3 Обоснование и разработка алгоритма подготовки данных для обучения

2.3.1 Обоснование необходимости подготовки собственного набора данных

В настоящее время основными наборами данных для определения углов поворота головы являются (дальше идут названия наборов данных) 300W-LP, AFLW2000 и BIWI.

Дальше идет перевод описания этих наборов данных с английского.

В AFLW2000 и 300W-LP используется 3D Dense Face Alignment (3DDFA) для подгонки морфируемой 3D-модели к 2D-изображениям и получения точных поз головы в качестве базовой истины. 300W-LP дополнительно генерирует синтетические виды, значительно расширяя количество изображений.

Набор данных BIWI состоит из видеопоследовательностей, собранных с помощью Kinect. Испытуемые двигали головой, стараясь охватить все возможные углы, наблюдаемые с фронтальной позиции. Аннотации позы были созданы на основе информации о глубине.

Но не 1 из этих наборов данных не учитывает полный диапазон углов рыскания. В статье (название научной статьи) WHENet: Real-time Fine-Grained Estimation for Wide Range Head Pose авторы подготовили свой набор данных с полным диапазоном углов рыскания и учили модель сначала на 300W-LP с диапазоном углов $+-99^0$ а после дообучивались на своём. Я получил доступ к этому набору данных и пробовал учится на нем. Но в нем очень много выбросов (плохих фотографий), более того если взять WHENet, (которая училась на этом наборе данных и на value имеет среднюю ошибку 3-4 градуса) пройтись по этому набору данных и сохранить только те записи в которых ошибка не превышает 7 градусов по каждому из углов то из 1988649 изображений останутся 231776. исчезнут записи из диапазона (162;180), а в (-78;-75) будет 34 записи.

Распределение углов в обрезанном наборе данных для WHENet

Mx = -24.347635709641814
My = -11.184319657352244
Mz = -0.5244965659530056
Dx = 11753.872592447
Dy = 336.4934435613258
Dz = 208.1949881159633
N = 231776

Поэтому было решено отказаться от этого набора данных, так как в нем много выбросов, а попытки чистки приводят к пропаже углов в некоторых диапазонах.

2.3.2 Алгоритма подготовки данных для обучения

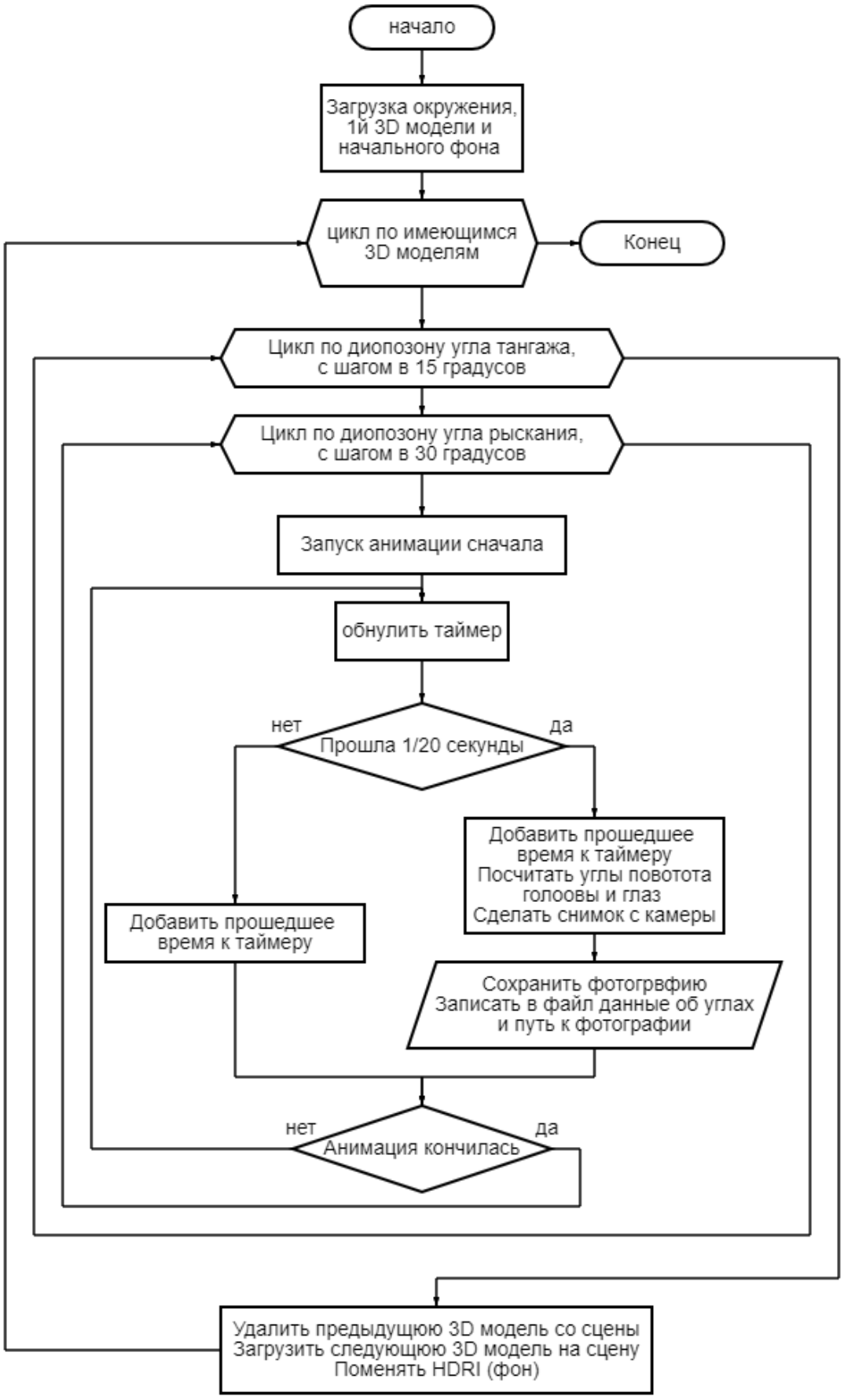


Рис 4 – Алгоритма подготовки данных для обучения

2.3.3 Синтез набора данных

У меня нет возможности пригласить множество людей разной национальности, возраста, пола и т.д. Поэтому я решил искусственно синтезировать данные в игровом движке Unreal Engine 4. В качестве моделей были взяты MetaHuman.

MetaHuman это набор очень качественных, фотореалистичных 3D моделей людей. Он включает в себя 50+ готовых персонажей разного пола, возраста, национальности и. т. д. И предоставляет удобный редактор, благодаря чему можно легко получить множество высококачественных 3D моделей человека.

Еще 1 + MetaHuman для данной задачи в том, что у всех готовых персонажей 1 скелет, благодаря чему настроив анимацию на 1 персонаже её можно будет импортировать на всех остальных.

После импорта нескольких десятков моделей MetaHuman и настройки сцены была создана анимация при помощи Control Rig и Level Sequence. Level Sequence это класс в UE4, при помощи которого можно создавать анимацию персонажей по точкам (позам). В нем нужно на временной шкале задать позы персонажу, а Level Sequence превратит их в анимацию. Control Rig позволяет удобно менять позу персонажа.

Я старался подготовить анимацию, включающую все возможные повороты головы человека и выдержать равномерность всех углов.

В качестве фона был выбран HDRI – фон с освещением, накладываемый на объект, например куб или сферу. Я скачал несколько десятков HDRI с разрешением 4к-30к.

Дальше я настроил смещение камеры на 30^0 по углу рыскания после конца анимации и запуск её с начала. Когда камера делала полный оборот по углу рыскания, я делал шаг на 15 градусов по углу тангажа. Когда суммарный добавленный угол тангажа превышал 60^0 я смещал камеру на -75^0 и далее на
-15^0 до тех пор, пока суммарный добавленный угол тангажа не превышал
-45^0. Далее менялся персонаж с фоном и всё начиналось сначала.

Камера снимала персонажа 20 раз в секунду, пока тот проигрывал анимацию.

После реализации сбора набора данных для определения поворота головы появилась задача на определение углов поворота взгляда. Для её решения я подготовил новую анимацию, но для глаз. Персонаж не двигался, а только смотрел в разные стороны. Анимация делалась тем же способом что и предыдущая. Камера перемещалась $\pm 45^0$ по углу рыскания и тангажа с шагом 15^0. Позиция (0; 0; 0) это положение камеры напротив лица.

Далее я подумал, а почему бы не делать сбор данных о повороте головы и взгляда одновременно, поэтому я переделал анимацию персонажа для поворота головы, добавив в неё поворот глаз.

Чтобы получить точные углы поворота головы относительно камеры я добавил в камеру дочерний невидимый объект. Перед съёмкой дочернему объекту камеры задавалось мировые координаты и вращение равными таковым у кости головы, а в качестве итогового поворота брались углы поворота дочернего объекта камеры относительно самой камеры (в UE4 есть функция, возвращающая эти данные).

Алгоритма подготовки данных для обучения см. рис. 4.

В процессе вышеупомянутых действий было получено множество размеченных снимков разных персонажей с разными углами поворота головы и взгляда на разных фонах, далее нужно было из фотографии 1920*1080 вырезать голову (глаза) чтобы подавать только нужные данные на вход модели.

Так как в автоматизированной системе допуска в организацию используется детектор лиц Retinaface, им и было решено обрезать вотографии.
Pytorch_Retinaface https://github.com/biubug6/Pytorch_Retinaface это детектор лиц, который возвращает координаты левого верхнего и нижнего правого угла лица на фотографии и ряд ключевых точек, включая координаты глаз.
После нахождения верхнего левого и правого нижнего угла лица фотографии их координаты пересчитывались так, чтобы увеличить вдвое ширину обрезки по x, на 1.5 вниз и на 1.35 вверх по y, оставляя центр обрезки. Но если увеличение обрезки приводило к выходу за границы фотографии, то новая обрезка не выходила за её пределы.
Для обрезки глаз высчитывалось расстояние по x и y между точками и считались координаты обрезки по формулам:
Левая граница = int(max(0,пиксель расположения глаза по x - 0.4*(расстояние между глазами по x)))

Правая граница = int(min (Размер картинки по x, пиксель расположения глаза по x + 0.4*(расстояние между глазами по x)))

Граница сверху = int(max(0, пиксель расположения глаза по y - 0.2*(расстояние между глазами по x) - 0.2*(расстояние между глазами по y)))

Граница снизу = int(min(Размер картинки по y, пиксель расположения глаза по y + 0.2*(расстояние между глазами по x) + 0.2*(расстояние между глазами по y)))

Пример обрезки для модели см. рис. 5

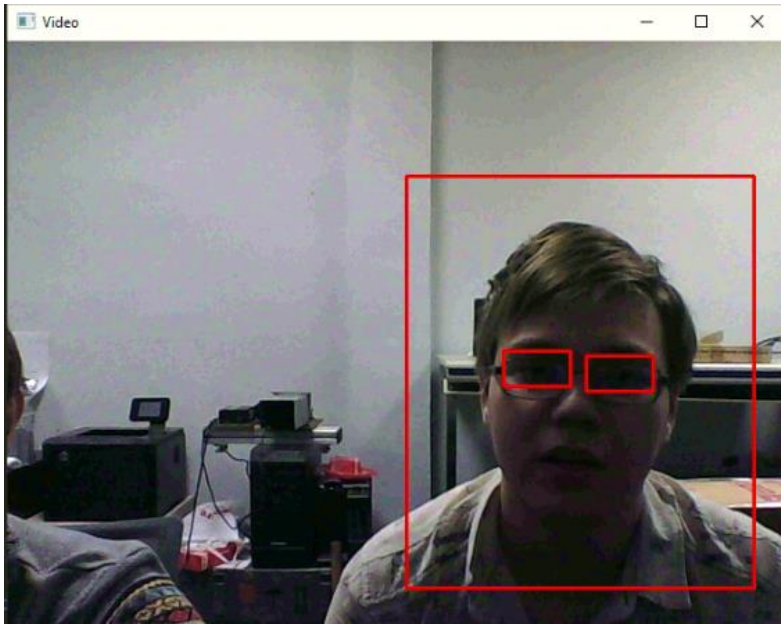


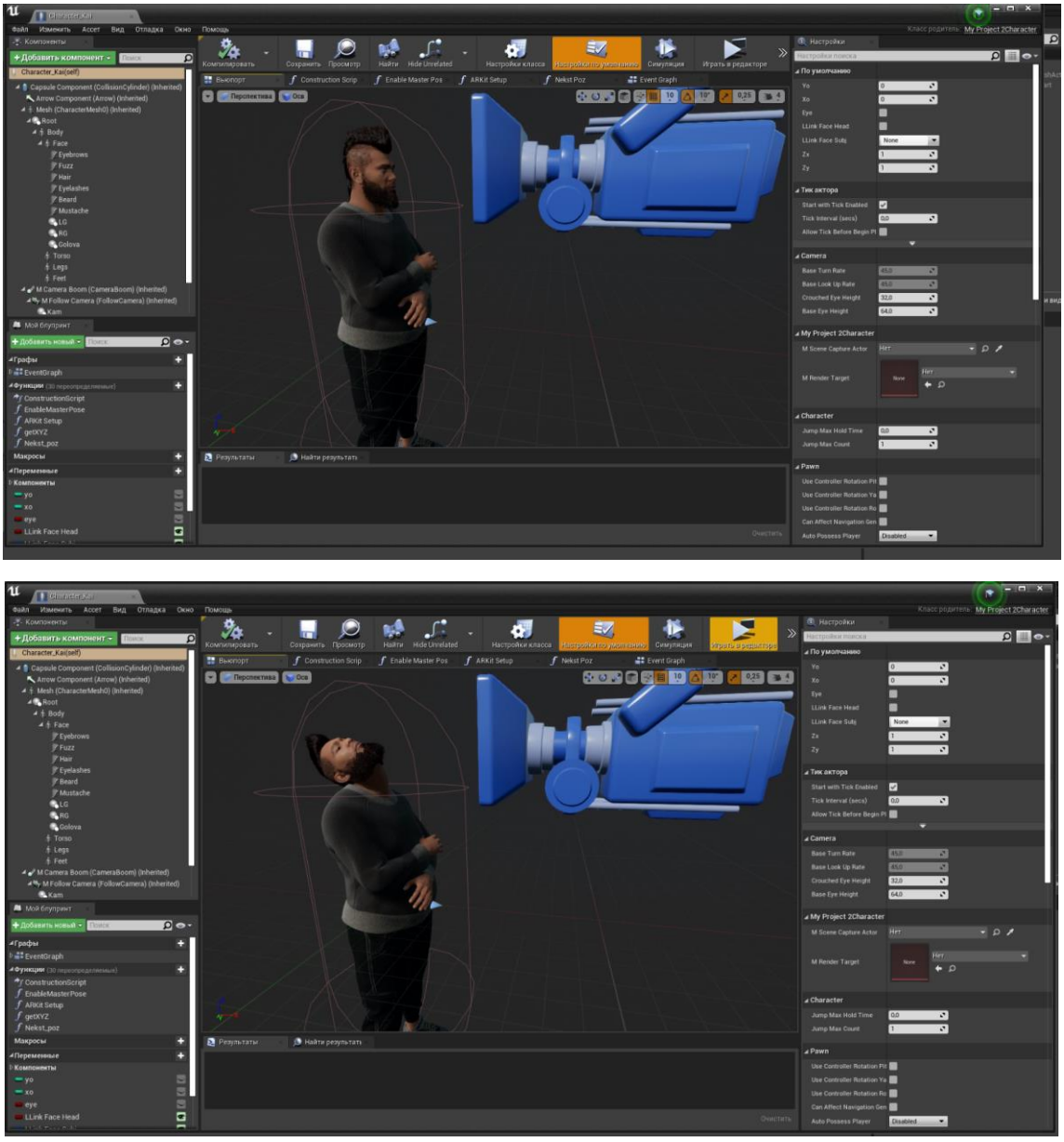
Рис 5 – Пример обрезки для модели.

Так как Retinaface это детектор лица, он не может видеть весь диапазон углов рыскания. Поэтому фотографии, на которых Retinaface не находил лица поступали на детектор голов YOLOV3 https://github.com/qqwweee/keras-yolo3. YOLOV3 находит только головы, поэтому данные о глазах на обрезанных фотографиях этим детектором терялись.

К сожалению, Retinaface недостаточно хорошо справляется с задачей нахождения координат глаз, поэтому соответствующий набор данных пришлось обрабатывать. Для этого была обучена модель на классификацию есть глаз / нет на фотографии. Я вручную прошёлся по кропам Retinaface 1 персонажа и фотографии, где невидно глаза поместил в 1 папку, а где видно в другую. Взял все ту же модель что и для определения углов, с теми же гиперпараметрами но после EfficientNet-B0 следовал линейный слой (1000:2). train:val разбил как 80:20. Сеть обучилась к 462 эпохе с точностью 0.9437. И она прекрасно дообрабатывала данные.

В конце набор данных для определения направления взгляда был обрезан так, чтобы диапазон угла рыскания составлял $\pm 99^0$, а тангажа и крена $\pm 54^0$. Набор данных для определения направления головы был обрезан так, чтобы диапазон углов тангажа и крена составлял $\pm 90^0$.

Скриншоты работы с MetaHuman в Unreal Engine 4 см. рис. 6.



Скриншоты работы с MetaHuman в Unreal Engine 4

Пример готовых изображений из Unreal Engine 4 см. рис. 7.

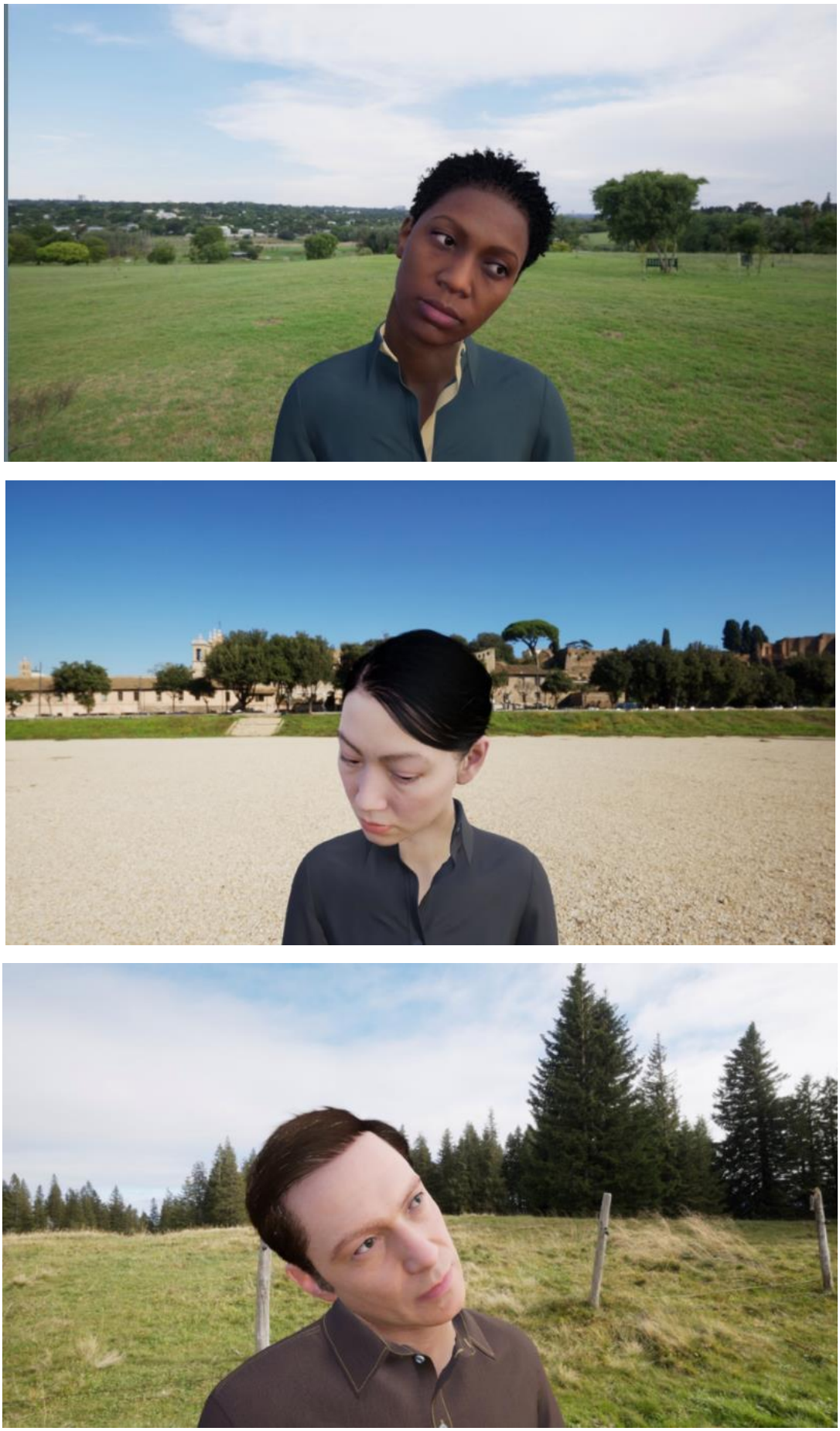


Рис. 7 – Пример готовых изображений из Unreal Engine 4

2.3.4 Синтетический набора данных

Датосет был собран на 13 уникальных персонажах MetaHuman. Из них 10 распределены в train и 3 в val.

Распределение углов в синтетическом наборе данных для определения углов поворота головы см. табл. 5

train	val
Mx = -0.3852056792859749 My = -0.5310587573011395 Mz = 0.9794597019938566 Dx = 10179.016679316705 Dy = 1207.8221065653856 Dz = 1277.6355470136584 N = 47155	Mx = -0.09467975948021413 My = -0.6170029512430701 Mz = 1.359568933585212 Dx = 10211.931531395214 Dy = 1211.8157432322407 Dz = 1320.273519829442 N = 16150

Табл. 5 – Распределение углов в синтетическом наборе данных для определения углов поворота головы

Хоть набор данных и куда меньше, чем в WHENet, но он гораздо качественнее, мат ожидание углов почти равна 0 и выдержана равномерность. К тому же увеличить набор данных не составит особого труда.
Распределение углов синтетического набора данных для определения углов поворота взгляда см. табл. 6.

train	val
Mx = 0.4887251151686053 My = 1.4264349895346098 Mz = -1.2706818114915006 Dx = 1635.3424309571312 Dy = 622.7653265642633 Dz = 401.93736538770685 N = 42600	Mx = 0.14890681152294674 My = 1.4750082431063696 Mz = -0.48385297121360266 Dx = 1608.2461255589944 Dy = 626.4363492388663 Dz = 419.81112452634795 N = 14120

Табл. 6 – Распределение углов синтетического набора данных для определения углов поворота взгляда

2.3.5 Объединение синтетического набора данных с 300W_LP

300W_LP (название набора данных) содержит 4 папки с уникальными наборами людей и 4 папки с отражением угла рыскания. Я разбил этот набор данных на train (название папок в наборе данных 300W_LP: IBUG, IBUG_Flip, AFW, AFW_Flip, HELEN, HELE N_Flip) 82372 изображений и val (LFPW, LFPW_Flip) 33111 изображений. Train из этого набора данных был объединён с train из синтетического набора данных 47155 и val с валом из синтетического набора данных 16150
Распределение углов в синтетическом наборе данных с 300W_LP, для определения углов поворота головы см. табл. 7.

train	val
Mx = -0.16108573393231138 My = -5.757936505309634 Mz = 0.30610803637975326 Dx = 5621.484366700841 Dy = 566.1090027274803 Dz = 559.8991564210407 N = 129527	Mx = -0.02369725951686617 My = -6.310886788520364 Mz = 0.4845329361132907 Dx = 5359.959049529323 Dy = 486.89974449264497 Dz = 510.4174560062028 N = 49261

Табл. 7 – Распределение углов в синтетическом наборе данных с 300W_LP, для определения углов поворота головы

Очевидно, что набор данных получился не сбалансированный. Улов в диапазоне (-3;0) 1665 а (168;171) 193. Но по крайней мере присутствуют все диапазоны углов и встречаются большие углы по всем 3м осям. А дисбаланс классов можно решить при помощи Sampler.

2.3.6 Sampler

Sampler определяет стратегию извлечения выборок из набора данных.
Я использую sampler по вероятности нахождения касса угла на train. Это делается для того, чтобы вероятность всех диапазонов углов была одинакова при чтении новой фотографии. Для этого я создаю txt файл, в котором каждая строчка равна количеству всех фотографий / количество фотографий, принадлежащих классу угла рыскания на этой же строчке в описании набора данных. Sampler внутри себя использует Softmax для преобразования полученных чисел в вероятности для каждого класса.

$$s\left(x_i\right) = \frac{e^{x_i}}{\sum_{j=1}^{n} e^{x_j}}$$

2.4 Обоснование и разработка программного обеспечения, реализующего алгоритм решения поставленной задачи

2.4.1 Обоснование разработки программного обеспечения, реализующего алгоритм решения поставленной задачи

Выше была описана процедура обучения нейронных сетей, для решения задачи определения углов поворота головы и направления взгляда. Однако сложно судить о проделанной работе по одной лишь нейросети. Поэтому было разработаны 3 программы. 1я получает на вход картинку, обрабатывает её при помощи Retinaface и отрисовывает углы поворота головы и направления взгляда на этой фотографии см. рис. 8. 2я получает на вход видео, обрабатывает его при помощи Retinaface и записывает новое, но с отрисованными углами поворота головы и направления взгляда см. рис. 9. 3я берет изображения с камеры и выводит его в формате видео в прямом эфире, позволяя лично убедится в состоятельности модели см. рис. 10. Пример результата работы программ см. рис. 11.

2.4.2 Алгоритм решения поставленной задачи с исходными данными в виде видеопотока с камеры

Рис. 8 – Алгоритм решения поставленной задачи с исходными данными в виде видеопотока с камеры

2.4.3 Алгоритм решения поставленной задачи с исходными данными в виде видео

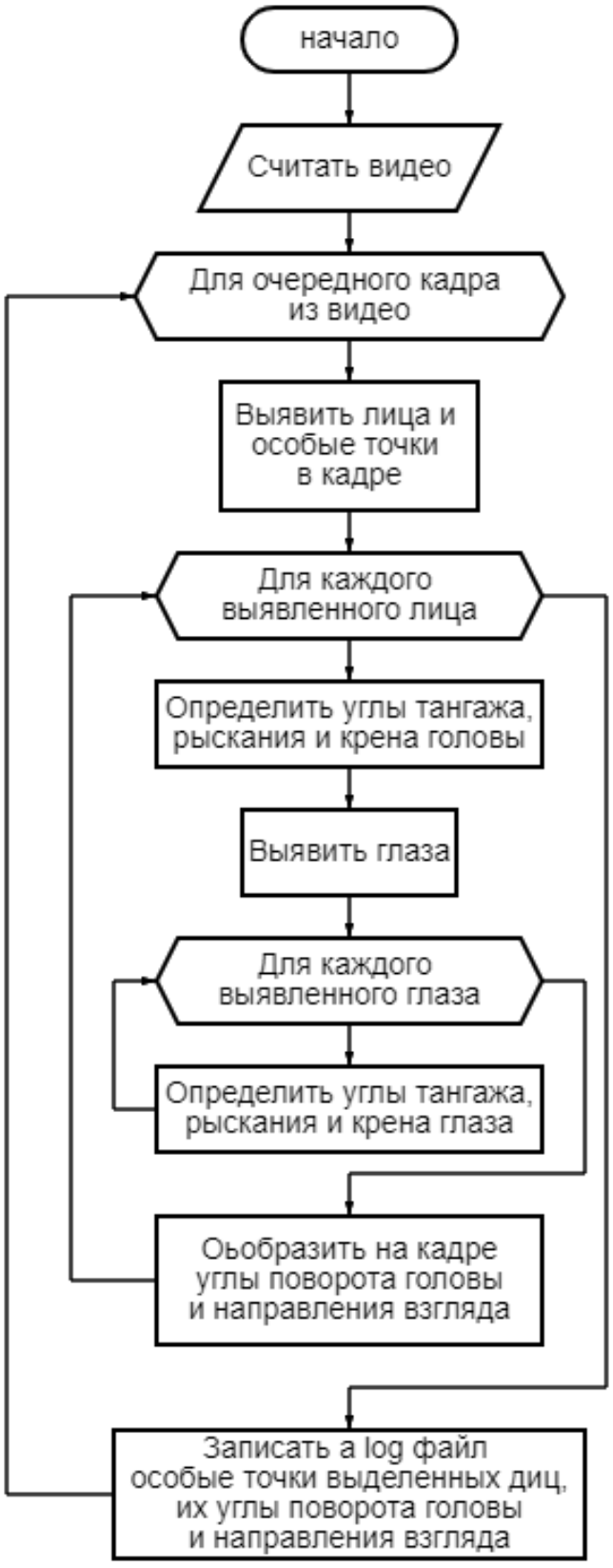


Рис. 9 – Алгоритм решения поставленной задачи с исходными данными в виде видео

2.4.4 Алгоритм решения поставленной задачи с исходными данными в виде фотографии

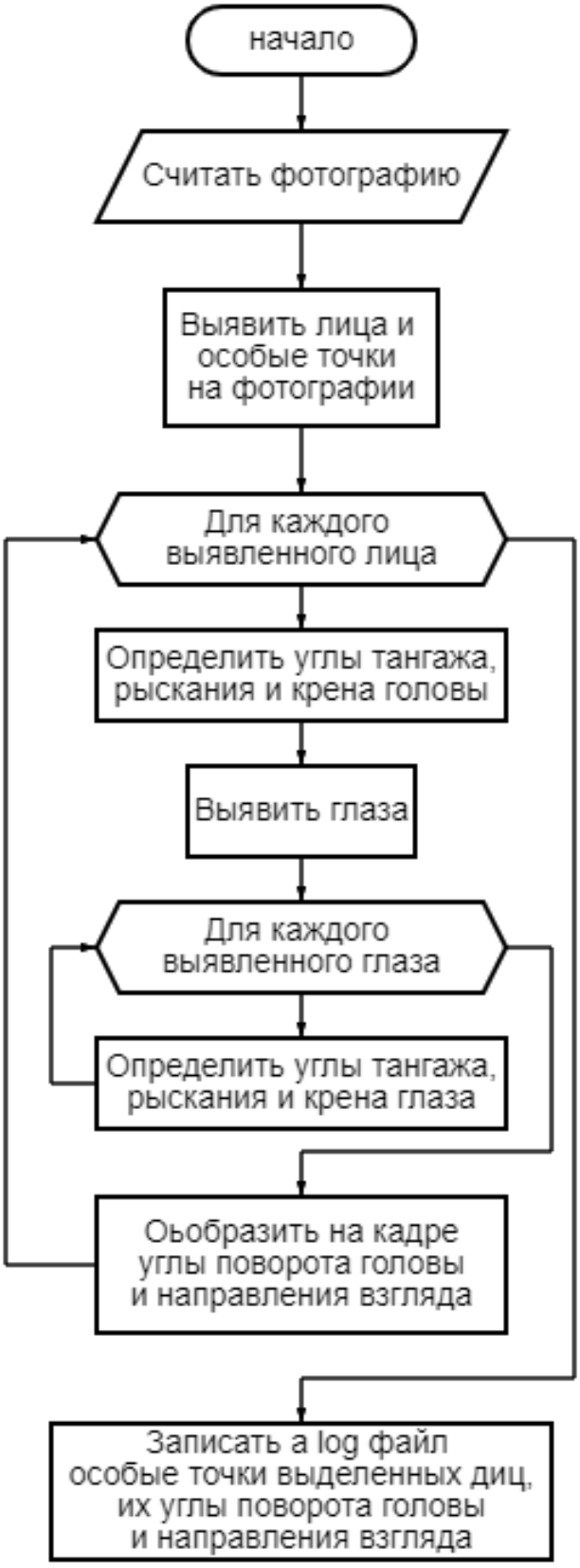


Рис. 10 – Алгоритм решения поставленной задачи с исходными данными в виде фотографии

2.4.5 Пример результата работы программ

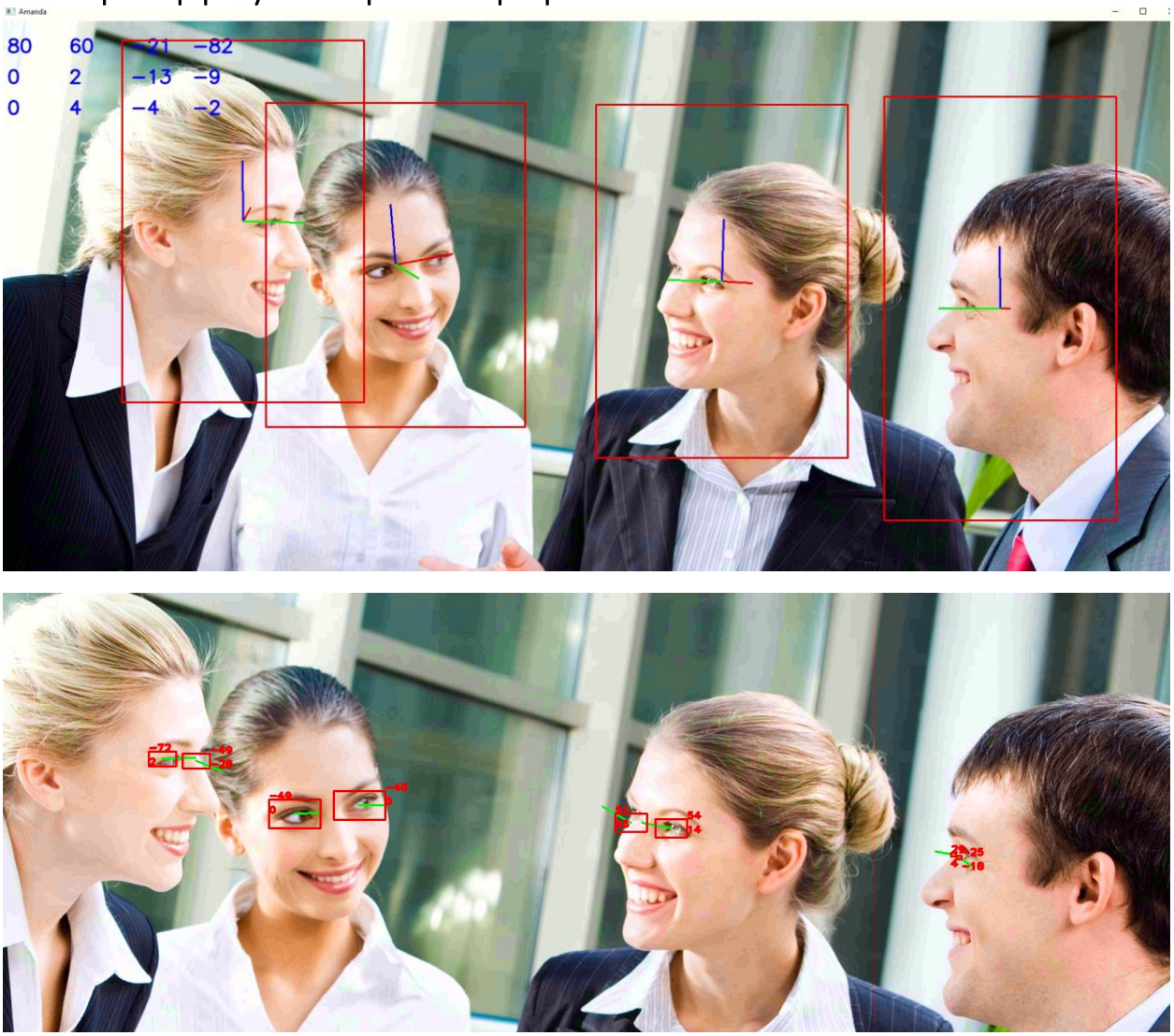


Рис. 11 – пример результата работы программ

3. Технологическая часть

3.1 Содержание и формы подготовки исходной информации для решаемой задачи

Исходная информация представлена в виде изображения в формате RGB см. рис. 12.

Рис. 12 – Пример исходной информации

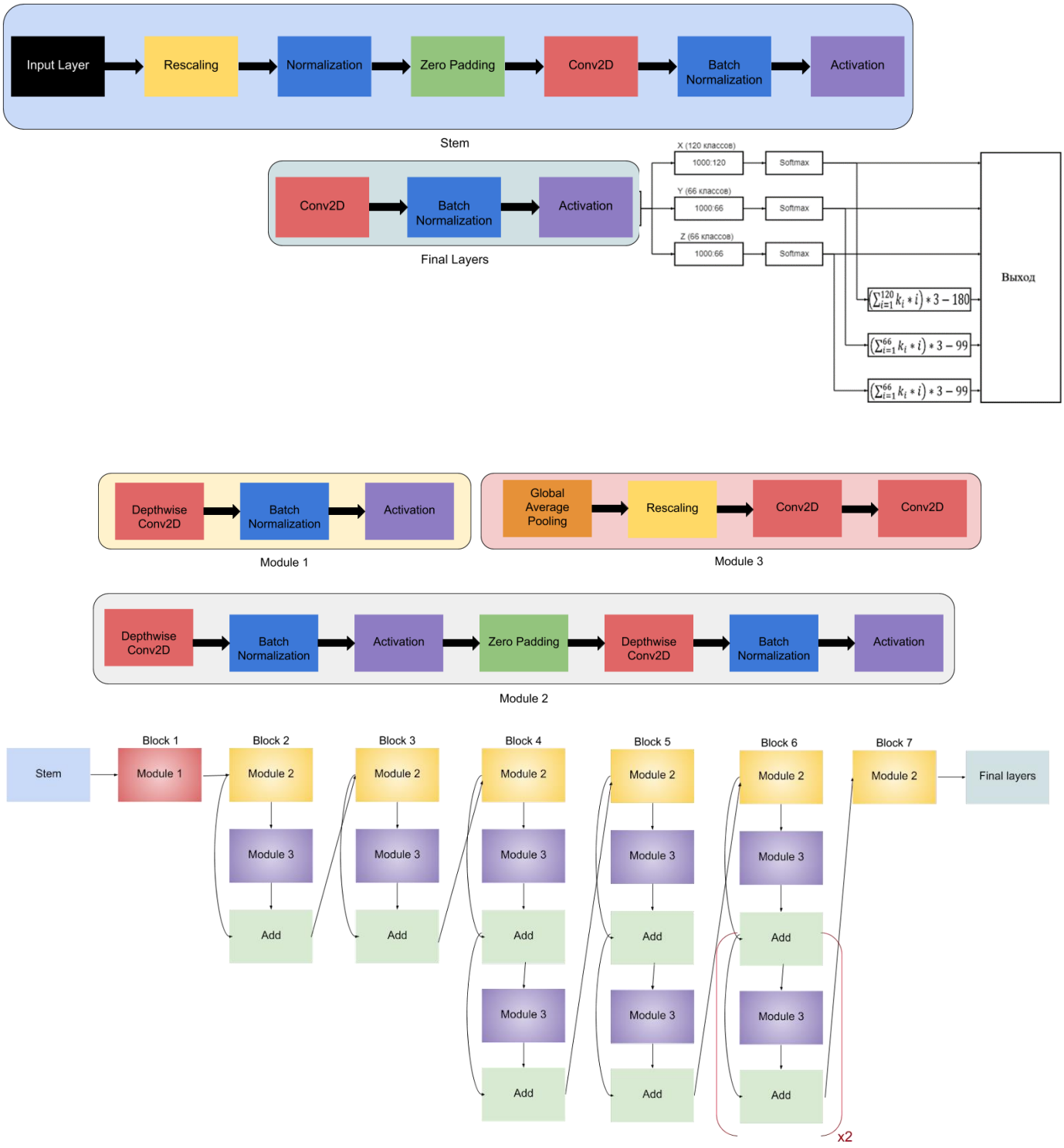
3.2 Участие человека в процессе решения задачи

Пользователю достаточно запустить скрипт. Определение углов поворота головы и направление взгляда полностью автоматизировано. Однако полученный алгоритм можно использовать в качестве доказательства “неправильного” внимания. Например, систему можно установить в Госдуму и выделять депутатов, которые больше 5 минут не смотрят на сцену. Человек сразу заметит выделенного красным прямоугольником спящего депутата и попросит коллег разбудить его. В киберспорте можно отслеживать направление взгляда спортсмена во время игры и, если он убивает противника, не взглянув на него, можно будит дисквалифицировать спортсмена за мошенничество. В этой ситуации человек должен посмотреть на запись предполагаемого нарушения и при помощи модели доказать/опровергнуть обвинение.

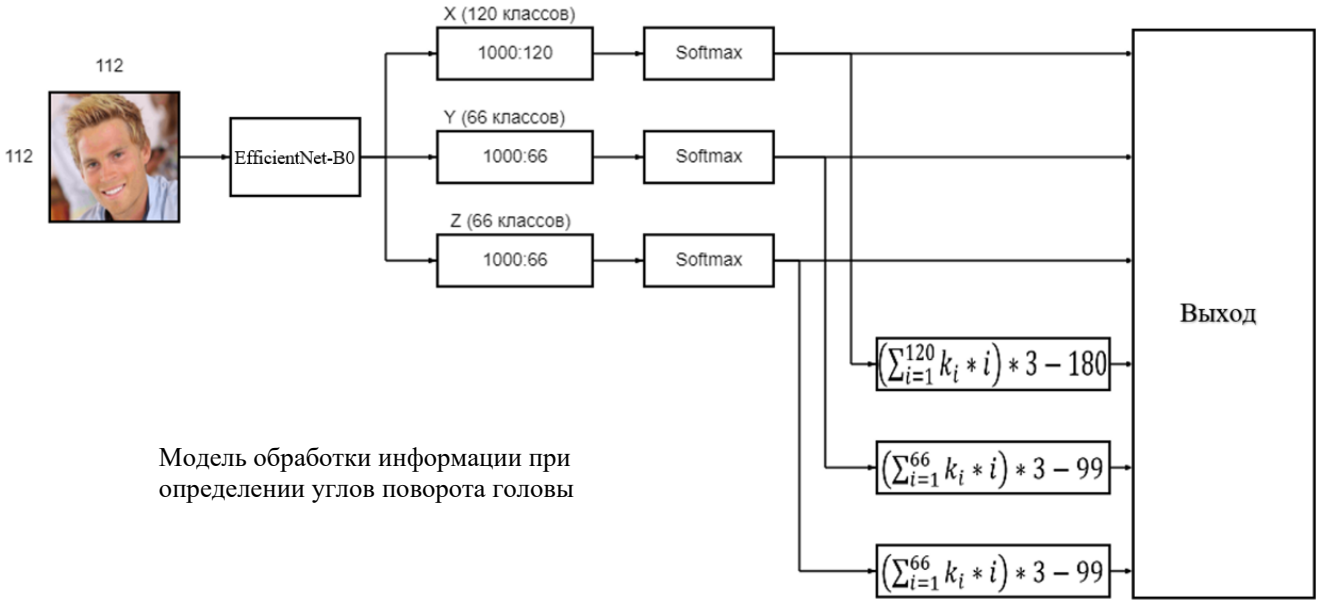
3.3 Технические средства, обеспечивающие технологический процесс обработки информации

Системы для функционирования необходима камера, снимающая видео в формате RGB и компьютер с необходимым программным обеспечением. Желательно наличие видеокарты nvidia.

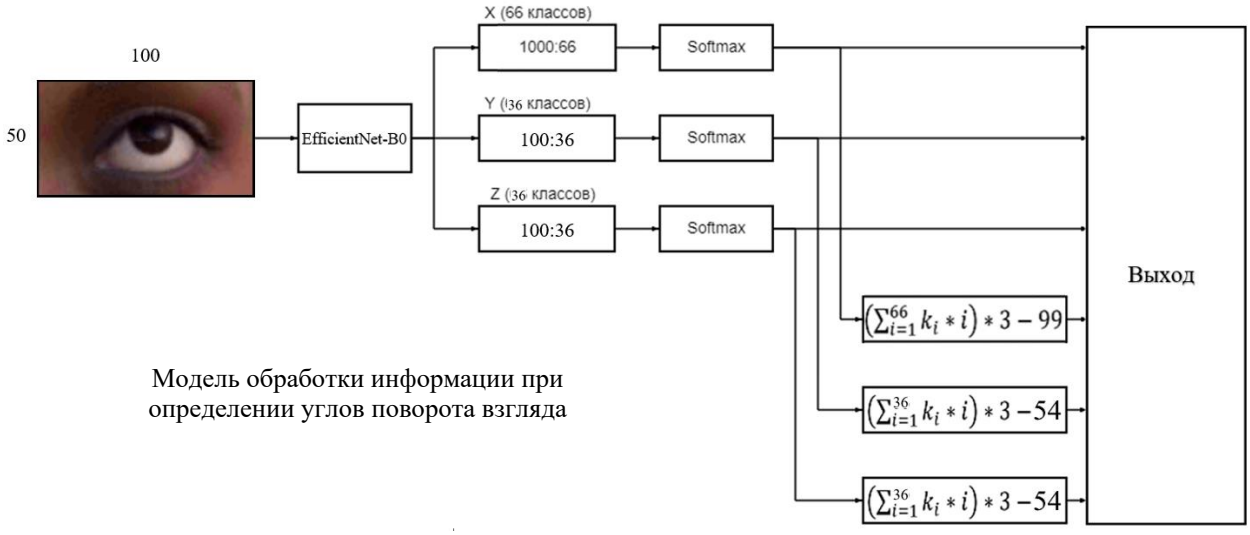
Графический материал



					Дипломный проект			
					Структура нейросети	Лист	Масса	Масштаб
Изм.	Лист	№ докум.	Подп	Дата				
Разраб.		Гордеев Н. М.						
Руков.		Ескин В.И.				Лист 1	Листов 5	
Конс.								
						МАИ (Технический университет)		
						Кафедра 304		
Утв.						Группа М30-307Б-18		

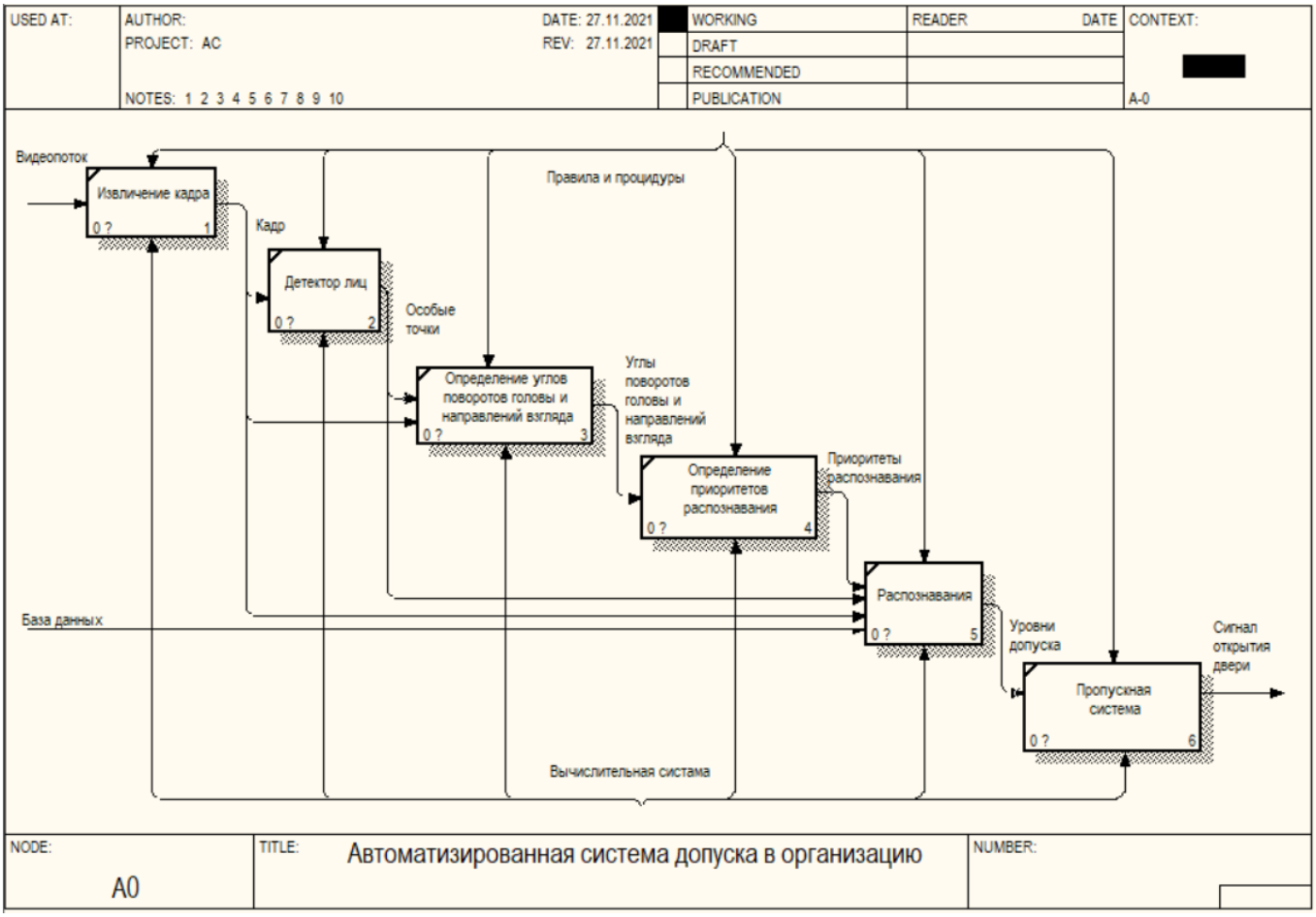


Модель обработки информации при определении углов поворота головы



Модель обработки информации при определении углов поворота взгляда

					Дипломный проект			
Изм.	Лист	№ докум.	Подп	Дата	Модель обработки информации	Лист	Масса	Масштаб
Разраб.		Гордеев Н. М.						
Руков.		Ескин В.И.				Лист 2	Листов 5	
Конс.						МАИ (Технический университет) Кафедра 304 Группа М30-307Б-18		
Утв.								



					Дипломный проект		
Изм.	Лист	№ докум.	Подп	Дата	Структура организационной структуры АС	Лист	Масса / Масштаб
Разраб.		Гордеев Н. М.					
Руков.		Ескин В.И.				Лист 3	Листов 5
Конс.						МАИ (Технический университет) Кафедра 304 Группа М30-307Б-18	
Утв.							

					Дипломный проект			
						Лист	Масса	Масштаб
Изм.	Лист	№ докум.	Подп	Дата	Постановка и формализация задачи			
Разраб.		Гордеев Н. М.				Лист 4	Листов 5	
Руков.		Ескин В.И.				МАИ (Технический университет) Кафедра 304 Группа М30-307Б-18		
Конс.								
Утв.								

					Дипломный проект			
Изм.	Лист	№ докум.	Подп	Дата	Структура ПО автоматизированной системы	Лист	Масса	Масштаб
Разраб.		Гордеев Н. М.						
Руков.		Ескин В.И.				Лист 5	Листов 5	
Конс.						МАИ (Технический университет) Кафедра 304 Группа М30-307Б-18		
Утв.								