Федеральное государственное бюджетное образовательное учреждение
высшего профессионального образования
Московский авиационный институт
(национальный исследовательский университет)

Пушилин Сергей Владимирович

ПРОГРАММНО-АЛГОРИТМИЧЕСКИЙ КОМПЛЕКС ДЛЯ БЕСКОНТАКТНОГО ВИЗУАЛЬНОГО УПРАВЛЕНИЯ ПОДВИЖНЫМ ОБЪЕКТОМ

Специальность 05.13.01 – Системный анализ, управление и обработка информации (приборостроение, биотехнические системы и технологии)

диссертация на соискание ученой степени
кандидата технических наук

Москва - 2015

ОГЛАВЛЕНИЕ

ВВЕДЕНИЕ.

В последнее время получили значительное развития технологии «бесконтактного управления». Суть их заключается в том, что они позволяют человеку-оператору взаимодействовать с какой-либо технической системой не при помощи традиционных органов управления, а при помощи системы сигналов, основанной на жестах, направлении взгляда, голосовых командах и т.д., не вступая в физический контакт.

С аппаратной точки зрения, система бесконтактного управления представляет собой считывающее устройство, регистрирующее какие-либо действия оператора, и вычислительное устройство, которое интерпретирует полученную информацию в управляющие сигналы по заложенному в него алгоритму.

Преимущество бесконтактного управления в том, что оно позволяет осуществляться в тех случаях, когда применение традиционных органов управления затруднено или вообще невозможно. Кроме того, за счёт замены традиционных органов управления компактным считывателем, система бесконтактного управления занимает меньше места и даёт большую свободу передвижения оператору, нежели при традиционном подходе.

На данный момент существует ряд подобных технологий, отличающихся друг от друга по следующим основным параметрам:

1. Характер управляющих сигналов от оператора. В зависимости от реализации конкретной технологии, оператор может осуществлять управление при помощи жестов руками, движений всего тела, изменения направления взгляда, голосовых команд и т.д. Также, может применяться сочетание двух или более вышеприведённых способов.
2. Считывающее устройство. Как правило, интерпретация команд от оператора происходит за счёт распознавания его изображения. Это изображение может быть получено как при помощи обычной цифровой камеры в видимом спектре, так и при помощи инфракрасной камеры. Также в некоторых системах применяются стереоизображения,

полученные при помощи нескольких камер, разнесённых на небольшое расстояние.

3. Расстояние от оператора до считывающего устройства. В различных системах существуют собственные требования к местоположению оператора по отношению к считывающему устройству. Оператор может как свободно перемещаться перед ним, так и быть вынужденным находиться в непосредственной близости от считывающего устройства. Считывающее устройство может даже быть закреплено непосредственно на теле оператора, например, как в системе прицеливания, встроенной в шлем пилота.
4. Предназначение. До недавнего времени технологии бесконтактного управления не были так широко распространены и использовались преимущественно в военных целях. Сейчас же подобные технологии общедоступны и могут использоваться в мультимедийных системах, как компьютерные интерфейсы, а также в медицине.
5. Точность позиционирования. В зависимости от решаемых задач, к различным системам бесконтактного управления выдвигаются различные требования к точности определения координат отслеживаемых объектов, таких как кисти рук или зрачки глаз оператора. В некоторых случаях достаточно только определить общий характер движений, например, взмах рукой или моргание глазами, в то время как для задачи управления подвижным объектом требуется получить координаты отслеживаемых объектов с точностью, достаточной для осуществления успешного управления.

В данной работе рассматривается система бесконтактного управления подвижным объектом, основанная на определении направления взгляда оператора по видеоизображению, а также различные алгоритмы и методы обработки информации, используемые ей во время работы.

Научная новизна работы.

1. Предложена двухканальная архитектура системы бесконтактного управления.

2. Разработан алгоритм работы системы бесконтактного управления, включающий в себя алгоритмы определения углового положения головы оператора, определения направления взгляда оператора и комплексирования данных от разных каналов.

3. Предложен способ экспериментальной оценки эффективности работы системы бесконтактного управления, разработаны алгоритмы проведения тестирования данной системы.

4. Получены экспериментальные данные по работе системы, сделаны выводы о практической эффективности предложенных решений.

Практическая значимость исследования состоит в том, что полученный алгоритм работы системы может быть использован для разработки действующего образца системы бесконтактного управления, которую можно применить в самых различных областях, от медицины до управления транспортным средством.

Достоверность результатов подтверждена серией экспериментов с использованием экспериментальной реализации системы на основе разработанных алгоритмов. Для их проведения был разработан испытательно-демонстрационный стенд. Результаты данных экспериментов приведены в главе 4.

Структурно работа поделена на четыре части (главы):

В **главе 1** проводится анализ существующих технологий бесконтактного управления. На основе этого анализа предлагается принцип работы системы бесконтактного управления, формируются основные требования к её работе, а также к аппаратному обеспечению. Кроме того, проводится оценка максимальных характеристик работы системы, таких как точность, быстродействие, максимальное расстояние оператора от считывающего устройства и т. д., с учётом выбранного принципа работы системы и аппаратного обеспечения.

В **главе 2** рассматривается метод обработки видеоизображения, используемый для определения направления взгляда, а именно алгоритм поиска объектов на изображении. Проводится анализ существующих алгоритмов распознавания образов, и из них выбирается наиболее подходящий для решения задачи бесконтактного управления.

В **главе 3** приводится алгоритм определения направления взгляда и формирования управляющего сигнала, разработанный в рамках диссертационной работы для системы бесконтактного управления.

В **главе 4** показаны результаты вычислительного эксперимента, проводимого при помощи опытного образца системы, использующего в работе алгоритм, изложенный в главе 3. Суть эксперимента заключалась в определении основных характеристик полученного образца, а также проанализировать влияние на процесс бесконтактного управления непроизвольных движений глаз человека, так называемых «саккад» (подробнее в главе 4). На основе полученных результатов предлагается способ компенсации вносимых ими помех в работу алгоритма определения направления взгляда.

ГЛАВА 1. АНАЛИЗ ВОЗМОЖНЫХ РЕШЕНИЙ ЗАДАЧИ И ФОРМИРОВАНИЕ ТРЕБОВАНИЙ К СИСТЕМЕ.

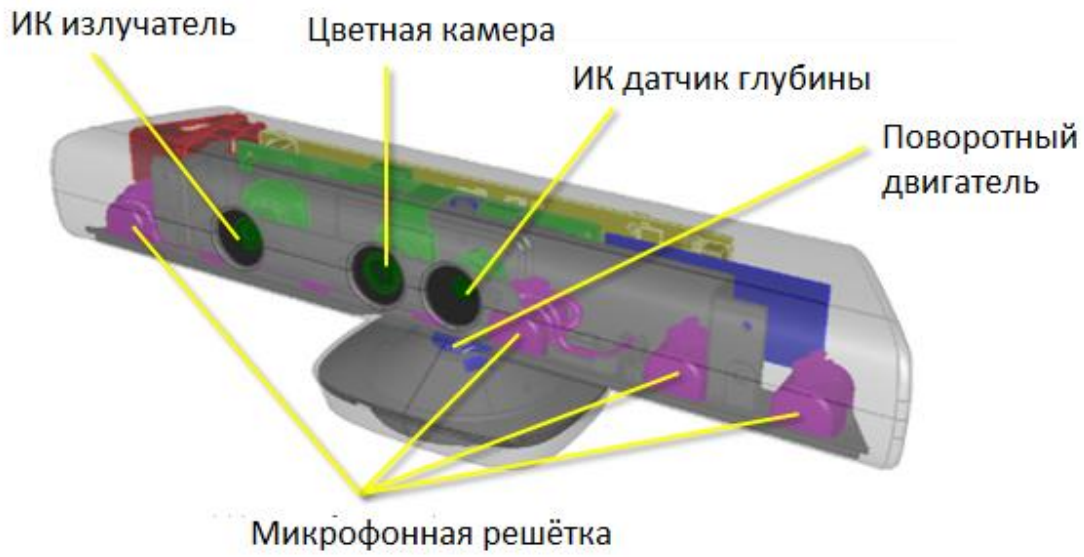
1.1. Обзор существующих систем бесконтактного управления.

На данный момент существует множество различных устройств и систем, так или иначе использующих принцип бесконтактного управления. Ниже приведены одни из самых известных представителей подобного рода продуктов.

1.1.1 Microsoft Kinect.

Бесконтактный сенсорный игровой контроллер от компании Microsoft. Позволяет пользователю взаимодействовать с программным обеспечением через устные команды и изменение позы тела. Основан на разработке израильской компании PrimeSense.

Аппаратное обеспечение состоит из двух сенсоров глубины, цветной видеокамеры и микрофонной решетки, заключённых в едином корпусе (Рис. 1).



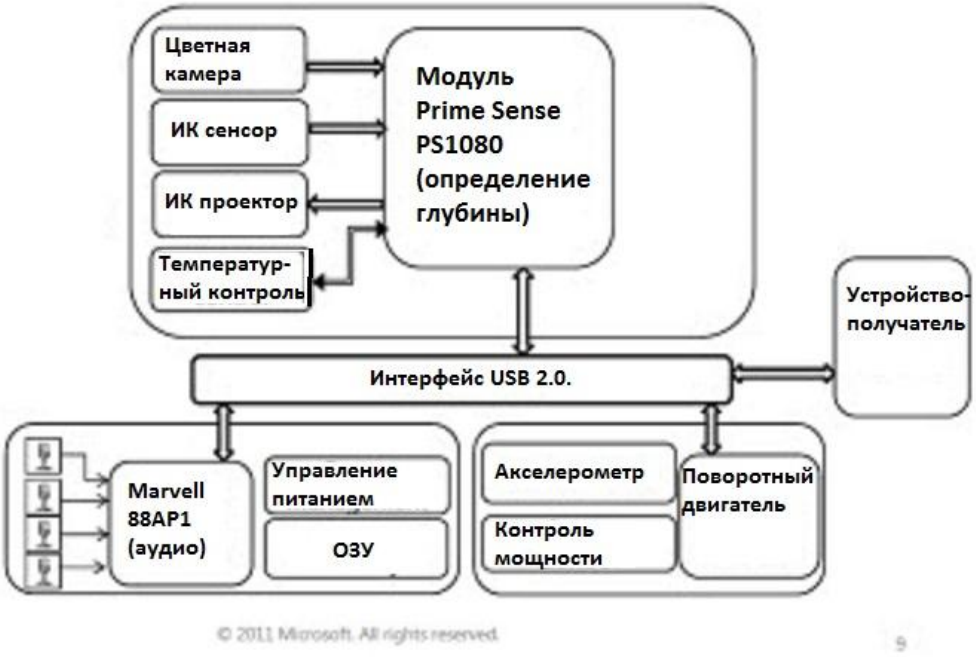


Рис.1. Сенсор Microsoft Kinect.

1.1.2 TrackIR.

TrackIR – система отслеживания движений головы пользователя, основанная на применении инфракрасной камеры с подсветкой инфракрасными диодами. Также для её работы необходима отражающая наклейка, закреплённая на голове пользователя.

Система позволяет отследить движение по шести степеням свободы. Также, в некоторых комплектациях, в качестве маркера используется набор из трёх инфракрасных светодиодов, который крепится к голове пользователя (Рис.2).

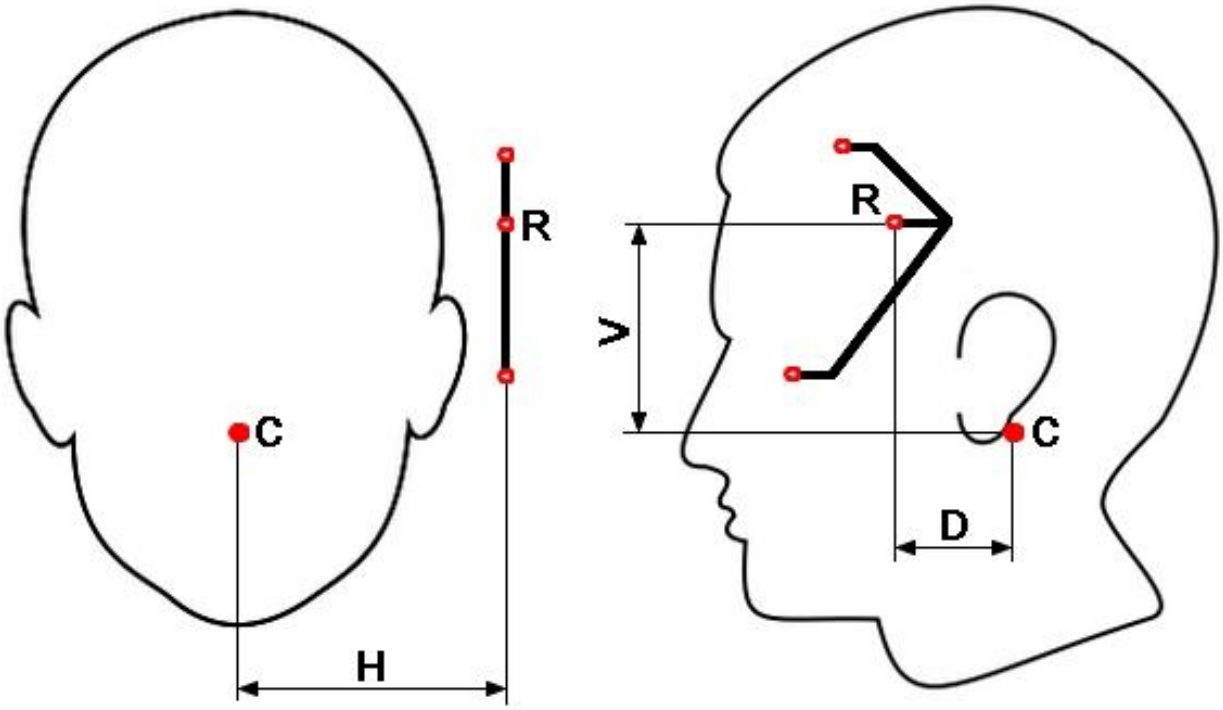


Рис. 2. Сенсор TrackIR.

1.1.3. Трекер RUCAP UM-5.

это беспроводной джойстик для управления обзором в компьютерных играх и манипуляции мышью без использования рук. Определяет угол поворота головы оператора и положение головы относительно монитора. Работа трекера основана на применении активного ультразвукового маркера, закрепляемого на голове оператора и четырёх ультразвуковых приёмников (Рис.3).

Рис.3. Трекер RUCAP UM-5.

1.1.4. Профессиональный трекинг A.R.T.

A.R.T. — профессиональная система отслеживания, применяемая в научных исследованиях, системах виртуальной реальности для разработки сложных технических изделий (авиакосмическая и автомобильная промышленность и т. д.), медицине и робототехнике. Система построена на инфракрасном оптическом принципе.

1.1.5. Системы прицеливания, встроенные в шлем пилота.

Данные системы предназначены для пилотов боевой авиации и позволяют осуществлять целеуказание бортового вооружения при помощи движений головы и глаз лётчика. Такая система, например, будет встроена в шлем пилота F-35C Lightning II (Рис. 4). Её разработкой занимаются компании Vision Systems International и Helmet Integrated Systems Limited из Великобритании. Кроме того, существуют аналогичные отечественные разработки.

Рис. 4. Шлем пилота самолёта F-35C со встроенной системой прицеливания.

1.1.6. Решения, использующие только веб-камеру.

В основе данных решений лежит компьютерная программа, использующая изображение с обычной веб-камеры без каких-либо дополнительных аппаратных средств. В качестве примеров можно привести программы Enable Viacam, FaceTrackNoIR и Cam2Pan. Все они осуществляют отслеживание головы оператора.

Помимо очевидного преимущества, заключающегося в отсутствии дополнительных сенсоров, данные решения обладают рядом недостатков, в частности, низкой скоростью обработки изображения и большой долей ошибок распознавания.

1.2. Структурная схема системы бесконтактного управления.

1.2.1 Выбор способа управления.

Как видно из обзора существующих решений, большинство из существующих технологий бесконтактного управления используют маркер, закрепляемый на голове оператора. Кроме того, в качестве сенсора в них используется нестандартные устройства, например, инфракрасная камера или даже ультразвуковой приёмник.

Такой подход позволяет упростить отслеживание движений оператора, но в то же время вносит такие дополнительные сложности в работе с системой, как

необходимость носить закрепляемый на голове или теле маркер, а также потребность в специализированном сенсоре. Чтобы их избежать, в системе бесконтактного управления, рассматриваемой в данной диссертационной работе, в качестве сенсора используется типовая цифровая видеокамера. В качестве источника управляющего воздействия предлагается использовать направление взгляда оператора.

Видеоизображение с камеры обрабатывается вычислителем, в качестве которого могут выступать различные устройства, от персонального компьютера до мобильных и встраиваемых устройств.

Среди рассмотренных существующих технологий, позволяющих осуществлять управление при помощи изменения направления взгляда оператора, можно выделить два основных подхода к определению направления взгляда:

1. *По угловому положению головы оператора*. Основным преимуществом данного подхода является относительная простота детектирования лица оператора на изображении. В то же время, из-за неточного определения местоположения отдельных частей лица на изображении, а также различий в строении лица у разных людей возникает погрешность в определении углового положения головы. Погрешность можно сократить, проведя предварительную калибровку системы перед её использованием конкретным оператором, но это создаёт дополнительные эксплуатационные сложности.
2. *По положению зрачков глаз оператора*. Данный подход требует от оператора минимум действий, что значительно удобнее, в особенности, когда движения головы затруднены или невозможны. Кроме того, точность позиционирования при таком подходе будет заметно выше. К недостаткам этого подхода можно отнести то, что детектирование зрачков глаз на изображении является более сложной задачей, поэтому высока вероятность появления ошибок распознавания. Помимо этого, глаза человека регулярно совершают микродвижения (так называемые «саккады») [], что затрудняет управление таким способом.

С целью объединения достоинств описанных выше подходов, предлагается использовать их параллельно, используя двухканальную схему построения системы.

1.2.2. Двухканальная схема построения системы.

Предлагаемая структурная схема системы показана на рисунке 5. Подсистема определения углового положения головы оператора даёт «грубую» оценку направления взгляда, которая уточняется за счёт второго канала, включающего в себя подсистему определения направления взгляда. Объединение сигналов разных каналов происходит в подсистеме объединения и взаимной коррекции сигналов.

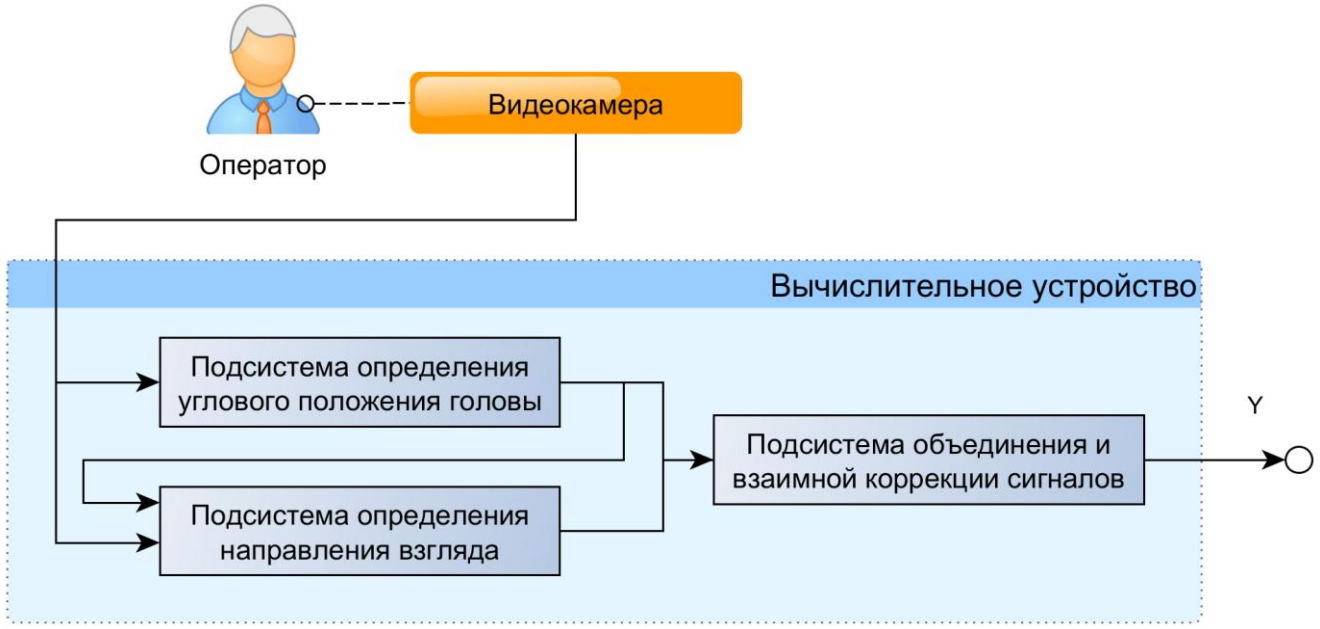


Рис. 5. Структурная схема системы.

Для упрощения работы подсистемы определения направления взгляда можно использовать данные об угловом положении головы, что также отображено на схеме.

1.2.3. Классификация системы как системы распознавания образов.

Рассматриваемая в данной работе система бесконтактного управления относится к системам распознавания образов (СРО) реального времени (РВ), которые, согласно статье [], классифицируются следующим образом (Рис. 6):

Рис.6. Классификация СРО РВ.

Принцип классификации по контролю времени распознавания образов разделяет СРО, на неконтролируемые по времени (т.е. не системы РВ и поэтому не отражены на рисунке 1), гибкие системы РВ и жёсткие системы РВ. Жёсткие СРО РВ характеризуются наличием жёстких сроков для каждого случая распознавания (в них обязательно необходимо укладываться). Гибкие СРО РВ отличаются тем, что нарушения сроков распознавания нежелательны, но допустимы. К неконтролируемым по времени системам относятся такие системы, которые проектируются для распознавания каких-либо объектов без конкретных сроков.

Рассматриваемая в данной диссертационной работе система относится к гибким СРО РВ, поскольку из-за разнородности входных видеоданных невозможно точно предопределить время обработки изображения, но, тем не менее, время обработки не может превышать допустимый максимум, определяемый разработчиком.

Если в качестве принципа классификации использовать влияние выходных результатов распознавания на управление системой, то СРО РВ можно разделить на системы без обратной связи и с обратной связью. Системы без обратной связи в самом простейшем случае можно описать в виде: $Y = f(X)$ (Рис. 7).

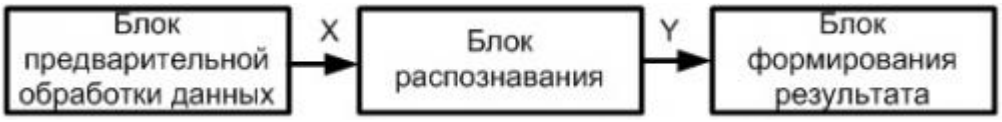


Рис. 7. СРО РВ без обратной связи

Блок распознавания принимает на вход данные в некотором формате X от блока предварительной обработки данных (к примеру, этот блок отвечает за создание изображений для распознавания, поступающих с видеокамеры), и выдает результат Y, который блоком формирования результата тем или иным способом доносится до пользователя либо до системы принятия решения. Примером таких систем без обратной связи являются системы детектирования движений, системы распознавания штриховых кодов, системы распознавания автомобильных номеров, системы неразрушающего контроля, основанные на распознавании некоторых, характеризующих объект признаков.

Системы с обратной связью используют данные о предыдущих результатах распознавания с целью настройки системы под конкретные образы и условия распознавания. Работа таких систем может описываться выражениями:

$Yt = f(Xt, Yt-1)$, $Xt = f(Xt, Yt-1)$, где t – текущий момент распознавания.

В данных системах результаты распознавания могут влиять не только на параметры работы функции распознавания, но и на предварительную обработку и формирование данных (Рис. 8).

Рис. 8. СРО РВ с обратной связью.

Примером систем с обратной связью являются системы оптического трекинга объектов, где предыдущее расположение объектов влияет на определение текущего местоположения, системы распознавания в подвижных роботах. Кроме того, многие системы без обратной связи могут быть модернизированы за счёт обработки результата распознавания (например, адаптивная система распознавания штриховых кодов).

Принцип классификации по автономности разделяет СРО РВ в соответствии со стандартом POSIX.13-2003 []:

• минимальные системы РВ;

• управляющие системы РВ;

• специализированные системы РВ;

• универсальные системы РВ.

Отличия данных систем обусловлены усложнением с программной точки зрения и поддержки аппаратных средств. Так минимальные системы РВ должны работать в составе АСУ, а универсальные системы являются самодостаточными.

Таким образом, рассматриваемая система бесконтактного управления в рамках данной классификации относится к гибким управляющим системам распознавания образов реального времени с обратной связью.

1.3. Основные обозначения.

Основной идеей, взятой за основу разрабатываемой системы бесконтактного управления, является определение направления взгляда человека – её оператора – и формирование управляющих команд в зависимости от полученного результата.

Объекты, управляемые при помощи данной системы, могут быть самыми разными – от виртуальных объектов в компьютерной программе, до различных транспортных средств, например, летательного аппарата или наземной движущейся платформы.

Цель алгоритма определения направления взгляда заключается в том, чтобы на основе анализа изображения оператора, полученного с камеры, определить местоположение точки, на которую направлен его взгляд. В целях дальнейшей разработки введено понятие «*экранная плоскость*», на которой и находится данная точка, на которую направлен взгляд оператора. Эту точку будем называть «*маркером*». Также, будем считать, что маркер может находиться только в прямоугольной области, расположенной на экранной плоскости, а сама эта область разбита на пиксели с разрешением Xs на Ys. (Рис. 9) В этом случае, координаты маркера (Cx,Cy) также будут измеряться в пикселях. В случае, когда при помощи системы бесконтактного

управления осуществляется управление виртуальным объектом на экране монитора, то под областью на экранной плоскости подразумевается дисплей монитора.

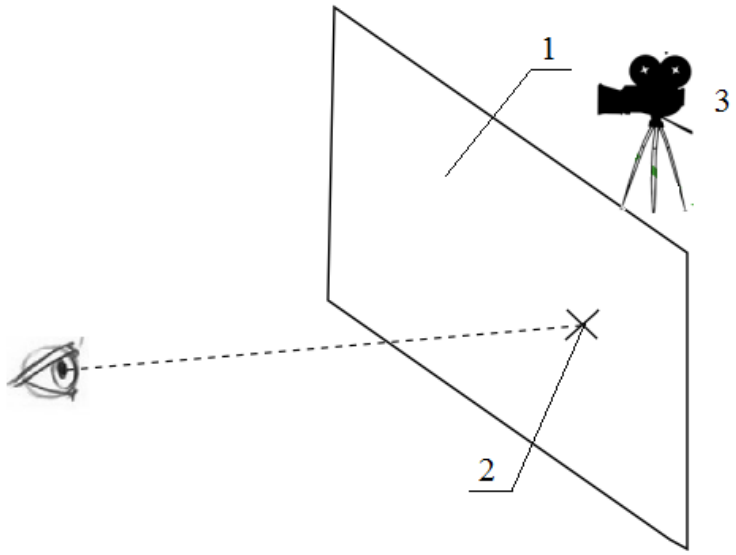


Рис. 9. Взаимное расположение объектов. 1 – экранная плоскость, 2 – маркер, 3 – видеокамера.

1.4. Оценка максимальной достижимой точности работы системы.

Как было сказано выше, максимальную точность позиционирования имеет подсистема определения направления взгляда по изображению зрачков глаз оператора.

Минимальное перемещение изображения зрачка на изображении составляет один пиксель. Примем также то, что линия взгляда оператора располагается по отношению к экранной плоскости под углом, близким к 90 градусам. Обозначим угол, на который поворачивается глазное яблоко оператора при минимальном перемещении изображения по горизонтали, как φ.

Equation Section (Next)

$$\varphi = \frac{1}{d'} \cdot \frac{\pi}{2}, \tag{1.1}$$

где d` - размер глазного яблока на изображении в пикселях.

$$d' = \frac{d}{2L\sin(\psi)} \cdot Xm, \tag{1.2}$$

где:

d – диаметр глазного яблока.

L – расстояние от камеры до зрачка.

Ψ – угол обзора камеры.

Xm – разрешение камеры по горизонтали.

Таким образом, минимальное смещение маркера на экранной плоскости будет приблизительно равно:

$$\delta x = Ls \sin(\varphi), \tag{1.3}$$

где Ls – расстояние от зрачка до экранной плоскости. Наиболее частым случаем будет расположение её на одном расстоянии с камерой, то есть Ls = L.

Рассчитаем минимальное смещение маркера при следующих параметрах:

L = 0,6м;

Xm = 640;

Ψ = 45°;

d = 0,024 м. (размер глазного яблока у всех людей одинаков)

В этом случае d` ≈ 18 пикселей, то есть угловое положение глазного яблока принимает 18 дискретных значений. Далее,

φ ≈ 10°;

δx ≈ 0,1м.

Таким образом, при вышеприведённых параметрах точность позиционирования составляет ±5см, что является достаточно низким результатом. Повысить точность можно за счёт повышения разрешающей способности камеры. При разрешении камеры по горизонтали 1920 пикселей и таких же остальных параметрах достигаются следующие результаты:

d` ≈ 54

φ ≈ 3°20`;

δx ≈ 0,035м.

Таким образом, за счёт повышения разрешения кадра, можно значительно увеличить точность системы. Однако, в то же время, повышение разрешения отрицательно скажется на быстродействии системы.

1.5. Выводы.

1. Большинство существующих систем бесконтактного управления использует специализированные сенсоры, что снижает удобство при работе с ними. В то же время, существуют технологии, позволяющие осуществлять управление, используя в качестве источника информации только видеоизображение, за счёт снижения быстродействия и точности управления.
2. Предложена двухканальная архитектура системы, с грубым каналом управления, использующим данные об угловом положении головы оператора и точным каналом, определяющим направление взгляда по изображению глаз оператора.
3. Система бесконтактного управления, описанная в данной диссертационной работе, относится к гибким управляющим системам распознавания образов реального времени с обратной связью.
4. Максимальная точность определения направления взгляда по положению зрачков на изображении сильно зависит от разрешающей способности камеры. При разрешении кадра 640 на 480 точек она составляет 10°, а при разрешении 1920 на 1080 точек - 3°20`.

ГЛАВА 2. АЛГОРИТМЫ ОБРАБОТКИ ИЗОБРАЖЕНИЯ.

В данной главе рассматриваются алгоритмы, используемые в данной работе для улучшения качества изображения, а также для определения места расположения на изображении ключевых областей затем, чтобы в дальнейшем определить направление взгляда оператора, в том числе алгоритмы, модифицированные или разработанные в рамках данной работы.

Основные задачи, которые необходимо решить в процессе обработки изображения, таковы:

1. Улучшение качества изображения для повышения качества обработки изображения на последующих этапах, в частности, компенсация неравномерного освещения.

2. Распознавание лица оператора на изображении.

3. Определение места расположения на изображении частей лица оператора (глаза, нос, губы).

4. Определение места расположения на изображении ключевых точек глаз оператора для определения направления его взгляда.

2.1 Сглаживание изображения при помощи фильтра Гаусса.

В первую очередь, необходимо очистить изображение от шумов, чтобы облегчить его последующую обработку.

Для подавления шума воспользуемся фильтрацией изображения при помощи свёртки, построенной при помощи функции Гаусса. Функция Гаусса для двумерного случая:

$$G\_gen(x,y,\sigma) := \frac{1}{2\pi\cdot\sigma^2}\cdot e^{\frac{-\left(x^2+y^2\right)}{2\sigma^2}}$$

Метод, при помощи которого получено ядро фильтра размером d с параметром σ:

$$\text{CrtGauFiltKern}(size,\sigma) := \left| \begin{array}{l} Shift \leftarrow \text{floor}\left(\frac{size}{2}\right) \\ \text{for } y \in 0..size-1 \\ \quad \text{for } x \in 0..size-1 \\ \quad\quad M_{y,x} \leftarrow G\_gen(x-Shift, y-Shift, \sigma) \\ \text{return } M \end{array} \right.$$

При выборе параметров фильтра необходимо руководствоваться следующими соображениями:

Влияние соседних пикселей изображения обратно пропорционально квадрату расстояния до них. Степень влияния соседних пикселей зависит от среднеквадратичного отклонения σ. Так как чрезмерное усреднение яркости изображения приведёт к потере его детализации, ограничимся σ = 1.

Размер ядра в таком случае также не стоит чрезмерно увеличивать, так как функция Гаусса стремительно убывает по мере удаления от центра. Поэтому можно ограничиться размером ядра в 7 пикселей.

Результат применения фильтра с выбранными параметрами изображён на рисунке 30.

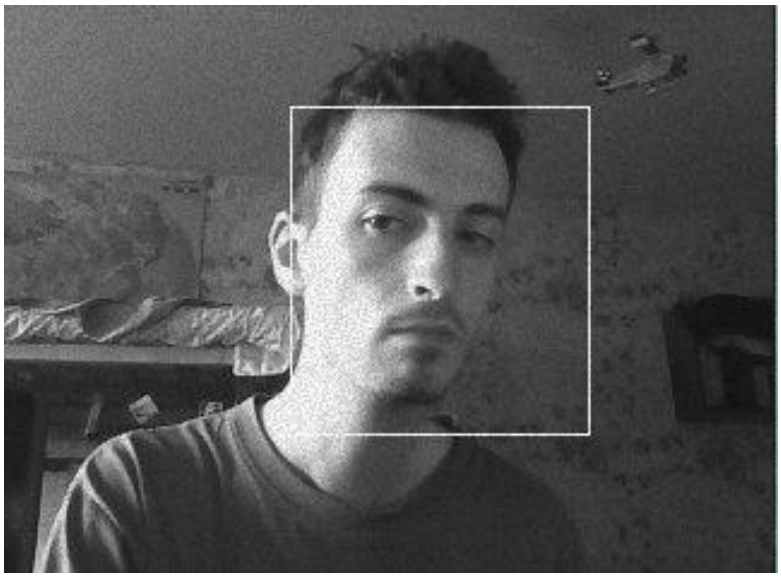


Рис.29. Исходное изображение.

Рис.30. Результат применения фильтра Гаусса с размером ядра 7 пикселей и среднеквадратичным отклонением σ=1

2.2 Компенсация неравномерного освещения.

Как показала опытная реализация программно-аппаратного комплекса в ходе данной диссертационной работы, одной из основных помех, встречающихся при эксплуатации комплекса, является неравномерность освещения сцены, в частности, лица оператора. Как правило, это происходит, когда источник света находится в стороне от лица, и свет падает сбоку, например, как на рисунке 30.

Чтобы устранить неравномерность освещения, применяется метод retinex [].

Суть его в том, что обрабатываемое изображение представляется как следующая комбинация:

$$I = R + L$$

,где R - базовая часть изображения, L – освещение. Получить L можно, применив к исходному изображению низкочастотную фильтрацию (фильтр Гаусса).

$$L` = G(I)$$

В данном случае, размер ядра фильтра и среднеквадратичное отклонение выбираются так, чтобы ядро фильтра было по размеру сопоставимо с обрабатываемым изображением.

Чтобы восстановить базовую часть изображения, из него нужно вычесть карту освещения, переведя перед этим её и исходное изображение в логарифмическую форму:

$$\log(R`) = \log(I) - \log(L`)$$

Результат выравнивания освещения можно увидеть на рисунке 31.

Рисунок 31. Компенсация неравномерного освещения.

2.2 Метод Виолы-Джонса.

2.3.1. Описание метода Viola Jones

Основные принципы, на которых основан метод, таковы:

1. используются изображения в интегральном представлении, что позволяет вычислять быстро необходимые объекты;
2. используются признаки Хаара, с помощью которых происходит поиск нужного объекта (в данном контексте, лица и его черт);
3. используется бустинг (от англ. boost – улучшение, усиление) для выбора наиболее подходящих признаков для искомого объекта на данной части изображения;
4. все признаки поступают на вход классификатора, который даёт результат «верно» либо «ложь»;

5. используются каскады признаков для быстрого отбрасывания окон, где не найдено лицо.

Обучение классификаторов идет очень медленно, но **результаты поиска лица очень быстры**, именно поэтому был выбран данный метод распознавания лиц на изображении. Виола-Джонс является одним из лучших по соотношению показателей эффективность распознавания/скорость работы. Также этот детектор обладает крайне низкой вероятностью ложного обнаружения лица. Алгоритм даже хорошо работает и распознает черты лица под небольшим углом, примерно до 30 градусов. При угле наклона больше 30 градусов процент обнаружений резко падает. И это не позволяет в стандартной реализации детектировать повернутое лицо человека под произвольным углом, что в значительной мере затрудняет или делает невозможным использование алгоритма в современных производственных системах с учетом их растущих потребностей.

Требуется подробный разбор принципов, на которых основан алгоритм Виолы-Джонса. Данный метод в общем виде ищет лица и черты лица по общему *принципу сканирующего окна*.

2.3.2. Принцип сканирующего окна

В общем виде, задача обнаружения лица и черт лица человека на цифровом изображении выглядит именно так:

имеется *изображение*, на котором *есть искомые объекты*. Оно представлено *двумерной матрицей пикселей размером* w*h, в которой каждый пиксель имеет значение:

— от 0 до 255, если это черно-белое изображение;

— от (0 до 255)*3, если это цветное изображение (компоненты R, G, B).

в результате своей работы, алгоритм должен определить лица и их черты и *пометить их* – поиск осуществляется в *активной области*изображения *прямоугольными признаками*, с помощью которых и описывается найденное лицо и его черты:

$rectangle_i = \{x,y,w,h,a\},$

где x, y – координаты центра i-го прямоугольника, w – ширина, h – высота, a – угол наклона прямоугольника к вертикальной оси изображения.

Иными словами, применительно к рисункам и фотографиям используется *подход на основе сканирующего окна (scanning window)*: сканируется изображение окном поиска (так называемое, окно сканирования), а затем применяется классификатор к каждому положению.*Система обучения* и выбора наиболее значимых признаков *полностью автоматизирована* и не требует вмешательства человека, поэтому данный подход работает быстро.

Задача поиска и нахождения лиц на изображении с помощью данного принципа часто бывает очередным шагом на пути к распознаванию характерных черт, к примеру, верификации человека по распознанному лицу или распознавания мимики лица.

2.3.3. Интегральное представление изображений

Для того, чтобы производить какие-либо действия с данными, используется *интегральное представление* изображений [3] в методе Виолы-Джонса. Такое представление используется часто и в других методах, к примеру, в вейвлет-преобразованиях, SURF и многих других разобранных алгоритмах. Интегральное представление позволяет быстро рассчитывать *суммарную яркость* произвольного прямоугольника на данном изображении, причем какой бы прямоугольник не был, время расчета неизменно.

2.3.4. Признаки Хаара

Признак — отображение f: X => D_f, где D_f — множество допустимых значений признака. Если заданы признаки $f_1,\ldots,f_n$, то вектор признаков x = $(f_1(x),\ldots,f_n(x))$ называется *признаковым описанием* объекта x ∈ X. Признаковые описания допустимо отождествлять с самими объектами. При этом множество X = D_{f1}* …* D_{fn} называют признаковым пространством [1].

Признаки делятся на следующие типы в зависимости от множества D_f:

- бинарный признак, D_f = {0,1};
- номинальный признак: D_f — конечное множество;
- порядковый признак: D_f — конечное упорядоченное множество;

- количественный признак: D_f — множество действительных чисел.

Естественно, бывают прикладные задачи с разнотипными признаками, для их решения подходят далеко не все методы.

В стандартном методе Виолы – Джонса используются прямоугольные признаки, изображенные на рисунке ниже, они называются *примитивами Хаара*:

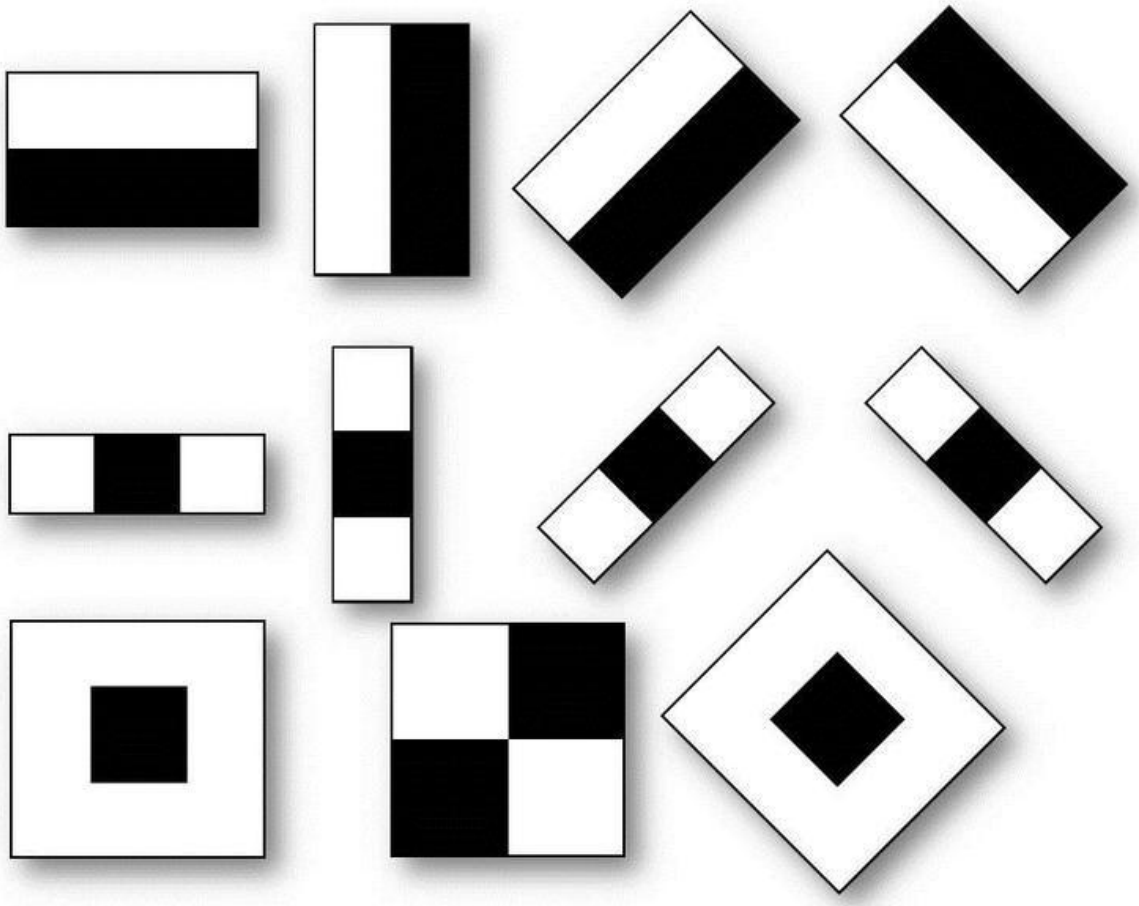


Рис. 24. Примитивы Хаара.

В расширенном методе Виолы – Джонса, использующемся в библиотеке OpenCV используются дополнительные признаки:

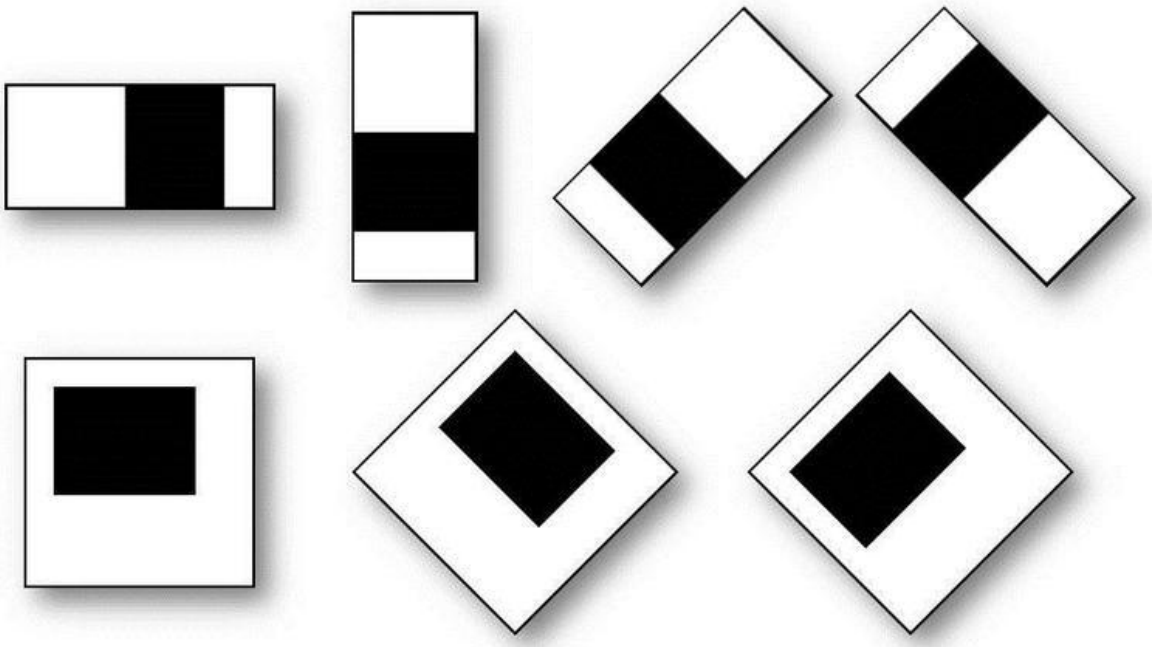


Рис. 25. Дополнительные примитивы Хаара.

Вычисляемым значением такого признака будет

`F = X-Y`, (1.5)

где X – сумма значений яркостей точек закрываемых *светлой частью признака*, а Y – сумма значений яркостей точек закрываемых *темной частью признака*. Для их вычисления используется понятие интегрального изображения, рассмотренное выше. Признаки Хаара дают точечное значение *перепада яркости по оси X и Y соответственно*.

2.3.5. Сканирование окна

Алгоритм сканирования окна с признаками выглядит так:

1. есть исследуемое изображение, выбрано окно сканирования, выбраны используемые признаки;

2. далее окно сканирования начинает последовательно двигаться по изображению с шагом в 1 ячейку окна (допустим, размер самого окна есть 24*24 ячейки);
3. при сканировании изображения в каждом окне вычисляется приблизительно 200 000 вариантов расположения признаков, за счет изменения масштаба признаков и их положения в окне сканирования;
4. сканирование производится последовательно для различных масштабов;
5. масштабируется не само изображение, а сканирующее окно (изменяется размер ячейки);
6. все найденные признаки попадают к классификатору, который «выносит вердикт».

В процессе поиска вычислять все признаки на маломощных настольных ПК просто невозможно. Следовательно, классификатор должен реагировать *только на определенное, нужное подмножество* всех *признаков*. Совершенно логично, что надо обучить классификатор нахождению лиц по данному определенному подмножеству. Это можно сделать, обучая вычислительную машину *автоматически*.

2.3.6. Используемая в алгоритме модель машинного обучения

Обучение машины — это процесс получения модулем новых знаний. Есть признанное определение данному процессу:

«Машинное обучение — это наука, изучающая компьютерные алгоритмы, автоматически улучшающиеся во время работы» (Michel, 1996)

. Данный процесс входит в концепцию и технологию под названием Data mining (извлечение информации и интеллектуальный анализ данных), куда входят помимо Машинного обучения такие дисциплины, как Теория баз данных, Искусственный интеллект, Алгоритмизация, Распознавание образов и прочие.

Машинное обучение в методе Виолы-Джонса решает такую задачу как *классификация*.

В контексте алгоритма, имеется множество объектов (изображений), разделённых некоторым образом на классы. Задано конечное множество изображений,

для которых известно, к какому классу они относятся (к примеру, это может быть класс «фронтальное положение носа»). Это множество называется *обучающей выборкой*. Классовая принадлежность остальных объектов не известна. Требуется построить алгоритм, способный классифицировать произвольный объект из исходного множества [4].

Классифицировать объект — значит, указать номер (или наименование класса), к которому относится данный объект.

Классификация объекта — номер или наименование класса, выдаваемые алгоритмом классификации в результате его применения к данному конкретному объекту.

Классификатор(classifier) — в задачах классификации это аппроксимирующая функция, выносящая решение, к какому именно классу данный объект принадлежит.

Обучающая выборка – конечное число данных.

В машинном обучении задача классификации относится к разделу **обучения с учителем** *когда классы поделены*. Распознавание образов по сути своей и есть классификация изображений и сигналов. В случае алгоритма Виолы-Джонса для идентификации и распознавания лица классификация является *двухклассовой*.
Постановка классификации выглядит следующим образом:

Есть X – множество, в котором хранится описание объектов, Y – конечное множество номеров, принадлежащих классам. Между ними есть зависимость – отображение Y*: X => Y. Обучающая выборка представлена $X_m = \{(x_1, y_1), \ldots, (x_m, y_m)\}$. Конструируется функция f от вектора признаков X, которая выдает ответ для любого возможного наблюдения X и способна классифицировать объект $x \in X$. Данное простое правило должно хорошо работать и на новых данных.

2.3.7. Применяемый в алгоритме бустинг и разработка AdaBoost

Для решения проблемы данного, столь сложного обучения существует технология бустинга.

Бустинг — комплекс методов, способствующих повышению точности аналитических моделей. Эффективная модель, допускающая мало ошибок классификации, называется «сильной». «Слабая» же, напротив, не позволяет

надежно разделять классы или давать точные предсказания, делает в работе большое количество ошибок. Поэтому бустинг (от англ. boosting – повышение, усиление, улучшение) означает дословно «усиление» «слабых» моделей [5] – это процедура последовательного построения композиции алгоритмов машинного обучения, когда каждый следующий алгоритм стремится компенсировать недостатки композиции всех предыдущих алгоритмов.

Идея бустинга была предложена Робертом Шапиром (Schapire) в конце 90-х годов [6], когда надо было найти решение вопроса о том, чтобы имея множество плохих (незначительно отличающихся от случайных) алгоритмов обучения, получить один хороший. В основе такой идеи лежит построение цепочки (ансамбля) классификаторов [5, 6], который называется каскадом, каждый из которых (кроме первого) обучается на ошибках предыдущего. Например, один из первых алгоритмов бустинга Boost1 использовал каскад из 3-х моделей, первая из которых обучалась на всем наборе данных, вторая – на выборке примеров, в половине из которых первая дала правильные ответы, а третья — на примерах, где «ответы» первых двух разошлись. Таким образом, имеет место последовательная обработка примеров каскадом классификаторов, причем так, что задача для каждого последующего становится труднее. Результат определяется путем простого голосования: пример относится к тому классу, который выдан большинством моделей каскада.
Бустинг представляет собой жадный алгоритм построения композиции алгоритмов (greedy algorithm) — это алгоритм, который на каждом шагу делает локально наилучший выбор в надежде, что итоговое решение будет оптимальным. Бустинг над решающими деревьями считается одним из наиболее эффективных методов с точки зрения качества классификации. Во многих экспериментах наблюдалось практически неограниченное уменьшение частоты ошибок на независимой тестовой выборке по мере наращивания композиции. Более того, качество на тестовой выборке часто продолжало улучшаться даже после достижения безошибочного распознавания всей обучающей выборки. Это перевернуло существовавшие долгое время представления о том, что для повышения обобщающей способности необходимо ограничивать сложность алгоритмов. На примере бустинга стало понятно, что хорошим качеством

могут обладать сколь угодно сложные композиции, если их правильно настраивать [5]. Математически бустинг объянсяется так:

Наряду с множествами X и Y вводится вспомогательное множество R, называемое пространством оценок. Рассматриваются алгоритмы, имеющие вид суперпозиции a(x) = C(b(x)), где функция b: X → R называется алгоритмическим оператором, функция C: R → Y –решающим правилом.

Многие алгоритмы классификации имеют именно такую структуру: сначала вычисляются оценки принадлежности объекта классам, затем решающее правило переводит эти оценки в номер класса. Значение оценки, как правило, характеризует степень уверенности классификации.

Алгоритмическая композиция – алгоритм a: X → Y вида

a(x) = C(F(b1(x), . . . , bT (x)), x ∈ X (1.6)

, составленный из алгоритмических операторов bt :X→R, t=1,..., T, корректирующей операции F: RT→R и решающего правила C: R→Y.

Базовыми алгоритмами обозначаются функции at(x) = C(bt(x)), а при фиксированном решающем правиле C — и сами операторы bt(x).

Суперпозиции вида F(b1,..., bT) являются отображениями из X в R, то есть, опять же, алгоритмическими операторами.

В задачах классификации на два непересекающихся класса в качестве пространства оценок обычно используется множество действительных чисел. Решающие правила могут иметь настраиваемые параметры. Так, в алгоритме Виолы-Джонса используется пороговое решающее правило, где, как правило, сначала строится оператор при нулевом значении, а затем подбирается значение оптимальное. Процесс последовательного обучения базовых алгоритмов применяется, пожалуй, чаще всего при построении композиций.

Критерии останова могут использоваться различные, в зависимости от специфики задачи, возможно также совместное применение нескольких критериев:

- построено заданное количество базовых алгоритмов T;
- достигнута заданная точность на обучающей выборке;

- достигнутую точность на контрольной выборке не удаётся улучшить на протяжении последних нескольких шагов при определенном параметре алгоритма.

Развитием данного подхода явилась разработка более совершенного семейства алгоритмов бустинга AdaBoost (adaptive boosting – адаптированное улучшение), предложенная Йоавом Фройндом (Freund) и Робертом Шапиром (Schapire) в 1999 году [9], который может использовать произвольное число классификаторов и производить обучение на одном наборе примеров, поочередно применяя их на различных шагах. Рассматривается задача классификации на два класса, Y = {−1,+1}. К примеру, базовые алгоритмы также возвращают только два ответа −1 и +1, и решающее правило фиксировано: C(b) = sign(b). Искомая алгоритмическая композиция имеет вид:

$$a(x) = C(F(b_1(x), \ldots, b_T(x)) = \text{sign}(\sum_{t=1}^{T} a_t b_t(x)), \quad x \in X. \quad (1.7)$$

Функционал качества композиции Qt определяется как число ошибок, допускаемых ею на обучающей выборке:

$$Q(b, W^l) = Q_T = \sum_{t=1}^{l} [\, y_i \sum_{t=1}^{T} a_t b_t(x_i)] < 0, \quad (1.8)$$

где Wl = (w1, …, wl) – вектор весов объектов.

Для решения задачи AdaBoosting’а нужна экспоненциальная аппроксимация пороговой функции потерь [z<0], причем экспонента Ez = e-z(видно на рисунке, демонстрирующем работу AdaBoost ниже).

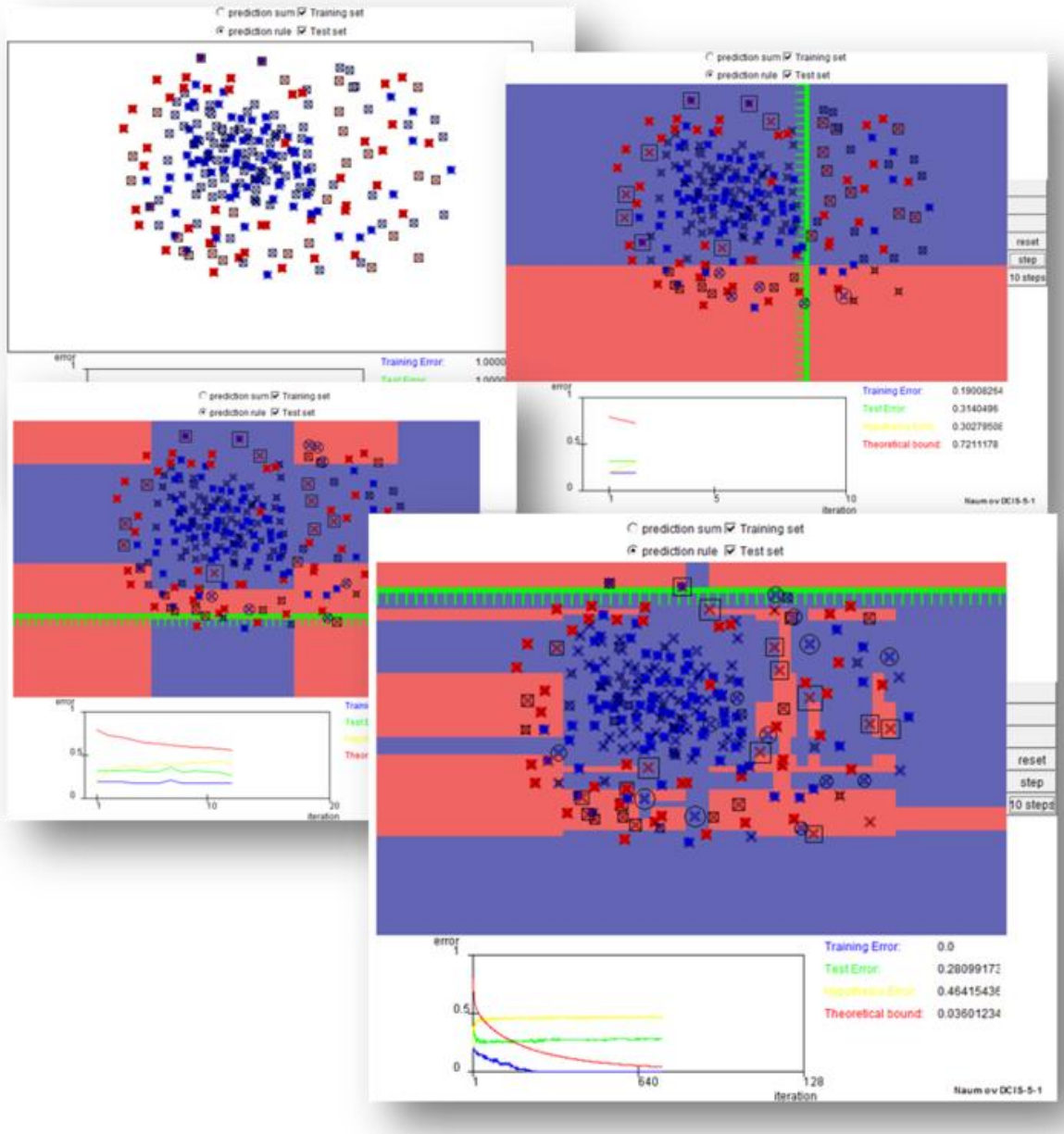


Рис. 26. Демонстрация алгоритма обучения AdaBoost.

Плюсы AdaBoost:

- хорошая обобщающая способность. В реальных задачах практически всегда строятся композиции, превосходящие по качеству базовые алгоритмы. Обобщающая способность может улучшаться по мере увеличения числа базовых алгоритмов;
- простота реализации;

- собственные накладные расходы бустинга невелики. Время построения композиции практически полностью определяется временем обучения базовых алгоритмов;
- возможность идентифицировать объекты, являющиеся шумовыми выбросами. Это наиболее «трудные» объекты x_i, для которых в процессе наращивания композиции веса w_i принимают наибольшие значения.

Минусы AdaBoost:

- Бывает переобучение при наличии значительного уровня шума в данных. Экспоненциальная функция потерь слишком сильно увеличивает веса «наиболее трудных» объектов, на которых ошибаются многие базовые алгоритмы. Однако именно эти объекты чаще всего оказываются шумовыми выбросами. В результате AdaBoost начинает настраиваться на шум, что ведёт к переобучению. Проблема решается путём удаления выбросов или применения менее «агрессивных» функций потерь. В частности, применяется алгоритм GentleBoost;
- AdaBoost требует достаточно длинных обучающих выборок. Другие методы линейной коррекции, в частности, бэггинг, способны строить алгоритмы сопоставимого качества по меньшим выборкам данных;
- Бывает построение неоптимального набора базовых алгоритмов. Для улучшения композиции можно периодически возвращаться к ранее построенным алгоритмам и обучать их заново.
- Бустинг может приводить к построению громоздких композиций, состоящих из сотен алгоритмов. Такие композиции исключают возможность содержательной интерпретации, требуют больших объёмов памяти для хранения базовых алгоритмов и существенных временных затрат на вычисление классификаций.

В наши дни подход усиления простых классификаторов является популярным и, вероятно, наиболее эффективным методом классификации за счёт высокой скорости и эффективности работы и относительной простоты реализации.

ГЛАВА 3. АЛГОРИТМ ОПРЕДЕЛЕНИЯ НАПРАВЛЕНИЯ ВЗГЛЯДА ОПЕРАТОРА.

Алгоритм определения направления взгляда оператора

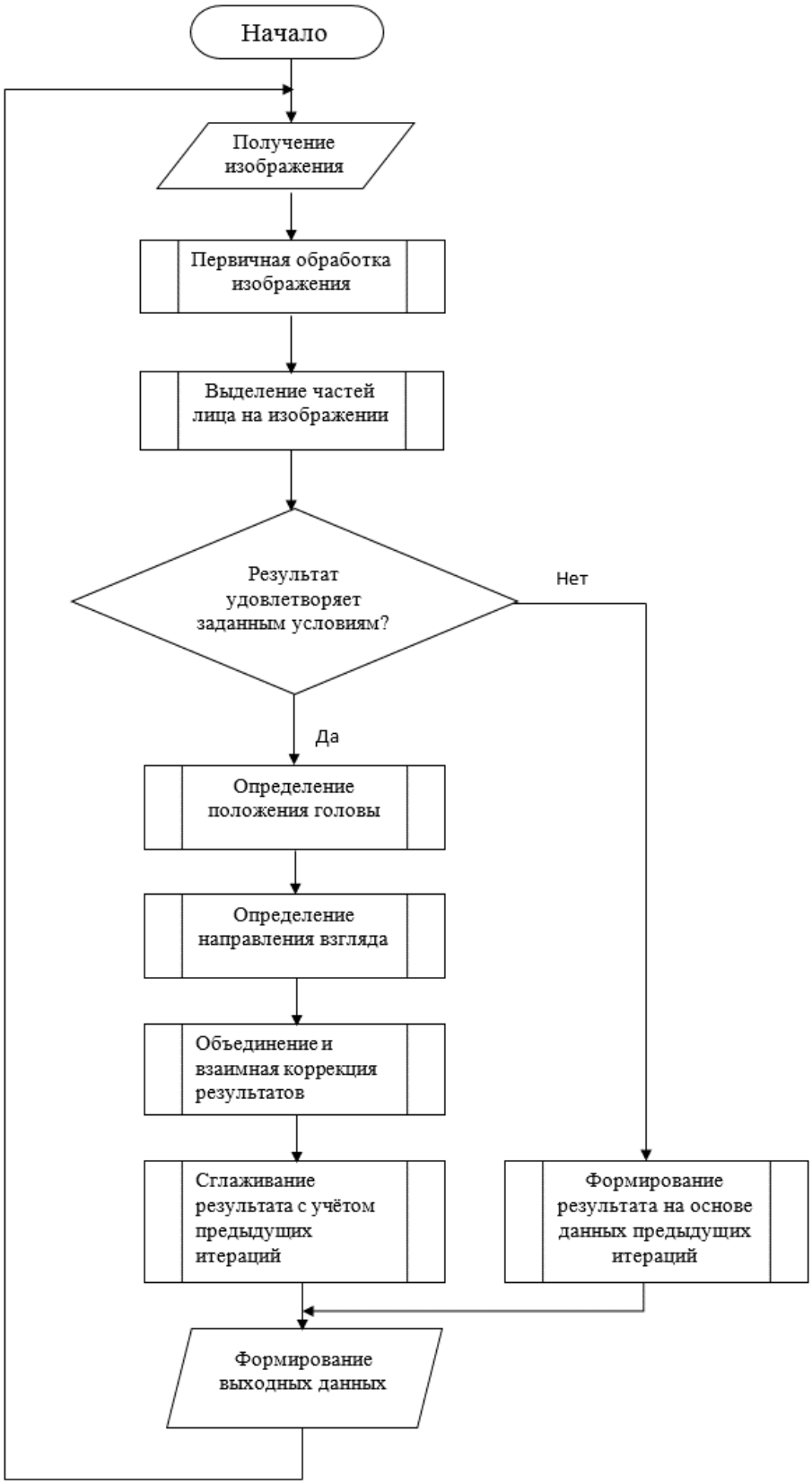


Рис. 1. Блок-схема алгоритма работы системы.

3.1. Первичная обработка изображения.

3.2. Выделение частей лица на изображении.

Следующим шагом алгоритма бесконтактного управления является поиск на изображении лица оператора, а также отдельных его частей, а именно:

- глаза
- нос
- губы

На основании взаимного относительного расположения этих частей на изображении можно судить о положении головы оператора. Также, местоположение глаз необходимо знать для работы алгоритма определения направления взгляда.

Для решения данной задачи подходят несколько способов [], а именно:

1. Применение свёрточной нейронной сети.
2. Метод SURF
3. Метод Виолы-Джонса
4. Метод Виолы-Джонса с переменным шагом.

Хотя каждый из этих методов решает задачу поиска частей лица на изображении, они различаются по своей эффективности, быстродействию и другим параметрам. По итогам вычислительных экспериментов был выбран метод Виолы-Джонса с переменным шагом окна.

3.3. Верификация координат элементов лица и определение углового положения головы.

В результате поиска частей лица на изображении, для каждой из них находится прямоугольная область, координаты центра которой можно использовать в дальнейшем. Полученные данные о расположении частей лица на изображении могут быть недостоверными из-за ошибок алгоритма распознавания. Поэтому, наряду с определением положения головы оператора по взаимному расположению частей лица на изображении, проводится ряд проверок, позволяющий отсеять недостоверные результаты. Пример такой проверки: линия, проведённая через центры областей,

отмечающих губы и нос, пересекает отрезок, соединяющий центры областей глаз посередине и под прямым углом (с допустимой погрешностью).

В итоге, координаты маркера на двухмерной плоскости, полученные на основе положения головы оператора, определяются по следующим формулам:

$$\text{B1x} = (Nx - \frac{Rx + \frac{Rx - Lx}{2} + Mx}{2}) \cdot K1x \cdot X + \frac{X}{2} \ ; \qquad (4)$$

$$\text{B1y} = (Ny - \frac{Ry + \frac{Ry - Ly}{2} + My}{2}) \cdot K1y \cdot Y + \frac{Y}{2} \ , \qquad (5)$$

где

B1x, B1y – координаты результирующего маркера по осям x и y соответственно,

Rx, Ry – координаты центра области правого глаза,

Lx, Ly – координаты центра области левого глаза,

Nx, Ny – координаты центра области носа,

Mx, My – координаты центра области губ,

K1x, K1y – корректирующие коэффициенты, зависящие от характеристик камеры, различий в характере вертикальных и горизонтальных движений головы человека и подбираемые экспериментально.

X, Y – размер области, в пределах которой осуществляется управление, как правило это размер в пикселях экрана монитора.

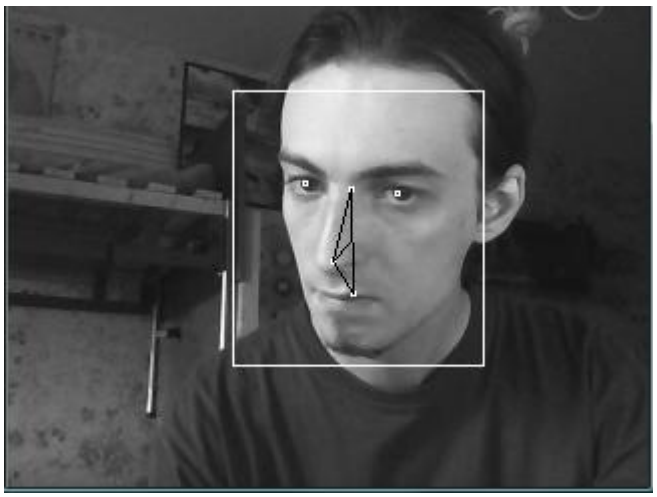


Рис. 31. Определение углового положения головы.

*3.*4. Определение направления взгляда.

Центры найденных эллипсов указывают на расположение зрачков. Затем смещение зрачков относительно центров областей глаз умножается на коэффициенты, линейно зависящие от положения головы оператора. В результате получаются координаты B2x и B2y результирующего маркера, определяемые направлением взгляда:

$$B2x = \frac{(1 - K2x \cdot (\mathrm{B1x} - \frac{X}{2})) \cdot \Delta Rx + (1 + K2x \cdot (\mathrm{B1x} - \frac{X}{2})) \cdot \Delta Lx}{2} K3x + \frac{X}{2}; \quad (6)$$

$$B2y = (\frac{\Delta Ly + \Delta Ry}{2}) \mathrm{K3y} + (\mathrm{B1y} - \frac{Y}{2}) \mathrm{K2y} + \frac{Y}{2}, \quad (7)$$

Где

ΔRx, ΔRy – Смещение зрачка правого глаза

ΔLx, ΔLy – Смещение зрачка левого глаза

K2x, K2y – Корректирующие коэффициенты, характеризующие влияние углового положения головы на изображение зрачков.

K3x, K3y - Корректирующие коэффициенты, зависящие от характеристик камеры, различий в характере вертикальных и горизонтальных движений глаз человека.

3.5. Объединение и взаимная коррекция результатов.

Рассматриваемая система бесконтактного управления в качестве управляющих воздействий использует данные, полученные от разных подсистем, а именно:

1. Подсистема определения направления взгляда оператора.

2. Подсистема определения поворота головы оператора.

Эти данные, в силу неоднородности их источников различаются по своим параметрам, таким как точность, достоверность и периодичность обновления. Например, подсистема определения направления взгляда оператора на основании распознавания зрачка глаза на видеоизображении даёт результат с наивысшей для трёх рассматриваемых каналов точностью, но из-за высокой вероятности возникновения ошибок распознавания второго рода, эти данные могут быть недостоверными.

Это послужило причиной для создания алгоритма объединения и взаимной коррекции этих данных для выработки единого сигнала, который и будет использован при управлении.

Представим желаемое управляющее воздействие в дискретный момент времени t как точку C_t на экранной плоскости, заданную дискретными координатами x и y, а воздействие, полученное по каналу i, как точку B_t^i на этой же плоскости.

В ходе разработки были опробованы два способа объединения сигналов – упрощённый и модифицированный.

Упрощённый способ объединения сигналов.

Первоначальный способ, возникший в результате разработки, заключается в получении результирующей точки C_t через усреднение координат точек B_t^i :

$$C_t = \left(\frac{\sum_{i=1}^{n} x_t^i}{n}, \frac{\sum_{i=1}^{n} y_t^i}{n} \right) \tag{8}$$

Наглядно данный способ показан на рисунке 33:

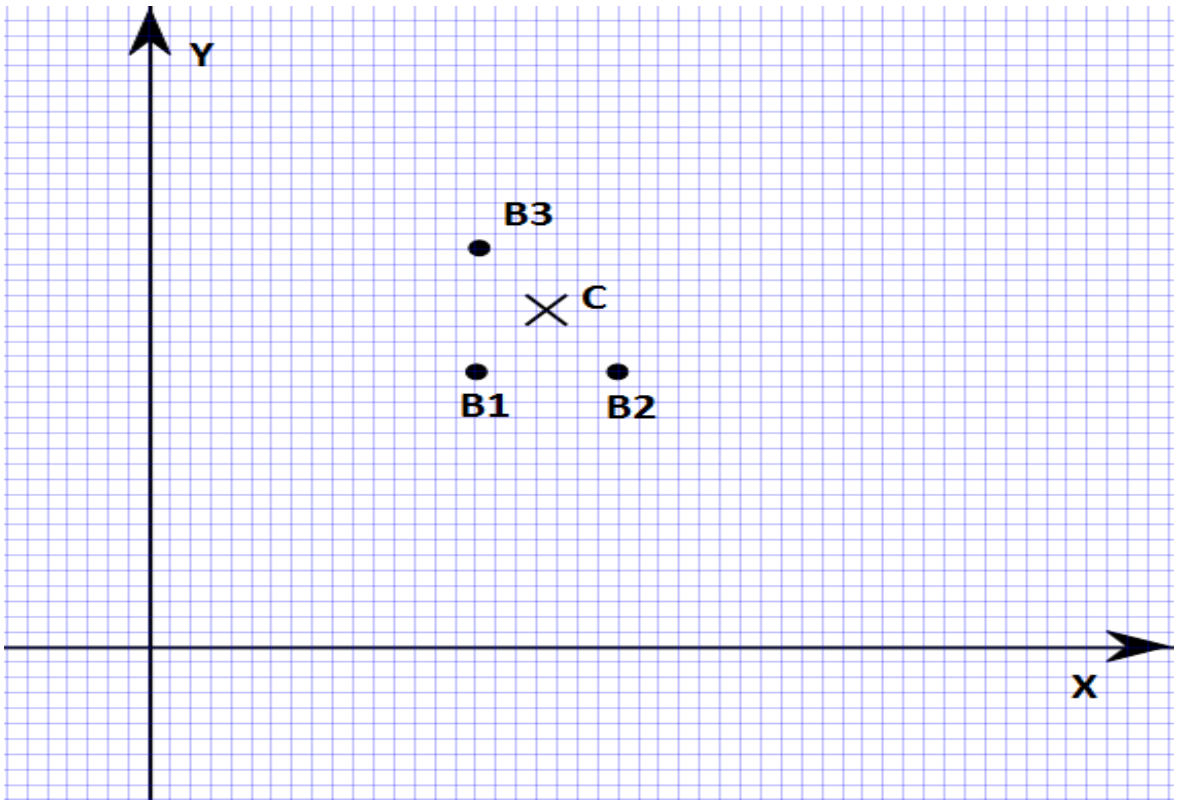


Рис. 33. Иллюстрация упрощённого способа объединения сигналов

Подобный подход к решению задачи комплексирования является довольно простым и легко реализуемым, но не учитывает неоднородность данных, получаемых от разных подсистем. Поэтому целесообразно провести модификацию данного способа.

Модифицированный способ объединения сигналов.

Введём для каждого из каналов набор характеристических коэффициентов:

R_i – достоверность данных канала i.

S_i – точность данных канала i.

τ_i – постоянная времени, характеризующая убывание актуальности

данных канала i с течением времени

Для начала, определим актуальность данных с канала, если они были получены не в текущий момент времени t. Особенности реализации подсистем таковы, что различные каналы работают асинхронно, выдавая результат по мере получения. То есть, для канала i в момент времени t есть интервал времени ΔT_t^i, прошедшего с последнего опроса этого канала.

Примем закон убывания актуальности A_t^i данных с канала i за

$$A_t^i = e^{-\frac{\Delta T_t^i}{\tau_i}} \tag{9}$$

Затем, получим результирующую точку как средневзвешенное значение координат $B_t^i(\mathrm{x}_t^i, \mathrm{y}_t^i)$ от разных n каналов:

$$C_t = \left(\frac{\sum_{i=1}^{n} x_t^i A_t^i S_i}{\sum_{i=1}^{n} A_t^i S_i}, \frac{\sum_{i=1}^{n} y_t^i A_t^i S_i}{\sum_{i=1}^{n} A_t^i S_i} \right) \tag{10}$$

Так как показания каналов могут быть недостоверными, исключим из результата данные от тех каналов, для которых расстояние от результирующей точки без учёта рассматриваемого канала j больше, чем порог доверия R_i:

$$\sqrt{\left(\frac{\sum_{i=1,i\neq j}^{n} x_t^i A_t^i S_i}{\sum_{i=1,i\neq j}^{n} A_t^i S_i} - x_t^j \right)^2 + \left(\frac{\sum_{i=1,i\neq j}^{n} y_t^i A_t^i S_i}{\sum_{i=1,i\neq j}^{n} A_t^i S_i} - y_t^j \right)^2} > R_j \tag{11}$$

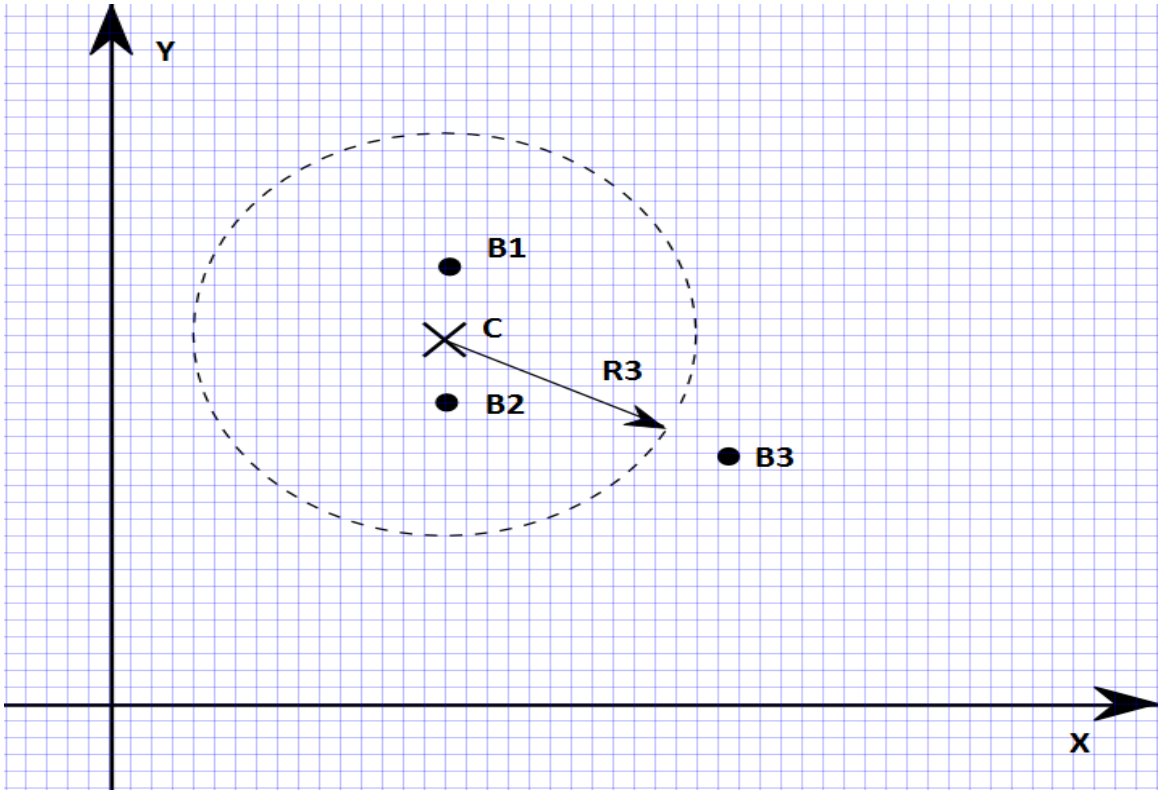


Рис. 34. Исключение недостоверных результатов. Сигнал от подсистемы 3 не учитывается при получении результирующего сигнала, так как расстояние от общего центра C точек B_1 и B_2 до точки B_3 больше, чем порог доверия R_3.

Определение значений коэффициентов для рассматриваемого алгоритма, которые были бы оптимальны для решения задачи, является известной проблемой, по причине их количества (в данном случае их 6) и отсутствия чёткого критерия оптимальности. Для поиска искомых значений коэффициентов R_i, S_i, τ_i при разработке системы бесконтактного управления использовался генетический алгоритм (ГА).

Критерием отбора для генетического алгоритма принята корректная работа системы на обучающем примере, подготовленном следующим образом: Был произведён тестовый запуск системы, во время которого оператору предлагалось перемещать бесконтактным способом указатель на экране вслед за целью, движение которой определялось программно. Таким образом были получены как входные данные – записанные сигналы подсистем, так и желаемые выходные – положение цели.

Далее, за критерий отбора было принято среднеквадратичное по времени отклонение результирующего воздействия, полученного в результате работы алгоритма объединения сигналов, от желаемого результата из обучающего примера.

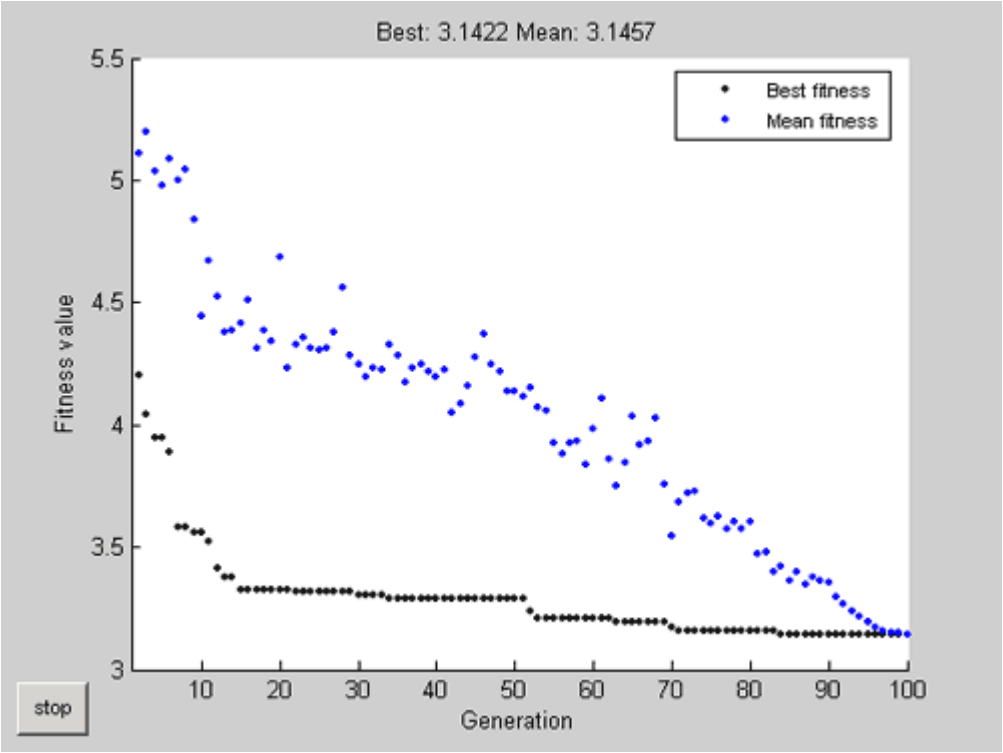


Рис. 35. Зависимость лучшего и среднего по популяции значения отклонения полученного результата от искомого от поколения при работе ГА.

Оптимизация коэффициентов при помощи генетического алгоритма была проведена в среде Matlab. Так как при количестве подсистем, равном двум, комплексирование вышеописанным способом не представляет вычислительной сложности, искомые значения коэффициентов были найдены генетическим алгоритмом менее чем за 100 поколений.

3.6. Итеративное сглаживание результата.

С целью обеспечить плавность управления целесообразно провести сглаживание результата с учётом данных, полученных при обработке предыдущих кадров видеоизображения. Самым простым, но тем не менее дающим удовлетворительные результаты способом является усреднение по координатам результатов последних N итераций. При разработке опытного образца системы было принято N = 5.

Кроме того, можно усовершенствовать сглаживание результата, применив фильтр Калмана [6]. Фильтр Калмана предназначен для рекурсивного оценивания вектора состояния априорно известной динамической системы, то есть для расчёта текущего состояния системы необходимо знать текущее измерение, а также предыдущее состояние самого фильтра. Таким образом, фильтр Калмана, подобно другим

рекурсивным фильтрам, реализован во временном, а не в частотном представлении, но в отличие от других подобных фильтров, фильтр Калмана оперирует не только оценками состояния, а еще и оценками неопределенности (плотности распределения) вектора состояния, опираясь на формулу Байеса условной вероятности. Наглядный пример возможностей фильтра — получение точных, непрерывно обновляемых оценок положения и скорости некоторого объекта по результатам временного ряда неточных измерений его местоположения.

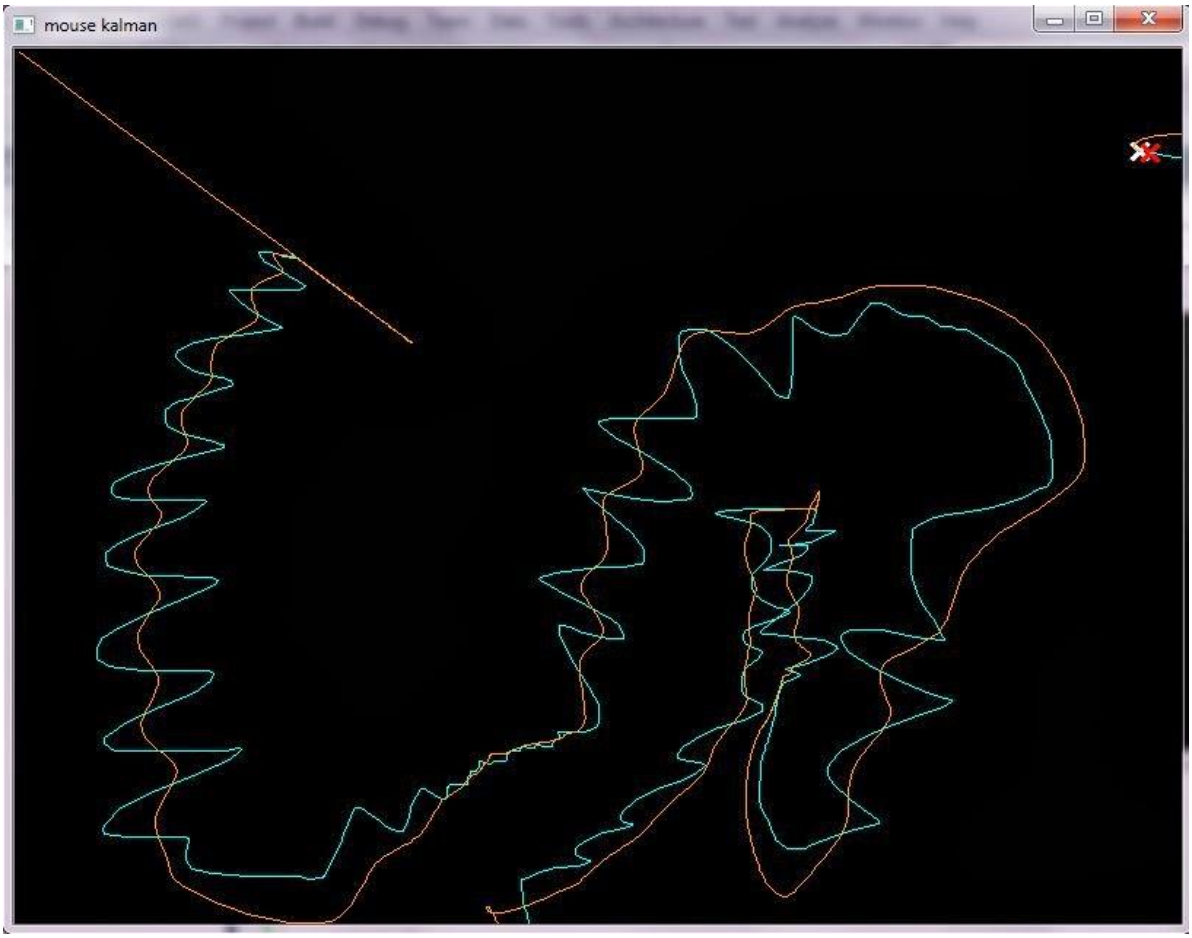


Рис. 36. Пример сглаживания выходного значения при помощи фильтра Калмана. Синяя линия – исходная траектория движения указателя, красная – сглаженное значение.

3.7. Выводы.

1. Предложен алгоритм определения направления взгляда, работающий в соответствии с принятым ранее двухканальным подходом к построению системы. Алгоритм состоит из нескольких крупных структурных частей, подробно описанных в данной главе.
2. Приведены основные соотношения, используемые для преобразования координат найденных на изображении частей лица в координаты маркера на экранной плоскости системы.
3. Предложен способ объединения и коррекции сигналов от разных каналов системы бесконтактного управления.
4. Чтобы сгладить движение указателя, осуществляемое при помощи системы бесконтактного управления и тем самым повысить плавность управления, можно использовать усреднение координат маркера по времени или применить фильтр Калмана.

ГЛАВА 4. ЭКСПЕРИМЕНТАЛЬНАЯ ОЦЕНКА ЭФФЕКТИВНОСТИ СИСТЕМЫ.

4.1. Программная реализация алгоритмов работы системы бесконтактного управления.

На основе вышеописанных алгоритмов был разработан опытный образец системы. Разработка велась на языках C и C++ с применением программных библиотек QT и OpenCV для платформы Win32. Для наглядной демонстрации работы системы осуществляется управление бесконтактным способом простейшим авиасимулятором (рис. 37.)

Рис. 37. Интерфейс работы с экспериментальной реализацией системы бесконтактного управления.

Интерфейс системы состоит из следующих основных элементов:

- **Окно с изображением, получаемым камерой**. При попадании в кадр человеческого лица, определяется направление взгляда и формируется управляющий сигнал. Область, в которой находится распознанное лицо на изображении, обводится рамкой белого цвета.
- **Окно симулятора.** В этом окне показан летательный аппарат (самолёт), летящий над бесконечным ландшафтом. Управление самолётом

максимально упрощено и заключается в передаче ему 8 команд-направлений – вверх, вниз, влево, вправо и четырёх диагональных направлений.

Так как в результате работы алгоритма определения направления взгляда получаются координаты одной точки – маркера – на экранной плоскости, чтобы преобразовать их в дискретные команды-направления сделано следующее: Экранная плоскость поделена на 9 контрольных областей, образующих прямоугольник 3 на 3 клетки. Центральная контрольная область соответствует нейтральному положению ручки управления, остальным присвоено по одной из вышеприведённых восьми команд.

Попадание маркера в одну из областей приводит к активации соответствующей команды и передаче её в симуляцию. Работа контрольных областей показана на рисунке 38.

Рис. 38. Контрольные области. Чёрной точкой показан маркер, серым подсвечена активная контрольная область

4.2. Способ проведения испытаний.

С целью оценки эффективности разработанной алгоритмической базы системы бесконтактного управления была проведена серия испытаний с использованием вышеописанной программной реализации системы.

Каждое из этих испытаний проводилось по следующей схеме:

4.2.1. Формирование задачи для оператора.

Для оценки эффективности управления бесконтактным способом необходимо иметь данные о желаемом значении выходного сигнала испытуемой системы бесконтактного управления, которое будет использоваться как эталон для сравнения с полученным реальным значением выходного сигнала. С этой целью реализовано программное формирование положения цели на экранной плоскости. Задача оператора при проведении испытания заключается в том, чтобы бесконтактным образом совместить маркер, показывающий выходное значение системы бесконтактного управления, и цель, положение которой определяется программно.

Формирование положения цели производилось двумя способами:

1. Координаты цели (Xt, Yt) выбираются случайным образом в пределах области управления. При достижении цели с заданной точностью D0 происходит отсчёт времени T на фиксацию цели, а затем выбирается следующее положение цели. (Рис.39).

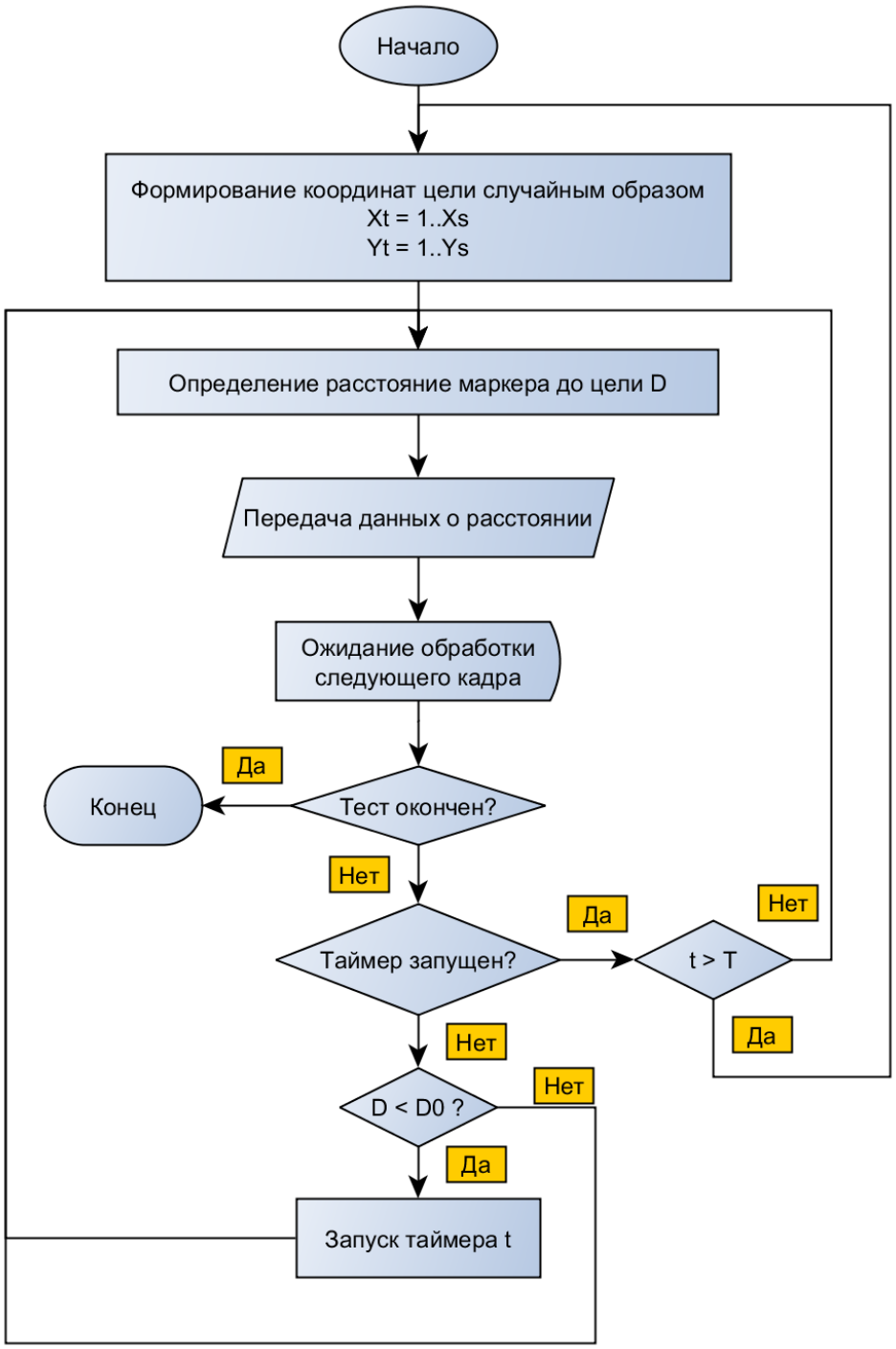


Рис. 39. Блок-схема алгоритма формирования эталонного сигнала для испытаний – вариант 1.

2. Координаты цели формируются при помощи упрощённого варианта численного интегрирования уравнения движения цели. В этом случае начальные координаты цели соответствуют центру области управления. При обработке каждого кадра к ним добавляются значения Vx и Vy – аналоги скорости, которые, в свою очередь, равны 0 в начале испытания и изменяются каждый кадр путём добавления

значений Ax и Ay – аналогов ускорения. Значения Ax и Ay задаются случайно каждые N кадров (Рисунок 40).

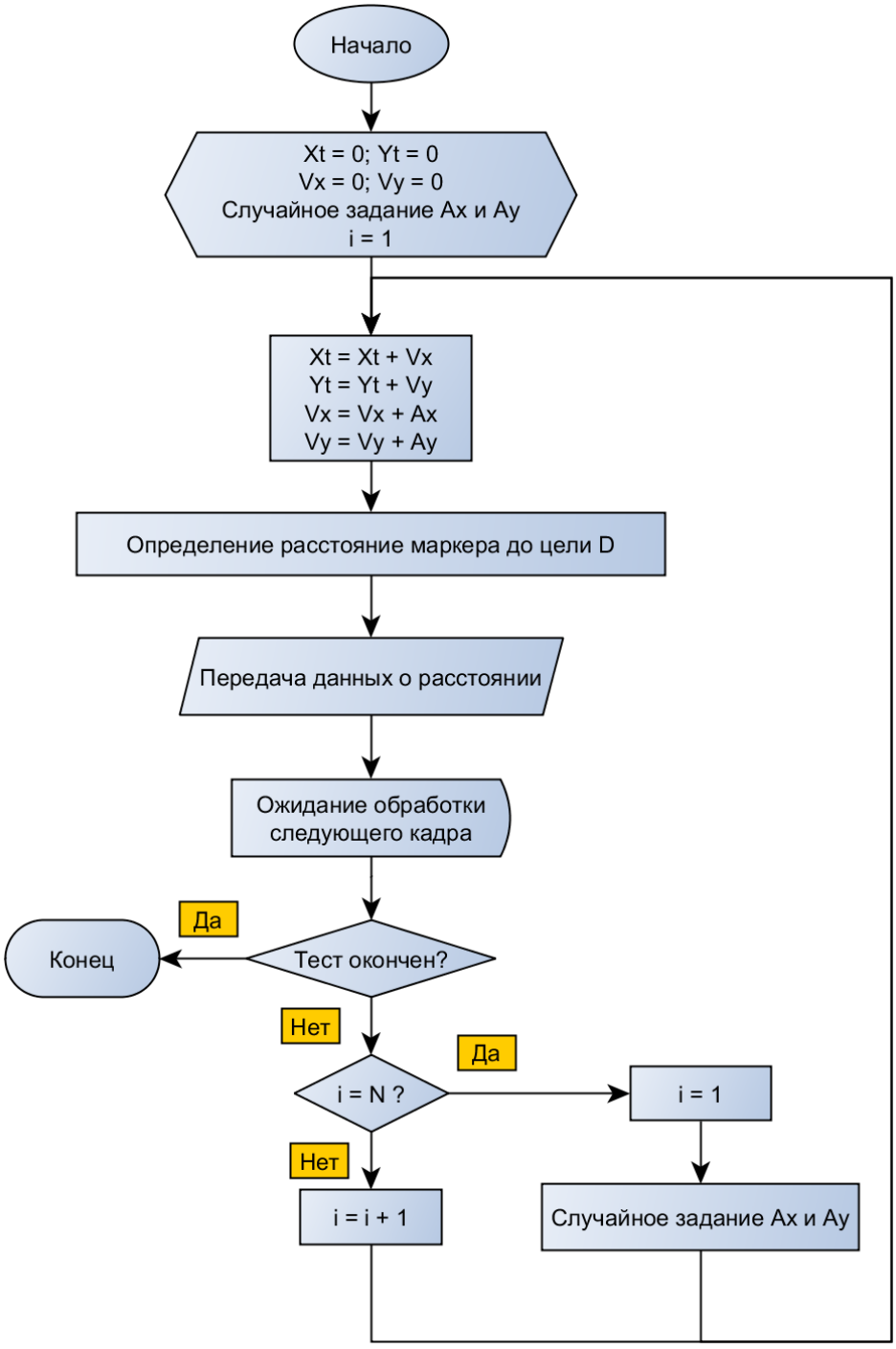


Рис. 40. Блок-схема алгоритма формирования эталонного сигнала для испытаний – вариант 2.

4.2.2. Осуществление бесконтактного управления.

После формирования координат цели одним из вышеизложенных способов, она выводится на экран, расположенный перед оператором, а ему предлагается

бесконтактным способом совместить маркер системы бесконтактного управления с целью. Испытания производились с участием нескольких операторов, а показатели работы системы усреднялись.

4.2.3. Фиксация результата.

В процессе испытания происходит сохранение различных числовых показателей с целью дальнейшего анализа. Такими показателями могут быть координаты маркера системы бесконтактного управления, координаты цели, и т.д. Так как система относится к «мягким» системам реального времени, для получения показателей в реальном времени необходимо также сохранять показатель времени. Сохранение происходит по завершении обработки каждого кадра.

4.2.4. Анализ результата.

С использованием записанных данных сформированы следующие показатели работы системы:

1. Среднее время обработки одного кадра. По этому показателю можно судить о быстродействии системы. Среднее время обработки одного кадра по результатам испытаний на персональном компьютере средней мощности составило 37 мс.

2. Скорость наведения на неподвижную цель. Вычисляется на основе испытаний с формированием координат цели первым способом. Так как скорость движения маркера на экранной плоскости зависит от удаления этой плоскости от оператора, целесообразно измерять угловую скорость наведения. Среднее значение по итогам испытаний составило 26°/с.

3. Точность наведения на неподвижную цель. Вычисляется на основе испытаний с формированием координат цели первым способом. Определяется на основе данных о расстоянии маркера до цели, после того, как это расстояние вошло в пределы допустимой точности (см. п. 4.2.1.). Среднее значение отклонения составило 4°10`.

4. Динамическая точность. Среднее значение отклонения между маркером и целью, получаемое при выполнении испытания с формированием координат цели вторым способом. Среднее значение отклонения составило 6°20`.

5. Среднеквадратичное отклонение, получаемое при следовании за подвижной целью. Другими словами, это среднеквадратичное отклонение получаемой при помощи системы траектории движения маркера от желаемой.

4.3. Основные наблюдения по результатам испытаний.

Таблица 1

Показатель	**По угловому положению головы**	**По направлению взгляда**	**Двухканальный подход**
Среднее время обработки одного кадра.	33 мс	37 мс	37 мс
Скорость наведения на неподвижную цель	14°/с	60°/с	28°/с
Среднее отклонение при наведении на неподвижную цель	9°	4°50`	4°10`
Среднее отклонение при следовании за подвижной целью	11°20`	35°	9°20`
СКО при следовании за подвижной целью	12°35`	42°	11°40`

Как видно из таблицы, наибольшее отклонение наблюдается при определении направления взгляда при слежении за подвижной целью. Это вызвано, в первую очередь, высоким количеством ошибок распознавания. Применение двухканального подхода позволяет существенно повысить точность за счёт отсеивания недостоверных результатов.

Также стоит отметить то, что введение двухканального подхода незначительно сказывается на быстродействии системы, так как применяемый в данной работе способ

определения направления взгляда в любом случае использует в качестве входных данных угловое положение головы оператора.

4.4. Особенности строения окуломоторной системы человека и их влияние на процесс управления бесконтактным способом.

При разработке системы бесконтактного управления необходимо уделить особое внимание характеру движений глаз оператора во время осуществления управления бесконтактным способом, так как они напрямую влияют на эффективность такого управления. Важную роль в формировании окуломоторных структур человека играют так называемые «саккады».

Саккады (от старинного французского слова, переводимого как «хлопок паруса») — быстрые, строго согласованные движения глаз, происходящие одновременно и в одном направлении. На записи имеют вид вертикальных прямых тонких линий. Специалисты нередко применяют термин «микросаккады» к быстрым движениям глаз, амплитуда которых не превышает 1 угл.град. А быстрые движения глаз амплитудой более 1 градуса называет «макросаккадами». С точки зрения В. А. Филина это деление чисто условное, так как предполагает, что эти два вида быстрых движений глаз имеют разный механизм происхождения. В настоящее время предполагается, что любые быстрые движения глаз имеют одну природу возникновения и поэтому их целесообразно называть одним словом — «саккада».

Автоматия саккад — это свойство глазодвигательного аппарата совершать быстрые движения глаз непроизвольно в определенном ритме. Саккады могут совершаться в бодрствующем состоянии при наличии зрительных объектов (в этом случае с помощью саккад происходит изменение точки фиксации взора, благодаря чему осуществляется рассматривание зрительного объекта), при отсутствии зрительных объектов, а также во время парадоксальной стадии сна (Филин В. А., 1987). Характер следования саккад обусловлен деятельностью центральной нервной системы, соответствующие структуры которой способны генерировать сигнал по типу автоматии, то есть способны к ритмогенезу. Каждому человеку присущ собственный паттерн следования саккад, который определяется тремя параметрами: интервалом между саккадами, их амплитудой и ориентацией. Наибольшее число саккад следует

через 0,2— 0,6 секунд, амплитуда саккад изменяется в большом диапазоне от 2 угл.мин до 15 угл.град, ориентированы саккады практически во всех направлениях (вправо, влево, вверх, вниз), но обычно их число больше в горизонтальной плоскости.

По сравнению с элементарными движениями глаз (саккадами, дрейфом, плавными прослеживаниями), окуломоторные структуры изучены довольно слабо. Не ясны механизмы их функциональной организации, не всегда понятен результирующий эффект разномерных детерминант, противоречивы данные о роли окуломоторных структур в процессе зрительного восприятия. Очевидно, что минуя уровень интеграции, знание об окуломоторной активности остается существенно неполным, а его практическое использование — очень ограниченным. Актуальность исследований подобного рода поддерживается живым интересом общества к Уникальному феномену биомеханики — направленности/движениям глаз, тесно связанному с процессами чувственного познания, деятельности и общения.

Понятие окуломоторной структуры восприятия обозначает три момента:

наличие устойчивых конфигураций (паттернов) движений глаз

представление в терминах целостного поведенческого акта

отнесенность как к объекту, так и к субъекту восприятия.

Первый — фиксирует интегративный уровень анализа окуломоторной активности, второй — ориентирует исследователя на поиск механизмов функциональной организации целенаправленных поворотов глаз, третий — указывает на внутреннюю связь зрительных и окуломоторных компонентов восприятия и вводит собственно психологический план анализа.

ЗАКЛЮЧЕНИЕ

1. На основе анализа существующих систем бесконтактного управления предложена двухканальная архитектура системы, с грубым каналом управления, использующим данные об угловом положении головы оператора и точным каналом, определяющим направление взгляда по изображению глаз оператора.
2. В качестве алгоритма поиска частей лица на изображении выбран метод Виолы-Джонса. Ограничение области поиска позволяет снизить среднее время выполнения алгоритма в экспериментальной реализации с 300 мс до 27 мс, а исключение из алгоритма проверок слабых классификаторов в процессе его работы позволило сократить среднее время выполнения до 24 мс.
3. Разработан алгоритм определения направления взгляда оператора, позволяющий избавиться от большей части ошибок распознавания за счёт введения дополнительных эвристических проверок, комплексирования данных от различных каналов и итеративной коррекции результата.
4. Работа представленных алгоритмов подтверждена испытаниями с использованием экспериментальной реализации системы. Полученные результаты: быстродействие системы достаточно для её работы с частотой 27 кадров в секунду, а среднее отклонение при наведении на неподвижную цель равно 4°10`, в то время как максимально достижимая точность составляет 3°20`.
5. Применение двухканального подхода позволяет повысить точность определения направления взгляда, в особенности при слежении за подвижной целью – в этом случае уменьшение среднего отклонения за счёт применения двухканального подхода составляет 2° и 25°40` для отдельных каналов.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Mark S. Nixon and Alberto S. Aguado. Feature Extraction and Image Processing.— Academic Press, 2008.— С.88

2. Визильтер, Ю.В.; Желтов, С.Ю.; Бондаренко, А.В. и др. Обработка и анализ изображений в задачах машинного зрения. М.: Физматкнига, 2010.

3. Кручинин А.Ю. Классификация систем распознавания образов реального времени: http://recog.ru/blog/theory/31.html. — 2011.

4. Петенёв Е. К., Пушилин С. В., Чемоданов В. Б. Сравнение эффективности различных способов обнаружения объектов на изображении в системе бесконтактного управления. – Современные технологии в задачах управления, автоматики и обработки информации. Сборник трудов XXIII Международного научно-технического семинара. 14–20 сентября 2014 г., Алушта. – М.: ИКД “Зерцало-М», 2014. – 208с.

5. John Canny. A Computational Approach to Edge Detection. IEEE Transactions on pattern analysis and machine intelligence, vol. PAMI-8, no. 6, November 1986

6. D.H. Ballard. Generalizing the Hough transform to detect arbitrary shapes. – Computer science department, University of Rochester, Rochester, NY 14627, USA

7. Петенёв Е. К., Пушилин С. В., Чемоданов В. Б. Комплексирование управляющих воздействий по разным каналам в многоканальной системе бесконтактного управления. – Научно-технический вестник Поволжья. №1 2014г. – Казань: Научно-технический вестник Поволжья, 2014. – 210 с. – с.134.

8. Филин В. А. Закономерности саккадической деятельности глазодвигательного аппарата // Автореф. дис. д-ра биол. наук, М.: 1987 б. 44 с.

9. Филин В. А. Автоматия саккад. М.: Изд-во МГУ. 2002. 240 с 113 илл.

10. Гиппенрейтер Ю. Б. Движения человеческого глаза / Ю. Б. Гиппенрейтер. — М.: Изд-во Моск. ун-та, 1978. — 256 с.

11. P. Viola and M.J. Jones, «Rapid Object Detection using a Boosted Cascade of Simple Features», proceedings IEEE Conf. on Computer Vision and Pattern Recognition (CVPR 2001), 2001

12. P. Viola and M.J. Jones, «Robust real-time face detection», International Journal of Computer Vision, vol. 57, no. 2, 2004., pp.137–154

13. Р.Гонсалес, Р.Вудс, «Цифровая обработка изображений», ISBN 5-94836-028-8, изд-во: Техносфера, Москва, 2005. – 1072 с.

14.Метод Виолы-Джонса (Viola-Jones) как основа для распознавания лиц // Habrahabr, 2 декабря 2011

15. Kinect for Windows SDK reference. www.microsoft.com

ПРИЛОЖЕНИЕ А РЕЗУЛЬТАТЫ ТЕСТИРОВАНИЯ ЭКСПЕРИМЕНТАЛЬНОГО ОБРАЗЦА СИСТЕМЫ.

А.1 Тестирование при наведении на неподвижную цель.

Таблица 2.

Порядковый номер	Время обработки кадра, мс			Скорость наведения на неподвижную цель, °/с			Отклонение при наведении на неподвижную цель, °		
	По угловому положению головы	По направлению взгляда	Двух канальный подход	По угловому положению головы	По направлению взгляда	Двухканальный подход	По угловому положению головы	По направлению взгляда	Двухканальный подход
1	32	41	35	15	48	26	9,159	4,544	3,511
2	35	42	35	16	62	30	8,241	3,552	4,501
3	35	32	33	14	66	31	8,114	4,614	3,898
4	30	34	35	14	63	30	7,466	5,608	4,501
5	30	41	34	12	53	31	8,245	4,759	4,169
6	35	38	42	14	67	26	9,389	5,383	3,285
7	36	33	41	13	47	27	8,121	4,209	4,829
8	31	39	40	12	61	27	10,572	3,897	5,059
9	29	40	34	14	62	25	10,49	4,221	5,115
10	32	40	35	15	47	30	8,566	5,147	5,08
11	29	32	32	12	46	31	9,646	4,724	4,479
12	29	40	39	16	50	24	10,486	3,701	4,196
13	35	32	33	12	73	27	10,22	4,247	3,841
14	32	33	38	16	45	27	8,925	5,574	5,04
15	31	33	39	15	51	28	7,852	4,489	3,653
16	32	38	36	16	65	26	10,312	5,256	3,957
17	30	36	32	13	70	28	8,797	5,014	3,685
18	30	34	38	15	60	26	7,585	3,439	3,508
19	34	42	39	15	62	25	8,581	3,882	4,155

20	31	40	35	15	46	27	9,268	5,317	4,79
21	34	36	36	15	66	25	9,176	4,621	4,848
22	29	39	37	13	65	30	7,363	5,282	3,736
23	31	38	33	13	61	32	8,565	3,65	4,286
24	35	35	34	12	47	27	7,639	3,851	3,201
25	35	40	42	15	72	29	8,373	4,069	4,84
26	32	37	41	15	74	24	8,775	4,497	4,236
27	31	32	38	12	64	26	9,391	3,505	3,232
28	33	40	34	14	72	29	10,403	5,233	5,129
29	36	42	41	16	68	28	8,238	5,066	4,019
30	32	40	40	15	68	29	10,114	4,138	3,608
31	34	33	37	15	74	30	7,441	4,605	4,691
32	36	32	34	14	68	29	7,651	4,633	4,102
33	31	38	40	15	57	28	9,352	4,392	4,965
34	34	42	37	12	67	26	8,165	5,546	3,311
35	34	40	42	16	75	32	9,741	3,893	4,498
36	34	42	41	12	56	32	10,448	3,367	3,485
37	33	42	42	12	47	28	9,763	5,088	5,219
38	30	32	36	12	64	32	9,162	5,113	3,857
39	35	32	41	15	61	30	7,585	5,193	3,137
40	34	39	32	16	75	26	9,721	3,78	4,217
41	30	37	34	16	66	27	8,179	4,049	3,129
42	30	34	37	14	49	32	7,865	5,524	4,302
43	36	36	36	16	60	31	7,794	3,399	4,933
44	35	39	41	13	69	28	10,435	5,046	3,636
45	31	40	37	12	51	27	10,123	4,059	3,686
46	35	41	35	14	48	28	9,295	4,032	4,449
47	31	32	37	15	75	24	9,68	4,717	3,635
48	31	36	41	13	50	28	9,459	3,655	4,845
49	29	40	37	13	52	26	8,87	4,855	5,11
50	33	33	32	13	45	24	7,874	5,581	4,512

51	32	33	36	14	60	27	9,576	5,218	4,795
52	29	34	35	12	71	27	7,485	3,862	4,635
53	29	41	33	14	55	24	9,508	5,27	4,4
54	30	40	38	13	74	24	10,399	3,714	4,067
55	31	41	42	14	56	32	7,65	5,532	5,029
56	30	42	37	13	60	24	9,866	4,832	4,701
57	34	35	32	13	63	28	9,621	4,969	3,116
58	35	33	33	13	56	25	8,516	4,723	4,83
59	30	36	38	12	53	30	10,476	3,352	3,34
60	33	36	34	15	60	27	9,547	3,869	4,745
61	36	33	36	12	48	24	9,923	5,56	3,737
62	34	40	38	12	73	32	10,501	5,689	3,517
63	34	33	35	12	74	32	8,367	3,547	5,146
64	32	35	37	16	53	31	9,507	5,592	4,589
65	30	36	40	12	67	32	8,357	4,128	5,073
66	29	40	39	13	63	26	8,888	3,482	3,829
67	34	42	38	15	47	31	9,806	4,518	4,208
68	35	38	34	12	68	25	9,075	4,291	3,809
69	31	32	35	13	74	25	10,351	4,799	5,055
70	36	36	41	15	61	25	7,787	5,458	5,182
71	31	37	37	16	48	27	7,324	4,391	3,417
72	29	39	41	13	53	29	8,233	4,192	4,256
73	32	34	42	13	75	25	7,412	3,447	3,545
74	33	33	41	12	68	30	8,113	5,204	3,604
75	36	39	42	15	59	25	8,338	4,429	3,491
76	34	36	34	15	67	30	9,499	5,288	4,775
77	32	37	37	12	62	30	7,44	3,96	3,363
78	29	33	34	14	56	30	7,775	5,336	5,176
79	30	40	37	15	46	26	8,275	3,317	3,43
80	36	41	33	14	46	30	9,61	3,939	3,547
81	30	40	39	12	45	28	7,39	5,155	3,383

82	34	39	38	12	75	26	8,237	4,9	5,071
83	36	42	42	15	48	25	8,589	4,575	4,487
84	32	32	41	13	67	30	8,553	4,767	4,583
85	29	38	33	15	75	24	9,803	3,989	4,343
86	36	42	40	16	49	28	8,287	4,976	3,433
87	30	42	38	13	52	29	9,092	5,08	3,828
88	29	37	40	14	70	30	10,499	4,301	4,641
89	33	41	38	14	49	30	10,567	3,623	3,384
90	30	39	40	13	56	31	10,141	4,018	3,763
91	34	34	34	12	71	30	7,869	3,703	4,391
92	29	35	32	13	55	27	10,617	4,182	5,093
93	29	34	37	15	62	26	9,612	5,579	3,885
94	35	39	42	13	57	27	10,296	4,817	3,511
95	30	32	36	13	46	31	10,498	4,262	3,934
96	31	40	33	14	63	30	10,587	3,608	4,647
97	29	42	41	16	45	26	7,445	5,023	4,821
98	33	40	36	13	51	25	8,902	4,024	5,216
99	30	38	37	15	67	27	10,193	4,435	3,959
100	31	36	39	15	56	24	10,697	4,798	4,397
101	31	39	37	15	58	29	9,99	4,201	5,012
102	36	42	32	15	59	24	9,964	4,964	3,363
103	35	42	42	16	68	27	8,184	3,821	3,803
104	36	41	37	15	63	29	10,004	4,316	4,427
105	30	38	37	12	60	29	8,659	5,177	3,589
106	30	37	40	13	47	30	8,512	4,116	4,561
107	31	36	36	12	72	28	7,844	5,206	4,004
108	35	39	36	14	55	31	10,325	4,843	4,806
109	33	32	35	13	60	25	9,32	4,176	3,928
110	32	39	39	12	75	28	7,89	5,544	4,254
111	29	40	34	13	64	32	9,285	5,032	4,206
112	34	39	37	14	73	28	10,484	3,99	3,646

113	32	38	32	12	49	25	8,024	5,469	4,451
114	33	42	38	15	45	32	8,304	4,601	3,716
115	29	37	38	13	62	25	8,681	5,449	3,278
116	29	34	36	16	55	24	8,706	4,209	3,207
117	29	41	35	16	52	25	7,64	4,295	4,699
118	35	38	38	15	72	24	9,385	3,68	4,613
119	32	41	33	15	48	29	8,403	4,732	5,132
120	32	33	32	13	51	29	7,839	5,013	4,805
121	33	38	42	13	46	32	9,89	3,385	4,01
122	35	32	36	14	68	30	7,779	4,934	4,26
123	31	41	32	16	47	27	9,699	5,197	3,184
124	33	40	34	12	54	32	7,572	5,58	5,159
125	31	33	40	13	57	24	8,869	3,318	3,478
126	31	34	34	15	50	29	9,01	3,319	3,302
127	35	37	42	15	52	28	8,493	4,923	4,399
128	33	33	38	14	65	24	8,29	4,814	3,308
129	30	34	41	15	51	27	10,386	4,143	4,313
130	30	34	39	14	56	26	10,239	3,594	4,897
131	32	34	38	16	46	26	8,284	3,47	3,15
132	29	36	35	16	71	24	8,617	4,065	3,594
133	36	42	37	14	56	29	7,886	3,693	4,745
134	29	36	36	14	45	32	9,072	4,325	3,827
135	34	35	33	12	45	26	9,98	5,472	3,913
136	32	39	34	15	53	26	9,499	4,977	4,661
137	35	37	34	16	67	28	8,484	5,331	3,427
138	35	41	34	13	72	25	10,123	4,446	5,18
139	35	34	38	14	46	29	9,748	5,133	5,038
140	32	36	41	14	68	29	7,341	5,385	5,238
141	36	41	40	13	61	29	7,691	4,593	4,762
142	34	37	37	15	55	27	7,574	5,401	4,511
143	31	34	41	14	57	30	10,642	4,578	4,567

144	36	42	42	13	73	26	8,605	3,304	4,791
145	29	34	41	14	73	25	9,888	4,393	4,681
146	34	38	34	15	49	25	7,818	3,521	3,729
147	36	40	34	16	57	31	10,348	4,557	3,842
148	31	41	40	16	72	32	10,241	4,412	4,252
149	30	35	37	14	46	26	10,385	5,255	4,24
150	36	32	36	14	72	32	9,671	4,509	4,897
151	30	36	36	15	48	25	7,317	5,638	3,307
152	32	36	36	13	68	26	8,871	4,348	4,13
153	32	33	40	12	66	27	10,312	5,494	3,711
154	34	35	41	16	52	28	8,42	4,661	3,284
155	31	32	39	14	53	30	10,074	3,404	3,989
156	35	41	37	14	59	30	9,975	3,975	5,167
157	35	42	37	13	69	27	7,475	3,971	3,807
158	29	35	35	14	70	31	7,933	4,211	4,02
159	34	41	38	14	50	32	8,107	4,582	4,276
160	35	34	38	15	47	29	8,743	3,835	4,632
161	29	39	36	16	69	31	10,683	4,857	5,16
162	29	33	39	15	57	28	10,468	4,257	4,396
163	34	41	34	15	66	28	10,617	5,681	4,671
164	31	32	34	12	68	30	10,576	4,757	4,171
165	31	40	32	14	65	30	10,469	3,794	5,109
166	30	42	40	16	57	27	8,532	4,007	3,673
167	29	39	39	13	49	29	10,411	3,328	3,426
168	35	35	33	13	75	24	8,922	3,758	4,103
169	31	37	32	16	53	31	8,891	5,487	4,513
170	36	34	36	12	59	31	10,342	3,784	3,574
171	34	42	37	16	66	25	9,298	5,145	4,028
172	33	40	42	16	60	25	8,958	5,468	5,067
173	35	38	36	14	56	24	7,55	4,214	4,054
174	31	39	38	16	54	29	9,102	4,031	3,368

175	30	36	38	12	48	29	10,686	4,15	3,577
176	35	38	32	14	49	27	10,507	4,074	4,685
177	29	39	36	14	72	25	10,647	3,854	3,621
178	36	39	35	13	67	30	7,878	5,246	5,148
179	36	41	35	12	58	28	9,958	3,755	3,751
180	31	33	33	13	75	29	8,899	5,209	4,226
181	34	41	37	14	49	24	9,984	5,554	3,113
182	34	42	40	15	50	28	9,073	3,746	4,248
183	36	41	35	15	48	25	9,844	5,213	4,518
184	30	36	42	16	71	24	9,697	3,973	3,508
185	31	40	42	13	61	24	9,779	4,994	3,785
186	29	33	34	16	61	28	9,56	5,436	4,043
187	35	36	35	14	53	30	8,861	4,194	3,492
188	36	33	32	16	58	24	8,495	4,455	4,163
189	36	34	38	14	69	31	8,319	5,417	4,988
190	35	39	36	13	65	27	7,556	3,465	4,105
191	34	33	35	13	53	30	8,041	3,934	3,552
192	32	42	33	12	51	27	10,121	4,458	4,088
193	30	37	34	15	63	28	8,748	4,395	5
194	32	35	39	16	67	26	7,471	4,189	3,211
195	36	32	38	16	66	29	9,078	5,564	3,199
196	31	35	35	15	69	31	8,041	3,411	3,216
197	35	33	40	14	53	25	8,097	5,046	4,178
198	30	37	39	13	51	28	7,772	3,656	3,351
199	31	38	38	14	71	26	9,414	3,692	4,362
200	34	35	35	12	62	27	8,621	4,558	4,526
201	35	36	38	14	53	28	7,498	5,044	5,029
202	29	38	40	15	49	26	10,644	4,259	3,764
203	34	38	34	16	47	27	8,721	3,627	3,263
204	31	32	33	14	75	30	9,317	3,873	4,76
205	29	36	39	16	69	27	10,43	5,295	4,41

206	34	40	35	15	65	26	8,228	5,697	4,078
207	29	42	33	16	51	27	8,785	4,197	4,213
208	29	42	35	16	62	24	8,432	3,61	4,68
209	36	42	37	15	53	30	9,575	5,56	3,971
210	30	35	34	14	61	32	8,067	3,504	4,347
211	33	37	35	16	52	32	9,985	3,722	3,315
212	34	39	34	15	66	29	7,652	3,736	4,108
213	34	38	40	14	60	25	9,341	4,665	3,946
214	33	40	38	12	46	29	7,481	5,136	4,124
215	35	40	36	16	63	24	8,026	3,793	4,71
216	31	37	33	14	71	25	8,53	5,257	5,041
217	30	34	32	13	63	26	9,507	4,755	3,704
218	30	32	32	13	49	30	8,724	5,12	3,743
219	31	34	38	13	73	24	9,6	4,573	3,874
220	29	35	40	13	65	24	7,336	4,263	4,084
221	34	32	34	16	73	27	8,297	4,986	4,044
222	35	38	41	13	68	24	8,25	4,911	4,173
223	33	38	39	12	49	32	10,063	4,556	4,271
224	29	34	35	12	63	31	9,86	5,585	5,099
225	30	37	41	16	67	32	8,833	3,649	3,363
226	32	36	36	12	45	32	7,873	3,396	5,013
227	36	36	36	14	60	32	8,998	5,691	4,639
228	36	40	36	15	58	26	7,937	3,342	4,37
229	31	41	37	14	45	25	7,699	3,479	4,597
230	33	41	39	15	47	26	8,977	5,191	5,194
231	32	34	34	16	69	25	8,684	4,847	3,835
232	31	35	40	13	75	26	8,233	4,116	5,009
233	32	42	38	12	58	28	9,663	4,326	4,915
234	29	33	32	16	64	26	7,531	3,694	3,188
235	34	35	35	16	59	27	8,41	4,327	4,388
236	34	39	35	13	49	27	9,278	4,271	3,96

237	35	38	39	13	51	27	8,466	5,177	4,812
238	34	39	41	13	61	31	9,949	5,479	4,834
239	34	32	40	16	46	26	7,524	3,94	3,298
240	34	39	40	13	62	29	7,7	4,984	4,302
241	34	36	32	13	58	30	10,27	5,544	3,092
242	32	33	40	15	50	28	8,147	4,841	4,509
243	32	32	34	14	66	28	8,603	4,514	3,377
244	29	35	34	16	70	25	10,48	4,465	5,044
245	36	37	41	16	45	28	7,632	3,477	3,102
246	36	33	37	13	68	25	7,383	4,088	4,477
247	30	35	32	15	62	26	9,55	5,455	3,991
248	29	39	42	14	60	27	9,278	4,03	3,402
249	34	36	38	16	54	25	9,942	3,804	3,988
250	29	41	33	14	55	26	8,298	3,332	3,941
251	30	35	33	13	65	29	8,257	5,092	3,744
252	35	39	41	15	58	24	7,891	5,595	3,716
253	33	39	33	15	71	24	9,188	5,359	4,795
254	31	36	36	15	61	29	9,619	4,391	3,351
255	29	34	34	12	66	32	9,685	4,368	3,99
256	30	32	34	15	64	31	7,875	4,55	3,582
257	33	40	36	16	64	31	9,378	4,259	3,467
258	36	36	42	13	72	31	7,351	4,123	3,898
259	33	36	41	15	51	28	9,54	4,64	4,565
260	35	32	36	12	74	24	10,421	4,352	3,396
261	29	32	33	14	66	29	7,653	4,184	5,238
262	35	35	36	13	69	25	9,381	3,993	3,498
263	34	33	42	16	66	32	8,502	3,515	5,129
264	35	36	33	13	49	27	10,106	5,647	3,783
265	30	39	34	15	74	31	10,69	4,205	3,931
266	29	42	32	14	68	24	8,478	3,476	3,805
267	31	39	36	15	72	25	7,519	4,868	3,211

268	29	36	35	14	64	26	7,574	4,883	4,086
269	34	34	33	15	70	27	8,693	3,36	4,838
270	32	32	33	14	62	29	8,147	5,424	3,5
271	35	34	40	13	55	27	8,114	4,56	3,771
272	31	39	32	14	63	32	8,125	4,228	4,068
273	30	36	36	15	57	30	7,465	5,289	4,623
274	32	40	38	14	63	29	10,559	4,165	3,318
275	35	36	34	14	64	25	8,825	5,332	5,196
276	35	32	39	15	71	29	7,671	3,502	5,044
277	34	33	38	12	52	32	10,069	3,86	3,369
278	31	34	36	16	74	29	9,087	3,994	3,293
279	35	32	38	13	62	32	8,636	4,717	5,161
280	36	36	39	12	60	24	10,489	4,143	5,135
281	36	34	37	16	59	26	9,852	3,552	3,444
282	32	39	35	12	45	25	9,836	4,414	4,859
283	35	37	39	13	48	30	7,531	3,31	3,882
284	29	39	35	13	52	31	10,064	4,74	4,203
285	29	34	41	15	51	27	10,609	4	4,623
286	33	39	36	16	56	25	8,138	3,406	4,049
287	30	42	35	14	54	25	9,695	4,894	3,82
288	35	39	42	13	48	30	9,299	4,723	4,83
289	29	37	39	13	45	24	10,234	3,464	5,075
290	36	39	39	13	59	24	9,623	3,904	3,305
291	33	33	42	13	51	30	10,206	3,484	3,645
292	34	41	32	14	74	29	7,555	3,565	3,812
293	34	35	38	12	66	27	9,851	4,347	5,211
294	30	36	38	15	74	24	9,17	5,599	4,703
295	36	39	34	12	65	24	7,356	4,541	4,687
296	32	35	33	12	51	26	10,69	4,679	4,713
297	31	42	37	14	50	30	8,9	5,581	4,697
298	33	40	39	12	53	24	9,659	4,99	5,133

299	30	40	42	15	68	29	10,666	5,503	3,75
300	33	34	41	14	63	28	7,439	3,733	4,903
301	30	35	35	16	71	29	8,784	4,287	5,093
302	33	42	39	12	53	29	7,898	5,111	4,559
303	36	41	32	13	64	25	7,732	4,516	3,352
304	35	37	33	13	55	28	7,343	5,038	5,122
305	34	38	34	14	67	31	10,467	3,847	4,306
306	29	33	39	12	54	30	9,889	4,316	3,661
307	30	34	34	14	47	29	9,742	4,95	4,914
308	31	39	32	12	69	32	8,373	3,362	3,244
309	33	37	42	12	62	30	8,164	5,44	3,518
310	31	36	42	12	51	30	9,692	3,598	4,124
311	29	41	41	15	70	27	10,227	3,758	3,102
312	33	37	33	12	68	28	9,235	5,146	4,105
313	32	32	41	12	56	31	10,543	3,988	4,948
314	29	35	39	12	57	26	8,074	4,381	3,418
315	31	40	36	14	47	31	7,39	4,744	4,758
316	35	33	36	15	57	32	8,41	3,731	5,097
317	31	33	33	15	45	28	7,391	4,359	4,376
318	29	38	39	12	55	28	8,345	5,406	4,983
319	31	36	41	16	62	30	8,353	5,369	4,565
320	34	42	40	15	60	28	8,864	3,315	5,123
321	31	42	40	15	73	24	10,661	3,38	4,148
322	33	41	37	16	60	28	9,762	5,115	3,885
323	36	35	40	16	69	30	8,549	3,877	3,704
324	32	39	37	13	49	29	8,366	3,673	4,857
325	34	39	40	14	48	30	8,154	3,783	4,459
326	31	38	33	16	48	28	7,387	3,816	3,726
327	31	33	37	14	61	25	9,567	3,89	4,597
328	34	33	42	13	54	30	8,141	4,225	3,357
329	32	41	34	16	64	29	7,596	4,93	4,79

330	36	34	38	16	62	25	8,653	4,822	5,06
331	34	37	35	14	60	28	9,2	4,847	4,859
332	32	40	32	16	64	25	10,152	4,775	4,08
333	33	37	42	16	56	31	10,302	4,92	4,105
334	35	41	39	12	51	32	7,459	4,713	3,391
335	35	42	34	16	58	26	9,023	5,422	3,947
336	31	39	36	16	68	27	10,666	4,661	3,69
337	36	41	35	16	60	25	9,324	4,449	3,415
338	30	40	32	15	64	30	8,005	3,395	3,547
339	34	34	42	12	60	32	8,214	3,761	4,015
340	36	35	40	13	55	25	9,484	3,479	4,585
341	32	42	42	13	65	24	9,072	5,571	4,795
342	32	36	32	13	72	28	8,851	3,567	5,065
343	33	40	34	16	64	30	10,517	4,844	3,341
344	35	40	38	16	56	28	8,462	4,384	4,677
345	34	35	37	13	64	25	9,708	5,07	4,279
346	34	37	32	16	50	26	7,973	4,413	4,231
347	29	38	40	15	55	32	8,097	5,228	5,092
348	35	36	42	14	66	26	8,343	4,596	4,776
349	33	38	35	16	54	27	10,562	4,823	4,402
350	34	35	38	13	68	27	9,514	5,005	3,158
351	34	34	33	13	46	31	8,473	4,512	4,535
352	31	38	35	15	52	25	9,814	4,275	4,541
353	29	34	41	15	63	29	9,438	4,794	3,224
354	32	33	36	15	45	30	9,533	4,049	4,564
355	31	39	37	15	63	24	8,615	5,626	5,025
356	31	34	38	15	47	26	9,381	4,407	3,249
357	30	39	32	12	60	29	8,936	5,415	4,35
358	35	37	41	15	49	24	8,672	3,667	3,571
359	29	32	34	12	74	28	7,913	4,092	5,189
360	32	32	41	13	67	28	9,723	5,209	4,349

361	32	37	38	13	53	26	10,25	4,962	3,662
362	35	35	34	14	54	32	9,744	3,427	3,588
363	35	37	36	13	71	31	8,23	5,534	3,909
364	29	36	41	13	48	31	8,146	5,363	3,198
365	35	42	39	13	57	26	8,569	3,756	3,419
366	35	42	42	12	46	26	8,282	3,619	3,642
367	31	35	33	15	56	26	10,052	5,025	4,43
368	29	32	38	14	45	32	8,37	4,016	3,809
369	29	36	33	13	53	26	9,764	4,413	4,71
370	30	32	33	16	68	32	8,02	3,9	3,913
371	35	37	34	15	62	28	8,915	3,519	3,718
372	33	38	38	16	66	24	8,428	4,526	3,219
373	36	38	37	12	54	25	7,947	5,303	4,174
374	33	35	39	16	48	28	10,142	5,316	3,101
375	36	35	33	16	55	28	8,03	4,687	5,142
376	31	33	33	16	51	32	8,15	3,425	3,432
377	36	42	37	16	52	29	10,242	4,181	4,678
378	32	34	40	15	61	31	7,392	3,969	4,308
379	36	37	38	12	52	32	9,67	5,424	3,602
380	35	41	32	13	53	31	8,1	3,522	3,97
381	33	41	36	15	64	29	8,078	4,76	4,119
382	30	36	34	15	65	27	9,427	5,005	3,316
383	32	33	36	14	45	30	7,81	3,416	3,293
384	31	36	34	13	53	27	7,397	4,504	3,673
385	30	34	34	16	59	28	10,583	5,641	3,856
386	33	39	33	12	64	27	9,446	3,582	4,635
387	35	39	39	12	66	31	7,514	5,698	3,823
388	32	34	33	14	60	25	10,153	5,551	4,48
389	33	37	41	16	70	24	9,033	5,175	4,359
390	29	34	36	14	63	29	8,941	5,568	4,313
391	29	36	33	16	69	26	10,444	3,441	4,802

392	30	32	36	14	56	28	8,062	4,401	5,09
393	35	38	42	12	66	26	7,388	4,872	3,883
394	34	33	39	12	49	32	10,514	4,801	4,68
395	35	35	40	15	64	26	8,536	3,82	4,626
396	29	41	36	13	54	27	7,332	4,331	4,811
397	34	34	35	14	53	27	9,992	3,469	3,156
398	31	36	35	14	47	31	7,879	5,258	5,104
399	31	39	41	13	59	27	10,586	4,384	4,547
400	31	36	34	13	58	32	8,072	4,853	3,664
401	33	33	39	16	62	26	10,573	4,26	4,016
402	30	38	37	13	48	26	8,67	4,352	3,649
403	33	36	36	15	73	29	10,675	5,539	4,708
404	32	40	33	13	46	26	8,693	3,773	4,2
405	32	32	34	13	70	27	9,782	4,962	4,169
406	33	42	41	16	75	28	7,998	4,981	3,828
407	32	39	35	12	74	29	8,467	4,625	4,662
408	36	32	41	13	49	26	10,344	3,425	3,408
409	33	35	41	12	55	28	8,35	4,669	3,676
410	36	32	32	16	66	31	9,781	4,782	3,268
411	36	35	37	13	51	24	10,155	4,676	4,555
412	33	34	41	14	70	25	9,332	4,182	4,139
413	36	39	35	12	47	27	10,081	4,205	3,889
414	34	41	32	16	64	31	9,689	3,303	4,942
415	30	33	33	14	70	27	8,556	5,311	4,733
416	35	36	42	16	52	29	7,489	3,857	5,238
417	30	39	32	13	74	27	9,035	3,496	3,312
418	29	32	34	14	68	26	7,595	4,769	4,112
419	36	41	39	15	47	29	7,38	3,695	4,574
420	29	42	41	13	71	29	10,584	5,496	3,407
421	29	40	36	15	69	28	9,88	4,313	4,177
422	36	39	39	14	73	31	10,586	4,371	4,757

423	31	35	36	16	72	26	10,618	5,343	3,554
424	35	41	40	14	51	31	10,149	4,322	5,147
425	30	38	33	15	74	32	9,835	4,949	4,102
426	32	40	35	15	72	26	8,08	3,574	3,374
427	34	36	33	13	51	24	8,843	3,424	3,717
428	33	34	42	15	64	28	8,024	3,933	3,401
429	34	38	41	15	57	26	10,39	5,221	3,145
430	33	37	37	14	47	24	9,133	5,192	4,545
431	29	37	42	14	50	29	8,059	3,513	3,235
432	31	32	39	13	68	27	9,666	5,25	4,226
433	31	32	36	15	48	29	9,05	5,048	5,017
434	30	37	34	16	73	28	7,704	5,571	4,496
435	31	39	35	15	62	32	10,034	5,238	3,449
436	36	33	33	12	62	32	8,9	5,613	3,785
437	31	33	41	13	68	31	10,076	5,641	5,206
438	33	39	34	12	68	24	9,831	5,063	3,228
439	30	37	38	15	58	27	10,133	5,52	4,072
440	34	36	34	16	54	28	10,066	4,29	3,828
441	32	37	34	14	62	28	8,757	4,117	3,229
442	32	39	34	13	61	30	8,727	5,008	4,059
443	29	32	32	15	60	27	9,027	4,676	3,213
444	29	32	40	15	58	31	9,708	5,134	5,206
445	35	42	39	15	66	25	9,151	5,229	3,712
446	31	37	40	12	61	26	8,804	3,309	4,654
447	29	33	34	16	74	26	7,334	3,332	5,155
448	31	41	41	14	60	28	7,833	5,309	3,941
449	30	37	42	15	55	28	8,485	5,082	3,055
450	36	33	40	14	54	25	9,059	5,414	4,093
451	34	38	37	12	73	28	7,687	4,573	4,217
452	33	33	35	15	47	24	8,312	3,844	3,311
453	31	37	42	14	55	32	9,584	4,65	4,324

454	34	36	32	16	65	28	8,163	5,661	3,309
455	32	40	42	16	49	32	10,618	4,575	5,138
456	32	40	39	16	53	24	7,628	5,322	3,378
457	30	41	41	13	58	27	9,17	4,665	5,176
458	31	37	32	16	71	24	9,937	4,353	3,674
459	32	35	36	14	55	27	9,628	3,501	4,218
460	36	37	40	15	58	28	10,121	5,517	3,254
461	29	38	35	16	61	24	9,607	4,329	3,708
462	32	36	33	15	72	30	9,063	4,684	4,454
463	30	32	35	13	48	31	7,932	3,368	3,438
464	29	38	38	14	72	28	8,988	3,874	3,156
465	33	32	37	14	75	27	8,402	4,665	5,182
466	29	38	33	14	73	25	9,107	5,453	3,251
467	32	35	37	12	54	26	8,032	4,042	4,119
468	29	32	40	12	51	27	7,778	4,126	3,21
469	36	33	35	12	58	25	9,272	5,584	3,734
470	29	42	37	13	67	25	9,555	5,493	3,432
471	35	37	42	14	69	32	8,076	5,587	5,071
472	33	38	39	12	54	31	8,739	5,144	3,506
473	29	37	42	12	54	30	8,012	5,17	4,401
474	34	37	36	13	46	24	9,208	4,419	4,349
475	36	38	37	14	55	25	9,448	4,366	4,995
476	36	37	36	16	68	30	9,182	4,825	3,431
477	31	33	35	14	58	29	8,79	4,506	3,51
478	35	34	40	16	48	24	9,18	4,184	4,694
479	31	41	40	13	64	32	9,455	4,031	3,648
480	35	37	40	16	69	31	8,59	5,69	4,907
481	36	35	38	13	70	28	8,951	4,635	4,938
482	33	33	38	14	72	27	10,696	3,868	3,632
483	31	33	34	13	45	27	10,371	4,88	3,569
484	31	36	38	14	67	29	10,696	4,697	3,741

485	32	41	32	12	59	24	10,059	4,957	3,569
486	34	34	39	15	57	24	9,881	3,859	3,425
487	34	34	36	15	73	31	7,315	5,137	3,392
488	31	42	37	16	64	25	7,746	4,326	4,044
489	33	40	38	12	52	24	8,762	4,873	4,582
490	36	33	34	16	66	32	9,488	3,708	3,229
491	36	35	34	16	45	25	10,499	3,46	4,86
492	32	38	38	15	56	26	7,37	5,413	4,913
493	31	35	32	15	69	25	7,61	4,105	3,331
494	30	37	32	15	68	31	8,361	3,787	5,002
495	30	36	39	13	62	32	8,994	3,841	4,965
496	30	34	39	13	66	24	10,118	5,293	5,228
497	34	39	33	15	68	30	7,862	3,865	5,097
498	36	42	32	16	50	25	8,248	4,468	4,451
499	36	41	41	13	72	32	7,352	3,376	3,822
500	29	38	38	12	46	29	10,345	4,584	4,251

А.2. Тестирование при слежении за подвижной целью.

Таблица 2

Порядковый номер	Время обработки кадра, мс			Отклонение при следовании за подвижной целью, °		
	По угловому положению головы	По направлению взгляда	Двухканальный подход	По угловому положению головы	По направлению взгляда	Двухканальный подход
1	34	40	38	11,676	5,538	9,256
2	31	41	38	10,541	5,94	7,98
3	36	32	41	11,983	5,339	9,409
4	34	39	40	11,903	6,463	10,174
5	35	36	40	10,047	5,301	9,403
6	33	38	41	9,777	6,527	10,832
7	34	39	40	10,385	6,743	7,988
8	32	37	39	11,35	6,382	9,141
9	33	38	40	10,045	5,702	10,531
10	34	40	34	11,961	6,572	7,749
11	29	37	33	10,07	5,526	9,183
12	30	33	32	10,142	61,152	7,945
13	33	35	36	11,695	5,226	8,503
14	33	34	35	10,383	5,651	10,449
15	34	33	33	9,982	7,072	9,757
16	34	37	42	10,724	5,478	7,946
17	33	36	40	10,629	6,654	8,779
18	33	42	38	10,027	6,829	9,623
19	34	42	36	9,675	6,942	7,611
20	35	41	40	12,221	6,509	10,378
21	33	33	34	10,192	6,379	8,416

22	36	41	35	11,159	5,373	8,163
23	34	38	40	12,831	6,905	8,935
24	31	41	34	9,966	7,177	7,717
25	31	36	41	10,088	7,051	9,028
26	36	42	36	10,561	6,171	9,397
27	31	35	39	12,722	7,198	9,655
28	36	41	34	9,974	6,758	9,149
29	29	41	37	12,257	6,054	8,595
30	35	33	34	11,039	6,721	8,499
31	34	33	37	10,723	6,063	8,741
32	32	34	37	11,869	7,289	8,358
33	36	35	42	11,071	7,189	8,828
34	29	33	32	10,088	5,387	8,404
35	30	38	38	11,088	6,104	9,844
36	32	37	37	11,779	5,342	8,614
37	34	33	35	12,112	6,087	7,643
38	29	32	38	11,729	6,845	8,558
39	32	37	33	9,683	5,714	8,357
40	33	39	41	10,669	6,285	9,865
41	36	32	37	11,228	5,622	8,352
42	36	36	34	11,206	6,267	9,884
43	29	39	33	11,551	6,817	9,173
44	36	32	34	12,289	5,783	9,834
45	34	41	34	12,849	6,915	9,619
46	35	36	32	12,839	5,217	9,09
47	33	42	32	12,352	6,877	9,12
48	36	42	36	11,149	7,012	9,986
49	30	34	40	12,487	7,231	10,994
50	36	41	33	10,934	62,383	9,494
51	32	36	33	10,555	7,113	9,513
52	29	33	38	9,82	5,57	10,28

53	30	39	32	9,763	6,869	10,449
54	36	42	42	9,731	5,366	7,717
55	35	38	38	11,088	5,372	9,879
56	35	32	37	10,803	5,711	9,841
57	30	38	34	12,692	6,114	9,766
58	32	34	37	12,758	5,421	9,165
59	31	41	36	12,564	6,025	7,807
60	30	40	42	10,675	5,888	10,082
61	32	36	40	11,278	6,418	10,047
62	32	35	35	10,739	6,859	8,031
63	35	40	41	10,891	5,699	7,817
64	31	33	42	11,534	7,202	9,389
65	33	39	34	12,706	6,03	7,886
66	32	37	41	11,852	6,004	8,134
67	33	37	37	12,83	7,388	10,286
68	29	33	32	12,645	6,371	8,402
69	29	40	41	10,925	7,01	9,447
70	36	41	42	11,75	5,695	7,76
71	29	34	35	11,52	7,136	7,82
72	35	39	41	12,476	7,328	9,996
73	30	36	42	12,234	5,263	8,42
74	30	33	34	11,853	5,231	10,036
75	29	42	34	12,189	6,481	9,474
76	32	42	36	12,63	6,54	8,044
77	33	38	34	10,536	6,743	9,098
78	30	40	33	10,812	7,365	9,085
79	31	42	41	10,339	6,469	9,338
80	35	35	32	11,14	6,975	8,965
81	33	33	36	11,957	5,283	10,3
82	35	40	36	11,245	5,535	8,267
83	35	40	36	10,413	7,313	10,555

84	33	33	40	9,889	7,311	9,756
85	35	33	33	12,407	7,385	10,039
86	34	42	42	12,382	6,022	9,977
87	30	32	32	11,784	5,667	7,696
88	29	38	40	12,311	6,191	9,657
89	35	42	42	10,849	5,533	8,925
90	34	39	34	10,136	6,658	9,41
91	33	37	40	11,108	6,809	9,174
92	36	33	38	9,898	6,839	7,644
93	33	33	36	11,487	5,794	9,973
94	29	32	38	12,676	5,502	8,168
95	31	34	38	12,397	6,553	9,523
96	33	38	36	11,839	6,377	10,764
97	31	33	40	10,751	43,15	7,621
98	33	35	40	11,371	6,595	9,34
99	30	37	38	12,231	6,715	8,61
100	34	36	37	10,125	6,991	10,914
101	36	36	36	11,952	5,366	10,252
102	34	41	40	12,77	5,479	8,447
103	35	34	34	9,655	6,156	10,867
104	32	38	33	12,016	43,953	10,33
105	30	39	40	12,06	7,26	10,529
106	36	37	38	12,155	5,775	9,974
107	33	35	38	10,436	7,399	10,555
108	29	33	38	10,236	6,53	9,952
109	35	32	33	10,863	6,339	10,128
110	30	42	36	9,87	5,89	9,312
111	33	34	32	12,097	5,617	9,721
112	33	41	34	12,411	7,287	7,634
113	30	42	36	12,252	5,494	10,581
114	31	38	32	10,698	7,103	7,928

115	33	32	36	12,025	5,979	10,105
116	35	34	35	10,336	5,691	10,909
117	36	41	39	12,238	7,177	8,421
118	32	35	36	9,785	7,144	10,748
119	31	39	40	11,502	61,322	10,449
120	31	33	41	9,606	5,876	7,737
121	35	32	42	9,668	6,219	10,363
122	35	37	42	11,558	7,298	8,403
123	30	35	39	11,68	6,064	10,695
124	33	38	32	11,855	7,041	8,731
125	34	39	33	10,708	5,934	8,267
126	34	37	40	10,55	6,692	9,326
127	29	42	36	12,506	6,605	10,77
128	35	40	40	11,952	6,755	8,541
129	36	37	32	11,126	6,774	9,247
130	35	41	36	10,064	6,368	9,763
131	30	39	39	10,631	5,887	10,963
132	29	37	32	11,976	5,417	8,932
133	32	41	34	11,486	5,595	8,151
134	29	32	41	11,385	6,129	9,312
135	34	32	37	10,421	6,051	8,352
136	31	41	41	10,038	7,307	9,786
137	32	36	33	11,741	6,814	8,107
138	34	33	32	12,717	6,643	9,254
139	29	36	34	9,667	7,221	9,95
140	35	40	32	11,488	6,431	9,518
141	33	34	35	12,592	6,258	9,933
142	32	33	41	10,846	5,499	10,46
143	31	41	33	10,734	6,657	10,531
144	30	41	32	12,49	45,831	10,644
145	36	42	38	10,291	5,45	8,157

146	29	34	36	11,272	5,924	9,542
147	31	34	33	9,724	6,736	8,483
148	30	32	33	10,929	7,299	9,887
149	36	33	36	12,025	5,711	9,254
150	31	40	37	9,9	6,311	8,349
151	32	35	34	12,648	6,787	10,653
152	34	40	40	12,912	5,507	7,788
153	36	42	33	12,246	6,689	8,053
154	31	40	41	11,324	7,079	10,487
155	32	39	35	12,975	6,455	8,809
156	35	40	34	10,776	5,571	10,702
157	34	41	35	12,035	5,948	8,627
158	33	40	37	12,224	6,736	10,234
159	35	42	35	12,678	5,703	9,06
160	29	40	34	10,581	6,978	7,643
161	31	40	40	11,894	6,518	9,061
162	36	36	40	10,118	6,109	8,909
163	36	33	41	12,863	6,888	9,76
164	29	35	38	12,709	50,693	7,64
165	33	36	37	10,492	5,594	8,081
166	33	33	35	12,459	6,609	10,465
167	32	36	39	12,608	6,693	10,699
168	30	41	36	11,097	6,695	9,726
169	36	40	39	10,405	6,953	9,391
170	31	42	41	10	6,482	7,98
171	30	40	39	9,795	6,995	9,963
172	29	38	40	10,23	7,164	7,707
173	32	32	33	10,192	5,73	9,501
174	32	35	32	10,152	6,479	9,473
175	35	36	37	10,301	5,947	10,858
176	31	40	34	11,301	6,831	8,294

177	32	33	39	12,973	5,41	8,221
178	34	37	40	9,644	5,626	7,747
179	32	33	36	10,805	6,261	9,854
180	36	39	39	11,272	6,371	10,942
181	32	36	41	11,62	7,363	7,922
182	34	40	42	11,506	6,95	10,355
183	33	37	37	11,751	6,747	9,495
184	35	42	38	12,022	7,269	7,87
185	33	41	32	12,458	6,795	8,008
186	32	33	34	10,457	5,611	7,665
187	32	35	32	12,829	7,306	8,883
188	33	36	39	11,255	5,773	8,137
189	36	37	36	11,855	63,657	7,606
190	29	38	40	10,755	7,346	9,88
191	35	40	36	10,96	5,857	8,965
192	29	41	33	10,113	5,33	7,933
193	30	34	41	12,793	7,022	9,656
194	31	42	35	10,669	6,906	10,765
195	33	34	41	10,509	7,081	8,597
196	33	38	37	12,935	5,409	9,746
197	32	38	35	10,066	7,281	9,626
198	35	42	39	12,552	5,698	8,088
199	29	40	36	11,582	7,05	8,711
200	32	34	33	10,46	6,487	10,93
201	35	39	38	10,606	5,33	8,851
202	33	34	34	9,81	5,38	8,901
203	33	36	36	12,158	6,315	8,735
204	33	33	42	11,669	5,376	9,634
205	29	32	32	12,57	6,975	8,222
206	33	33	40	12,222	6,589	10,599
207	30	41	39	10,695	6,625	10,131

208	34	32	33	11,824	6,639	8,286
209	32	35	41	12,018	6,892	10,128
210	35	40	38	11,06	7,37	9,374
211	29	33	36	12,351	6,229	10,799
212	30	38	34	10,373	6,791	10,929
213	33	34	38	12,646	6,424	7,701
214	33	39	37	10,504	6,301	8,922
215	34	39	42	12,567	6,419	8,92
216	29	32	35	12,933	5,284	7,824
217	36	36	38	10,968	7,026	10,735
218	30	37	41	12,528	6,151	8,143
219	30	34	41	11,486	7,113	8,089
220	34	38	41	12,093	6,47	10,055
221	35	39	32	10,312	7,118	10,024
222	31	33	34	10,648	5,343	10,558
223	35	35	38	11,937	48,723	9,397
224	35	34	35	12,296	6,671	8,476
225	35	38	36	10,978	63,647	10,079
226	36	32	33	11,88	6,889	8,701
227	31	39	36	12,747	7,141	10,868
228	32	39	36	12,602	6,624	7,82
229	29	35	38	10,062	6,705	9,637
230	36	36	38	9,954	7,398	9,36
231	29	35	42	12,003	7,022	10,54
232	36	40	39	12,171	7,062	8,912
233	29	42	36	9,815	6,635	9,278
234	31	42	35	10,644	5,928	7,914
235	35	32	33	10,829	5,94	8,255
236	32	41	33	10,772	6,753	9,408
237	30	32	42	12,213	6,196	10,455
238	30	41	42	10,495	7,132	9,527

239	36	38	40	11,048	6,574	7,621
240	32	37	32	10,369	5,585	9,004
241	35	32	40	10,743	5,38	10,46
242	32	38	32	11,656	6,373	9,343
243	30	32	34	10,052	6,179	9,817
244	31	36	35	11,969	5,892	10,646
245	32	35	38	10,952	6,656	10,536
246	29	40	40	11,954	7,132	8,449
247	31	34	40	10,375	5,3	9,623
248	32	36	41	9,675	6,233	8,419
249	32	39	35	9,618	6,203	10,208
250	32	32	35	10,992	7,246	7,61
251	33	38	42	11,333	6,7	10,833
252	30	36	40	11,286	6,422	8,598
253	36	40	37	12,818	6,813	9,222
254	34	37	38	12,997	51,047	9,878
255	31	41	33	11,421	7,127	8,615
256	29	42	35	9,739	41,709	10,683
257	35	33	42	12,723	5,932	10,729
258	34	37	35	11,568	6,158	9,13
259	36	35	40	11,578	6,387	8,212
260	34	32	32	11,182	5,223	8,847
261	33	41	35	11,359	6,155	8,627
262	31	39	42	12,801	6,86	8,52
263	36	41	39	10,945	6,683	10,631
264	29	35	37	12,227	6,312	8,567
265	36	37	39	10,899	6,864	9,344
266	36	35	36	12,474	5,446	9,304
267	35	35	37	12,083	5,465	8,923
268	36	41	37	12,352	5,795	9,082
269	31	33	38	12,654	6,367	8,874

270	30	36	41	11,327	5,269	10,394
271	31	35	32	10,131	6,669	8,622
272	35	32	36	9,81	6,741	9,712
273	33	32	32	11,741	5,669	10,793
274	30	39	36	10,471	6,737	7,92
275	30	39	42	11,877	64,307	7,844
276	36	33	42	11,444	5,686	8,648
277	35	34	36	11,035	7,125	10,304
278	35	36	35	10,797	6,385	9,679
279	30	34	41	12,274	5,971	9,338
280	34	34	42	11,999	7,146	9,926
281	33	37	32	11,416	5,899	8,031
282	34	34	34	12,144	6,717	9,241
283	33	32	41	9,999	6,758	9,832
284	34	35	35	10,293	6,66	10,985
285	30	35	41	10,411	52,143	8,004
286	33	37	37	12,677	5,306	9,959
287	35	37	39	10,387	5,431	8,888
288	34	33	34	9,697	7,284	7,811
289	33	35	38	9,625	5,766	9,506
290	32	38	36	11,811	6,866	8,772
291	32	41	39	12,774	6,143	10,598
292	34	36	36	10,151	5,768	9,923
293	35	42	37	9,639	60,017	7,719
294	30	35	40	10,776	6,859	9,487
295	30	40	41	11,283	6,527	8,063
296	33	35	41	10,503	6,972	9,415
297	33	42	38	10,011	5,993	10,602
298	36	33	32	9,72	5,603	9,273
299	33	38	35	10,001	5,706	7,828
300	35	40	39	12,647	6,819	7,648

301	29	39	40	9,97	6,914	10,812
302	31	39	37	11,546	6,77	7,978
303	29	36	42	10,196	7,195	10,474
304	35	37	36	10,667	6,117	10,483
305	35	32	35	10,202	6,247	9,782
306	35	32	40	12,61	5,996	8,924
307	35	36	41	9,838	7,127	8,085
308	33	35	34	10,31	6,594	9,188
309	36	41	35	12,222	6,702	8,642
310	30	42	35	12,995	6,407	7,885
311	30	34	35	12,177	6,763	9,2
312	34	38	39	10,567	5,434	8,219
313	32	34	37	12,893	5,499	8,763
314	35	37	37	12,082	6,674	8,059
315	35	33	41	11,292	5,841	8,471
316	31	40	38	12,212	6,786	9,027
317	31	32	37	9,867	5,498	10,735
318	29	35	37	10,19	6,312	9,349
319	30	33	35	10,508	7,073	9,941
320	35	33	34	11,959	5,731	7,976
321	31	41	37	10,895	6,371	7,815
322	35	40	38	12,822	6,393	10,451
323	30	32	34	11,968	5,537	10,258
324	31	36	42	10,593	5,243	9,893
325	31	35	33	12,215	6,355	9,173
326	35	32	40	10,774	6,617	10,128
327	31	32	40	10,984	6,634	10,137
328	36	38	34	12,39	6,396	9,315
329	29	42	35	12,909	6,594	9,804
330	31	42	38	11,487	6,737	10,262
331	30	35	32	10,177	6,627	8,884

332	30	36	32	10,611	6,469	9,462
333	35	35	40	10,925	7,29	9,773
334	34	40	38	10,184	5,91	10,421
335	36	38	34	12,544	5,65	10,306
336	35	36	38	11,215	6,145	9,941
337	32	38	35	10,4	7,019	9,948
338	35	32	34	10,701	6,793	9,727
339	32	36	36	10,433	6,976	9,436
340	34	37	39	10,465	5,987	10,795
341	36	32	41	11,477	5,316	10,181
342	29	38	37	10,087	5,84	7,965
343	35	33	36	11,765	6,169	9,883
344	30	33	34	10,795	6,288	7,733
345	32	32	39	12,469	6,493	8,812
346	35	34	34	11,65	45,614	9,46
347	31	38	33	10,549	7,181	8,771
348	32	37	34	11,299	5,448	10,648
349	32	42	37	11,955	6,025	8,035
350	29	38	42	11,032	59,757	8,972
351	32	37	33	10,81	5,908	9,59
352	30	36	38	10,328	6,502	10,153
353	30	32	36	12,337	5,377	7,997
354	31	36	33	9,867	7,086	8,542
355	29	37	37	11,19	6,499	8,181
356	30	38	39	12,203	6,486	9,557
357	30	39	33	10,063	5,993	10,047
358	29	42	39	11,681	5,755	10,118
359	33	41	39	10,903	7,325	10,33
360	30	41	36	10,011	58,437	10,314
361	35	32	34	10,376	6,2	8,312
362	32	39	35	10,238	5,21	10,756

363	35	42	32	12,885	7,292	9,01
364	29	38	38	9,817	7,396	7,624
365	34	32	32	10,677	6,967	10,149
366	34	38	38	9,916	5,619	10,339
367	29	42	35	11,785	7,319	8,694
368	30	32	33	10,892	64,612	8,553
369	34	34	41	10,961	6,877	9,421
370	33	32	38	10,544	7,032	8,896
371	31	32	37	9,721	5,976	9,234
372	35	40	37	11,134	7,359	8,251
373	34	37	35	10,908	6,763	8,001
374	34	33	34	11,15	7,203	9,653
375	32	33	35	11,244	6,731	10,272
376	35	34	34	12,263	6,197	8,98
377	30	40	39	11,01	5,479	9,008
378	29	38	38	12,581	7,331	7,796
379	32	32	38	12,796	5,705	10,651
380	33	32	35	11,147	6,361	8,186
381	33	42	42	12,118	5,92	7,97
382	32	41	34	10,963	6,476	7,97
383	32	38	40	12,123	6,193	10,646
384	36	38	37	10,952	6,088	8,179
385	30	39	35	12,839	6,88	7,694
386	31	42	40	9,694	5,462	8,163
387	30	34	40	11,63	5,888	9,36
388	30	37	33	12,723	5,275	8,365
389	36	32	37	11,753	7,121	8,764
390	36	40	40	10,275	7,129	7,873
391	32	37	39	10,147	6,995	10,445
392	32	37	37	12,124	7,248	8,544
393	32	32	35	9,762	5,936	8,253

394	31	36	36	9,818	7,19	8,853
395	34	38	32	10,198	5,364	10,005
396	30	34	41	12,681	55,478	9,675
397	30	33	42	10,262	6,502	7,724
398	31	32	37	11,374	6,001	9,061
399	29	33	41	9,766	6,963	9,108
400	29	40	38	10,74	6,816	9,145
401	35	33	33	10,497	7,358	10,338
402	33	36	41	10,5	7,312	9,755
403	35	40	39	12,36	5,586	9,502
404	35	35	36	12,805	6,038	8,218
405	32	34	35	9,716	58,34	9,83
406	34	38	37	12,886	6,121	9,161
407	32	40	34	9,943	6,031	8,205
408	32	41	41	11,965	6,417	9,591
409	33	35	36	11,493	5,639	10,13
410	29	34	39	9,611	6,227	10,211
411	33	34	33	11,068	6,023	9,934
412	33	34	37	11,56	6,446	10,5
413	36	42	38	12,473	7,087	10,33
414	30	39	41	11,618	6,313	7,859
415	29	42	38	12,908	7,293	10,57
416	31	42	42	12,082	7,239	9,883
417	30	34	35	10,737	5,267	7,867
418	33	32	40	10,677	6,189	7,971
419	33	32	41	11,679	6,41	9,596
420	36	41	36	11,385	6,965	8,456
421	32	34	35	9,818	7,248	8,083
422	35	40	39	12,716	49,758	10,17
423	36	42	34	10,598	6,495	8,466
424	35	38	42	12,654	6,054	8,292

425	29	40	36	10,032	6,837	9,154
426	29	40	35	10,072	50,217	8,815
427	30	41	36	10,489	6,24	8,941
428	31	39	33	9,718	6,3	10,474
429	33	40	34	10,258	6,303	9,636
430	34	40	32	10,824	7,13	8,616
431	30	33	34	9,813	5,305	9,817
432	30	34	33	9,781	5,598	8,651
433	34	42	36	12,825	6,429	9,485
434	30	39	33	10,614	7,332	10,487
435	35	32	41	10,032	5,976	8,553
436	34	41	33	10,727	5,959	10,028
437	34	32	34	9,842	7,281	8,988
438	31	34	34	11,86	5,958	7,995
439	32	33	41	12,621	7,026	10,914
440	33	37	35	10,323	5,688	8,582
441	32	33	33	10,278	5,232	8,422
442	30	33	41	12,653	5,283	9,823
443	34	39	35	11,814	5,497	10,519
444	29	36	38	11,648	6,984	10,51
445	35	37	35	9,836	5,857	8,371
446	29	41	34	11,697	7,247	8,488
447	29	38	36	10,025	7,272	10,564
448	36	42	40	10,695	5,854	8,843
449	31	41	37	9,775	5,608	10,551
450	32	38	37	11,442	6,967	10,599
451	34	37	35	10,214	6,412	7,973
452	30	34	32	12,201	5,2	8,603
453	32	32	40	11,76	6,453	8,687
454	33	36	36	10,054	5,817	9,152
455	35	41	33	12,751	6,525	10,455

456	31	38	34	10,116	5,936	10,836
457	29	39	34	12,75	5,973	10,697
458	34	32	38	9,875	6,447	8,728
459	32	40	35	12,748	5,216	9,946
460	35	33	40	11,878	5,431	10,78
461	33	41	39	9,827	5,412	9,241
462	36	37	36	12,025	6,113	8,222
463	29	34	39	12,738	6,56	9,425
464	34	38	40	10,671	5,807	10,926
465	32	35	39	12,254	5,311	10,515
466	30	41	42	10,388	7,391	10,612
467	32	33	35	11,078	6,381	10,414
468	31	37	37	12,267	6,909	10,618
469	34	32	32	12,796	5,915	8,477
470	30	39	34	12,153	6,743	10,575
471	36	37	42	11,953	6,696	9,145
472	30	41	40	11,301	6,363	8,233
473	29	37	36	11,903	6,303	10,268
474	33	35	34	9,778	6,225	8,022
475	29	36	39	12,905	6,923	10,494
476	34	42	35	10,573	6,007	8,578
477	31	34	39	10,723	6,329	10,82
478	29	36	33	9,926	6,079	9,906
479	29	40	39	11,455	5,458	8,134
480	34	33	34	10,303	5,51	8,701
481	33	35	40	10,75	6,838	7,604
482	29	34	39	12,969	6,534	8,833
483	30	40	34	12,782	7,258	9,585
484	29	41	41	10,124	54,883	10,184
485	29	33	37	10,744	5,764	8,343
486	35	39	34	11,439	5,231	8,38

487	33	37	38	9,935	6,194	9,952
488	35	41	36	10,64	7,108	10,253
489	36	39	38	12,412	5,991	7,727
490	36	38	40	9,872	6,318	10,277
491	31	33	37	10,809	54,704	8,465
492	33	38	40	9,791	6,898	10,247
493	29	42	42	10,213	6,527	8,114
494	34	41	39	12,645	7,293	10,572
495	30	42	34	10,115	6,684	8,145
496	30	38	38	10,306	5,207	8,894
497	29	40	37	11,982	7,382	9,423
498	31	32	36	12,192	5,739	7,752
499	33	34	40	12,015	6,014	9,695
500	35	32	36	12,699	5,397	10,636

ПРИЛОЖЕНИЕ Б ИСХОДНЫЙ КОД ПРОГРАММНОЙ РЕАЛИЗАЦИИ РАЗРАБОТАННЫХ АЛГОРИТМОВ.

```
import sys
import cv2
import cv2.cv as cv
import numpy as np

from parts import Face, Nose, Mouth, Eyes
from threading import Thread
from time import time

path = "cascades/"

# Load Haar cascades
cface  = Face(path)
cnose  = Nose(path, cface)
ceyes  = Eyes(path, cface)
cmouth = Mouth(path, cface)

# Preview scale
scale = 2
```

```
# Mouth vert. center correction
mvcc = 0.3

# Main triangle ratio
k = 0.5

# X correction
mxc = 0.05

# Y correction
myc = 0.035

# Head width history size
memory = 100

# Cursor history size
rmemory = 8

# Width queue
sQueue = []
```

```
# X cursor queue
xQueue = []

# Y cursor queue
yQueue = []

capture = cv2.VideoCapture(0)

last = (0, 0)
lost = False

silent = False
sneak = True

if len(sys.argv) > 1:
        silent = True

def detect (frame):
        global last
        global lost
```

```
size = (frame.shape[1], frame.shape[0])
xsize = int(size[0] / scale)
ysize = int(size[1] / scale)

small = cv2.resize(frame, ((xsize, ysize)))
temp  = cv2.cvtColor(small, cv.CV_BGR2GRAY)
temp  = cv2.flip(temp, 1)

# Detect face
face = cface.find(temp)

if face is None:
        lost = True
        return (last, temp)

lost = False

(x, y, w, h) = face

cface.highlight(temp)
```

```
face = temp[y:y+h, x:x+w]

# Detect eyes in thread
te = Thread(target=Eyes.find, args=(ceyes, face))
te.start()

# Detect mouth in thread
tm = Thread(target=Mouth.find, args=(cmouth, face))
tm.start()

# Detect nose
nose = cnose.find(face)

tm.join()
te.join()

eyes = ceyes.last
mouth = cmouth.last

# Draw rectangle around eyes
if not sneak:
```

```
        ceyes.highlight(temp)

if not mouth is None and not nose is None:
        nx = x + nose[0]  + int(nose[2] / 2)
        ny = y + nose[1]  + int(nose[3] / 2)

        mx = x + mouth[0] + int(mouth[2] / 2)
        my = y + mouth[1] + int(mouth[3] * mvcc)

        if not sneak:
                # Draw nose
                cv2.rectangle(temp, (mx - 1, my - 1), (mx + 1, my + 1), 255)

                # Draw mouth
                cv2.rectangle(temp, (nx - 1, ny - 1), (nx + 1, ny + 1), 255)

                # Mouth -> Nose
                cv2.line(temp, (mx, my), (nx, ny), 2, 0)

        if not eyes is None:
                ex = x + eyes[0]
```

```
ey = y + eyes[1]

cx = int(mx * (1 - k) + ex * k)
cy = int(my * (1 - k) + ey * k)

if not sneak:
        # Mouth -> Eyes
        cv2.line(temp, (mx, my), (ex, ey), 2, 0)

        # Nose -> Eyes
        cv2.line(temp, (nx, ny), (ex, ey), 2, 0)

        # Nose -> Center
        cv2.line(temp, (nx, ny), (cx, cy), 2, 0)

sQueue.append(w)
if len(sQueue) > memory:
        sQueue.pop(0)

cursorScale = sum(sQueue) / len(sQueue)
```

```
# Marker position
rx = xsize / 2 + int((nx - cx) / mxc / cursorScale * xsize / 2)
ry = ysize / 2 + int((ny - cy) / myc / cursorScale * ysize / 2)

# Update cord queues
yQueue.append(ry)
xQueue.append(rx)

if len(yQueue) > rmemory:
        yQueue.pop(0)

if len(xQueue) > rmemory:
        xQueue.pop(0)

# Get average x and y
ry = sum(yQueue) / len(yQueue)
rx = sum(xQueue) / len(xQueue)

if rx < 4:
        rx = 4
```

```
                if ry < 4:
                        ry = 4

                if rx > xsize - 4:
                        rx = xsize - 4

                if ry > ysize - 4:
                        ry = ysize - 4

                #cv2.rectangle(temp, (rx - 4, ry - 4), (rx + 4, ry + 4), 255)
                #cv2.rectangle(temp, (rx - 2, ry - 2), (rx + 2, ry + 2), 255)
                last = (int(rx * scale), int(ry * scale))

        return (last, temp)

while True:
        (success, frame) = capture.read()
        (w, h) = (frame.shape[1], frame.shape[0])

        if not success:
                break
```

```
c = cv.WaitKey(1)

grid = np.zeros((h, w, 1), np.uint8);

((x, y), screen) = detect(frame)

w3 = int(w / 3)
h3 = int(h / 3)

cv2.rectangle(grid, (0, h3), (w - 1, 2 * h3), 255)
cv2.rectangle(grid, (w3, 0), (2 * w3, h - 1), 255)

nx = int(x / w3)
ny = int(y / h3)

if lost:
        print -1, -1
else:
        print x, y
```

```
        sys.stdout.flush()

        if silent:
                continue

        # Highlight selected
        cv2.rectangle(grid, (nx * w3, ny * h3), ((nx + 1) * w3, (ny + 1) * h3), 127, cv.CV_FILLED)

        # Draw cursor
        cv2.rectangle(grid, (x - 4, y - 4), (x + 4, y + 4), 0)
        cv2.rectangle(grid, (x - 2, y - 2), (x + 2, y + 2), 0, cv.CV_FILLED)

        cv2.imshow('grid', grid)
        cv2.imshow('capture', screen)
import cv2

class Part:
        def __init__ (self, path, size):
                self.cascade = cv2.CascadeClassifier(path)
                self.last = None
                self.remains = 0
```

```
        self.dump = False

        # Min pattern size
        self.size = size

        # Store last detected part for % frames
        self.memory = 10

def __getitem__ (self, item):
        return self.last[item]

# Save last seen part
def setLast (self, last):
        self.last = last
        self.remains = self.memory
        return last

# Restore last seen part
def getLast (self):
        if self.remains:
```

```
                self.remains -= 1
                return self.last

        return None

# Find part on frame
def find (self, frame):
        parts = self.cascade.detectMultiScale(frame, 1.2, 2, 0, self.size)

        if self.dump:
                for (x, y, w, h) in parts:
                        cv2.rectangle(frame, (x, y), (x + w, y + h), 0)

        # Try to restore last part
        if not len(parts):
                return self.getLast()

        parts = self.best(parts)

        if not len(parts):
                self.remains = 0
```

```
                self.last = None
                return None

            return self.setLast(parts[-1])

        # Select best available part
        def best (self, parts):
            return sorted(parts, key=lambda part: part[3] * part[2])

        # Draw rectangle around part
        def highlight (self, frame, color=255):
            if self.remains:
                (x, y, w, h) = self.last
                cv2.rectangle(frame, (x, y), (x + w, y + h), color)


class Face (Part):
    def __init__ (self, path):
        Part.__init__(self, path + 'haarcascade_frontalface_default.xml', (100, 100))

    def find (self, frame):
```

```
                result = Part.find(self, frame)

                # Extend face
                if self.remains is self.memory:
                        self.last[3] *= 1.10

                return result


class Nose (Part):
        def __init__ (self, path, face):
                Part.__init__(self, path + 'haarcascade_mcs_nose.xml', (18, 15))
                self.face = face

        def best (self, parts):
                face = self.face.last

                r = face[3] * 0.8
                l = face[3] * 0.2

                return Part.best(self, [
```

```
                part for part in parts if l < part[1] and (part[1] + part[3]) < r
            ])


class Mouth (Part):
    def __init__ (self, path, face):
        Part.__init__(self, path + 'haarcascade_mcs_mouth.xml', (25, 15))
        self.face = face
        self.up = 0.675

    def best (self, parts):
        face = self.face.last

        l = face[3] * self.up

        return Part.best(self, [
            part for part in parts if l < part[1] + part[3] / 2
        ])

class Eyes (Part):
    def __init__ (self, path, face):
```

```
        Part.__init__(self, path + 'haarcascade_eye.xml', (10, 10))
        self.left  = None
        self.right = None
        self.memory *= 2
        self.face = face

def pair (self, parts):
        for (ax, ay, aw, ah) in parts:
                for (bx, by, bw, bh) in parts:
                        if ax is bx and ay is by:
                                continue # same

                        if abs(ay - by) < ((ah + bh) / 4):
                                if ax + aw < bx or bx + bw < ax:
                                        return ((ax, ay, aw, ah), (bx, by, bw, bh))

        return None

def find (self, frame):
        eyes = None
        parts = self.cascade.detectMultiScale(frame, 1.2, 2, 0, self.size)
```

```
        h = self.face.last[3] / 2

        parts = self.pair(Part.best(self, [
                part for part in parts if part[1] + part[3] / 2 < h
        ]))

        if parts is None:
                return self.getLast()

        return self.setLast(parts)

def setLast (self, parts):
        parts = sorted(parts, key=lambda part: part[0])

        l1 = parts[0]
        r1 = parts[1]

        if not self.left is None and not self.right is None:
                l0 = self.left
                r0 = self.right
```

```
                self.left  = [(l0[0] + l1[0]) / 2, (l0[1] + l1[1]) / 2, (l0[2] + l1[2]) / 2, (l0[3] + l1[3]) / 2]
                self.right = [(r0[0] + r1[0]) / 2, (r0[1] + r1[1]) / 2, (r0[2] + r1[2]) / 2, (r0[3] + r1[3]) / 2]
        else:
                self.left  = parts[0]
                self.right = parts[1]

        self.last = [
                ((self.left[0] + self.right[0]) + (self.left[2] + self.right[2]) / 2) / 2,
                ((self.left[1] + self.right[1]) + (self.left[3] + self.right[3]) / 2) / 2
        ]

        self.remains = self.memory
        return self.last

def highlight (self, frame, color=255):
        if self.remains:
                x = self.face.last[0]
                y = self.face.last[1]

                for (ex, ey, ew, eh) in [self.left, self.right]:
```

```
                    nx = x + ex + ew / 2
                    ny = y + ey + eh / 2
                    cv2.rectangle(frame, (nx - 1, ny - 1), (nx + 1, ny + 1), color)

                nx = x + self.last[0]
                ny = y + self.last[1]
                cv2.rectangle(frame, (nx - 1, ny - 1), (nx + 1, ny + 1), color)
```

```
unit main;

interface

uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
  Dialogs, GLScene, GLVectorFileObjects, GLCoordinates, GLObjects,
  GLCadencer, GLMaterial, GLWin32Viewer, GLCrossPlatform, BaseClasses,
  GLLensFlare, GLNavigator, GLSimpleNavigation, GLFireFX, GLAtmosphere,
  GLSkydome, GLTerrainRenderer, GLHeightData, ExtCtrls, GLHUDObjects,
  BCPort, GLMesh;

type
  TInput = class(TThread)
```

```
private
  inpPipe: string;
  procedure SetProgress;
protected
  procedure Execute; override;
end;

TForm1 = class(TForm)
  GLScene1: TGLScene;
  GLSceneViewer1: TGLSceneViewer;
  Target: TGLDummyCube;
  GLActor1: TGLActor;
  GLCamera1: TGLCamera;
  GLLightSource1: TGLLightSource;
  gml: TGLMaterialLibrary;
  Cadencer: TGLCadencer;
  GLFireFXManager1: TGLFireFXManager;
  LeftFlare: TGLDummyCube;
  RightFlare: TGLDummyCube;
  GLSkyDome1: TGLSkyDome;
  GLTerrainRenderer1: TGLTerrainRenderer;
```

```
hdsBmp: TGLBitmapHDS;
moveControl: TGLDummyCube;
RollControl: TGLDummyCube;
GUIML: TGLMaterialLibrary;
s_up: TGLHUDSprite;
s_down: TGLHUDSprite;
s_left: TGLHUDSprite;
s_right: TGLHUDSprite;
com1: TBComPort;
barrelRollControl: TGLDummyCube;
tree1: TGLActor;
ssTarget: TGLHUDSprite;
procedure FormCreate(Sender: TObject);
procedure CadencerProgress(Sender: TObject; const deltaTime,
  newTime: Double);
procedure FormKeyDown(Sender: TObject; var Key: Word;
  Shift: TShiftState);
procedure FormKeyUp(Sender: TObject; var Key: Word;
  Shift: TShiftState);
procedure FormMouseMove(Sender: TObject; Shift: TShiftState; X,
  Y: Integer);
```

```
  procedure FormClose(Sender: TObject; var Action: TCloseAction);
private
  StdOutPipeRead, StdOutPipeWrite: THandle;
  PI                        : TProcessInformation;
  procedure createPipe_;
  procedure WMHotKey(var Message:TMessage); message WM_HOTKEY;
public
  ForvardSpeed: real;
  SideSpeed: real;
  RollAngle: integer;
  PitchAngle: integer;
  RollLock: boolean;
  PitchLock: boolean;
  TimeStamp: double;
  Height : double;
  VertSpeed: Double;
  ILine: string;
  inpCom: string;
  outCom: string;
  hInp, vInp: integer;
  voiceCommand: integer;
```

```
    engineCommand: integer;
    barrelLock: integer;
    barrelTime: double;
    procedure ParseInput;
    procedure processInput;
    procedure doBarrelRoll;
  end;

var
  Form1: TForm1;
  input: TInput;

implementation

uses glfile3ds, jpeg, math;

{$R *.dfm}

procedure TInput.Execute;
begin
  while true do
```

```
  begin
    readln(inpPipe);
    Synchronize(SetProgress);
  end;
end;

procedure Tinput.SetProgress;
begin
  try
    form1.com1.ReadStr(form1.inpCom,1);
    form1.com1.WriteStr(form1.outCom);
  except
    form1.inpCom := '0';
  end;
  form1.ILine := inpPipe;
  //form1.Caption := inpPipe;
end;

procedure Tform1.createPipe_;
var
    SA                      : TSecurityAttributes;
```

```
  SI                    : TStartupInfo;
begin
  With SA do
  Begin
    nLength := SizeOf( SA );
    bInheritHandle := True;
    lpSecurityDescriptor := nil;
  end;

  CreatePipe( StdOutPipeRead,
              StdOutPipeWrite,
              @SA,
              0);
  with SI do
  Begin
    FillChar( SI, SizeOf( SI ), 0 );
    cb := SizeOf( SI );
    dwFlags := STARTF_USESHOWWINDOW ;
    wShowWindow := SW_HIDE;
    hStdInput := GetStdHandle( STD_INPUT_HANDLE );
    hStdOutput := StdOutPipeRead;
```

```
    hStdError := StdOutPipeWrite;
  end;

  CreateProcess( nil,
                 PChar( 'C:\Python27\python.exe .\detect.py' ),
                 nil,
                 nil,
                 true,
                 0,
                 nil,
                 PChar( extractFilePath(application.ExeName) ),
                 SI,
                 PI);
end;

procedure TForm1.FormCreate(Sender: TObject);
var path: string;
begin
  iline := '1 1';
  Input := Tinput.Create(false);
```

```
hdsBmp.MaxPoolSize:=32*1024*1024;
Path := 'media\textures\';
gml.Materials[0].Material.Texture.Image.LoadFromFile(Path + 'landscape.bmp');
gml.Materials[1].Material.Texture.Image.LoadFromFile(Path + 'detailmap.jpg');
hdsBmp.Picture.LoadFromFile(path + 'terrain.bmp');
GLTerrainRenderer1.TilesPerTexture:= 64/GLTerrainRenderer1.TileSize;
GLSceneViewer1.Align := alclient;
glactor1.LoadFromFile('SHUKOI_L.3DS');
//tree1.LoadFromFile('tree1.3ds');
ForvardSpeed := 200;
SideSpeed := 0;
RollAngle := 0;
PitchAngle := 0;
RollLock := false;
PitchLock := false;
Height := 0;
VertSpeed := 0.5;
try
  com1.Open;
except
end;
```

```
  engineCommand := 0;
  barrelLock := 0;

  {RegisterHotKey(Handle, 1, MOD_CONTROL or MOD_SHIFT, vk_f3);
  RegisterHotKey(Handle, 2, MOD_CONTROL or MOD_SHIFT, vk_f4);
  RegisterHotKey(Handle, 3, MOD_CONTROL or MOD_SHIFT, vk_f5);
  RegisterHotKey(Handle, 4, MOD_CONTROL or MOD_SHIFT, vk_f6);
  RegisterHotKey(Handle, 5, MOD_CONTROL or MOD_SHIFT, vk_f7);}
end;

procedure TForm1.CadencerProgress(Sender: TObject; const deltaTime,
  newTime: Double);
begin
  if (newTime - TimeStamp >= 0.02) then
  begin
    form1.processInput;
    TimeStamp := newTime;
    if not RollLock then
    begin
      if RollAngle > 0 then
        RollAngle := RollAngle - 1;
```

```
  if RollAngle < 0 then
    RollAngle := RollAngle + 1;
end;
if not PitchLock then
begin
  if PitchAngle > 0 then
    PitchAngle := PitchAngle - 1;
  if PitchAngle < 0 then
    PitchAngle := PitchAngle + 1;
end;
if PitchAngle > 0 then
  if height < 70 then
    height := height + VertSpeed;
If PitchAngle < 0 then
  if height > -10 then
    height := height - vertSpeed;
if barrelLock = 1 then
begin
  if newtime - barrelTime > 1 then
  begin
    barrelLock := 0;
```

```
      barrelrollcontrol.RollAngle := 0;
    end
    else
      barrelrollcontrol.RollAngle := 360 * (newtime - barrelTime);
  end;
end;
RollControl.Up.Y := cos(DegToRad(RollAngle));
RollControl.Up.X := -sin(DegToRad(RollAngle));
RollControl.Direction.z := Cos(DegToRad(PitchAngle));
RollControl.Direction.y := sin(DegToRad(PitchAngle));
Target.Up.Y := cos(DegToRad(RollAngle / 2));
Target.Up.X := -sin(DegToRad(RollAngle / 2));
Target.Direction.z := Cos(DegToRad(PitchAngle / 2));
Target.Direction.y := sin(DegToRad(PitchAngle / 2));
SideSpeed := -2 * RollAngle;
if PitchAngle > 0 then
begin
  if RollAngle < 0 then
    MoveControl.Turn(-PitchAngle / 100);
  if RollAngle > 0 then
    MoveControl.Turn(PitchAngle / 100);
```

```
      end;
      MoveControl.Position.Z := MoveControl.Position.Z + ForvardSpeed * deltaTime * MoveControl.Direction.Z - SideSpeed
* deltaTime * MoveControl.Direction.X;
      MoveControl.Position.X := MoveControl.Position.X + SideSpeed * DeltaTime * MoveControl.Direction.Z +
ForvardSpeed * DeltaTime * MoveControl.Direction.X;
      MoveControl.Position.Y := Height;

      //tree1.Position.y := hdsbmp.InterpolatedHeight((MoveControl.Position.x + 50)/4,(MoveControl.Position.z + 50)/4,32) *
GLTerrainRenderer1.Scale.Z * 0.007 - 45;

      glsceneviewer1.Invalidate;
    end;

    procedure TForm1.FormKeyDown(Sender: TObject; var Key: Word;
      Shift: TShiftState);
    begin
      if key = vk_left then
      begin
        if RollAngle > - 45 then
          RollAngle := RollAngle - 5;
        RollLock := true;
```

```
end;
if key = vk_right then
begin
  if RollAngle < + 45 then
    RollAngle := RollAngle + 5;
  RollLock := true;
end;
if key = vk_up then
begin
  if PitchAngle > - 20 then
    PitchAngle := PitchAngle - 1;
  PitchLock := true;
end;
if key = vk_down then
begin
  if PitchAngle < + 35 then
    PitchAngle := PitchAngle + 1;
  PitchLock := true;
end;
if key = vk_f1 then engineCommand := 1;
if key = vk_f2 then engineCommand := 2;
```

```
  //if key = vk_f3 then form1.doBarrelRoll;
end;

procedure TForm1.FormKeyUp(Sender: TObject; var Key: Word;
  Shift: TShiftState);
begin
  if key in [vk_left, vk_right] then
  begin
    RollLock := false;
    TimeStamp := cadencer.CurrentTime;
  end;
  if key in [vk_up, vk_Down] then
  begin
    PitchLock := false;
    TimeStamp := cadencer.CurrentTime;
  end;
end;

procedure TForm1.FormMouseMove(Sender: TObject; Shift: TShiftState; X,
  Y: Integer);
begin
```

```
if X <  450 then
begin
  if RollAngle > - 45 then
    RollAngle := RollAngle - 5;
  RollLock := true;
end;
if X > 900 then
begin
  if RollAngle < + 45 then
    RollAngle := RollAngle + 5;
  RollLock := true;
end;
if (X < 900) and (x > 450) then
begin
  RollLock := false;
  TimeStamp := cadencer.CurrentTime;
end;
if Y < 250 then
begin
  if PitchAngle > - 20 then
    PitchAngle := PitchAngle - 1;
```

```
    PitchLock := true;
  end;
  if Y > 450 then
  begin
    if PitchAngle < + 35 then
      PitchAngle := PitchAngle + 1;
    PitchLock := true;
  end;
  if (Y >= 250) and (Y <= 450) then
  begin
    PitchLock := false;
    TimeStamp := cadencer.CurrentTime;
  end;
end;

procedure Tform1.ParseInput;
var x,y: integer;
begin
  try
    X := abs(StrToInt(copy(iline,1, Pos(' ',iline) - 1)));
    Y := abs(StrToInt(copy(iline, Pos(' ',iline) + 1, Length(iline) - Pos(' ',iline))));
```

```
  if iline = '-1 -1' then
  begin
    hinp := 1;
    vinp := 1;
  end
  else begin
    if X <  213 then
      hinp := 0;
    if x > 426 then
      hinp := 2;
    if (x>=213) and (x <= 426) then
      hinp := 1;
    if y <  160 then
      vinp := 0;
    if y > 320 then
      vinp := 2;
    if (y>=160) and (y <= 320) then
      vinp := 1;
    X := x * 2 + 43;
//    y := y;
    ssTarget.Position.X := x - 77;
```

```
      ssTarget.Position.Y := y - 77;
    end;
  except
    vinp :=1;
    hinp :=1;
  end;
  if inpCom = '' then
  begin
    voiceCommand := -1;
    exit;
  end;
  try
    voiceCommand := strtoint(inpCom);
    if not voiceCommand in [1,2,3,4,5] then voiceCommand := -1;
  except
    voiceCommand := -1;
  end;
  if voiceCommand = 5 then doBarrelRoll;
  if voiceCommand = 1 then ssTarget.Visible := not ssTarget.Visible;
end;
```

```
procedure Tform1.processInput;
begin
  ParseInput;
  form1.Caption := inttostr(hinp) + ' ' + inttostr(vinp);
  if voiceCommand = 3 then hinp := 0;
  if voiceCommand = 4 then hinp := 2;
  if length(outcom)> 0 then
   if outcom[1] in ['G', 'S'] then exit;
  if engineCommand > 0 then
  begin
    if engineCommand = 1 then outcom := 'G';
    if engineCommand = 2 then outcom := 'S';
    engineCommand := 0;
  end
  else begin
    if hinp = 0 then outcom := 'L';
    if hinp = 1 then outcom := 'F';
    if hinp = 2 then outcom := 'R';
  end;
  //form1.Caption := outcom;
  {try
```

```
  form1.com1.WriteStr(form1.outCom);
except end;}
outcom := '';
if hinp = 0 then
begin
  if RollAngle > - 45 then
    RollAngle := RollAngle - 5;
  RollLock := true;
end;
if hinp = 2 then
begin
  if RollAngle < + 45 then
    RollAngle := RollAngle + 5;
  RollLock := true;
end;
if hinp = 1 then
begin
  RollLock := false;
  TimeStamp := cadencer.CurrentTime;
end;
if vinp = 2 then
```

```
  begin
    if PitchAngle > - 20 then
      PitchAngle := PitchAngle - 1;
    PitchLock := true;
  end;
  if vinp = 0 then
  begin
    if PitchAngle < + 35 then
      PitchAngle := PitchAngle + 1;
    PitchLock := true;
  end;
  if vinp = 1 then
  begin
    PitchLock := false;
    TimeStamp := cadencer.CurrentTime;
  end;

end;

procedure TForm1.FormClose(Sender: TObject; var Action: TCloseAction);
begin
```

```
  com1.Close;
end;

procedure tform1.doBarrelRoll;
begin;
  barrelLock := 1;
  barrelTIme := cadencer.CurrentTime;
end;

procedure TForm1.WMHotKey(var Message: TMessage);
begin
  {if message.WParam in [1..5] then
    voiceCommand := message.WParam
  else inherited;
  form1.Caption := inttostr(voicecommand);
  if voiceCommand = 5 then doBarrelRoll;  }
end;

end.
```