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

ФАКУЛЬТЕТ **Специального машиностроения**

КАФЕДРА **СМ11 «Подводные роботы и аппараты»**

**ОТЧЕТ ПО ПРОИЗВОДСТВЕННОЙ ПРАКТИКЕ**

Студент Андреев Евгений Викторович

*фамилия, имя, отчество*

Группа СМ11-11М

Тип практики **Научно-исследовательская работа**

Название предприятия **НУК СМ МГТУ им. Н.Э. Баумана**

Студент **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_** Андреев Е. В.

*подпись, дата фамилия, и.о.*

Руководитель практики **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_** Макашов А. А. *подпись, дата фамилия, и.о.*

Оценка \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

*2020 г.*

Министерство науки и высшего образования Российской Федерации

Федеральное государственное бюджетное образовательное

учреждение высшего образования

«Московский государственный технический университет имени Н.Э. Баумана

(национальный исследовательский университет)»

(МГТУ им. Н.Э. Баумана)

Кафедра **«Подводные роботы и аппараты»** **(СМ11)**

**ЗАДАНИЕ**

**на выполнение научно-исследовательской работы**

**(производственной практики)**

на предприятии **НУК СМ МГТУ им. Н.Э. Баумана**

Студент Андреев Евгений Викторович, СМ11-11М

(фамилия, имя, отчество; индекс группы)

**Тема научно-исследовательской работы:**

Использование каскадного детектора для построения системы позиционирования подводного аппарата

**Дата выдачи задания «» февраля 2020 г.**

**Руководитель НИР**   **/** Макашов А. А.

(подпись, дата)

**Студент \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ /** Андреев Е. В.

(подпись, дата) (Фамилия И.О.)

РЕФЕРАТ

Отчёт на стр., ч., рис., источников, таблиц.

ИСПОЛЬЗОВАНИЕ КАСКАДНОГО ДЕТЕКТОРА ДЛЯ ПОСТРОЕНИЯ СИСТЕМЫ ПОЗИЦИОНИРОВАНИЯ ПОДВОДНОГО АППАРАТА

Перечень ключевых слов: АНПА, опорный маркер специального вида, алгоритм детектирования, каскадный детектор Хаара, донная зарядная станция, видеокамера.

Целью данной работы является исследование возможности использования каскадного классификатора Хаара для позиционирования АНПА у донной зарядной станции с помощью опорных маркеров специального вида.

В процессе работы был проведён сбор и систематизация информации по подготовке данных и обучению каскадного классификатора Хаара методом Виолы-Джонса. Был изготовлен макет донной зарядной станции и спроектированы опорные маркеры специального вида двух типов. Обучены несколько каскадных детекторов различных конфигураций для обнаружения каждого типа маркеров. Исследованы величины задержек и потоков IP-камеры.

В результате исследования были проанализированы скорость и точность работы алгоритмов позиционирования на базе каскадов Хаара и LBP, а также влияние разрешения видеопотока и количества позитивных и негативных примеров, использованных при обучении. Разработан пример продвинутой фильтрации информации, полученной на выходе каскадного детектора.

СОДЕРЖАНИЕ

[ПЕРЕЧЕНЬ СОКРАЩЕНИЙ И ОБОЗНАЧЕНИЙ 6](#_Toc41490451)

[ВВЕДЕНИЕ 7](#_Toc41490452)

[1 Каскадный классификатор Хаара 8](#_Toc41490453)

[1.1 Этап подготовки данных 8](#_Toc41490454)

[1.2 Этап обучения каскада 11](#_Toc41490455)

[1.3 Сравнение каскдов Хаара и LBP 16](#_Toc41490456)

[2 Применение каскадного детектора для определения координат 19](#_Toc41490457)

[2.1 Разработка маркеров специального вида 19](#_Toc41490458)

[2.1 Калибровка камеры по расстоянию 20](#_Toc41490459)

[2.2 Определение относительных смещений 22](#_Toc41490460)

[2.3 Определение поворота относительно нормали 22](#_Toc41490461)

[СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ 23](#_Toc41490462)

[Приложение А. Сценарий для администрирования большого количества изображений в процессе обучения каскадного классификатора 24](#_Toc41490463)

[Приложение Б. Сценарий для наполнения файла фоновых изображений 25](#_Toc41490464)

# ПЕРЕЧЕНЬ СОКРАЩЕНИЙ И ОБОЗНАЧЕНИЙ

В настоящем отчете о НИР применяются следующие сокращения и обозначения:

АНПА – Автономный необитаемый подводный аппарат;

ВК – Видеокамера;

СИД – Светоизлучающий диод;

ROS – Robot operating system.

# ВВЕДЕНИЕ

# 1 Каскадный классификатор Хаара

Каскадный классификатор Хаара представляет собой специальный детектор границ, построенный на каскадах решающих деревьев, каждое из которых содержит в своих листьях один из примитивов Хаара (см. рисунок Рисунок 1) и два пороговых значения соответствующие ему.

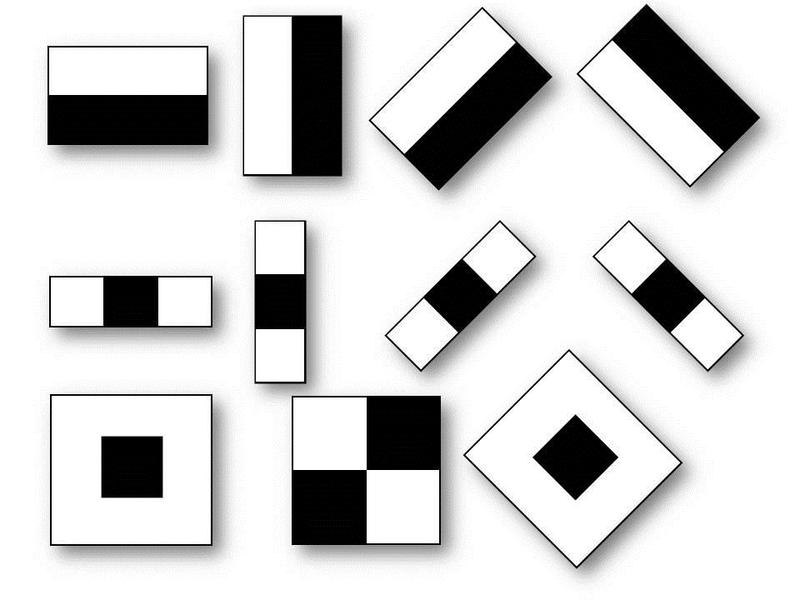


Рисунок – Некоторые примитивы Хаара

Обучение каскада построено на последовательном переборе примитивов и расчёте значений признаков как разницы между суммарной яркостью пикселей, покрытых былой областью текущего примитива и суммарной яркости пикселей чёрной области.

## 1.1 Этап подготовки данных

Далее приведена последовательность действий для подготовки изображений и их разметки

1. Для тренировки каскада потребуется два набора данных: с объектом и без него. Допускается наличие в кадре одновременно нескольких объектов. Удобнее всего не снимать последовательно несколько сотен кадров, заснять два видео. Подобный подход позволит существенно сэкономить время, а также одновременно учесть все негативные эффекты, такие как размытия, расфокусировка, возникающие во время движения реального робота.
2. Полученные видеофайлы разбиваются на кадры. Например, это можно сделать с помощью бесплатной утилиты “Free Video to Jpeg Converter”. Наибольшее количество *уникальных* кадров определяется по формуле:

F \* T,

где

F – частота, кадров/с;

T - длительность видеоролика, с.

1. Для удобства предлагается переименовать изображения, оставив короткий псевдоним из латинских символов и порядковый номер (далее будет обоснована важность этого шага). Например, IMG\_20200509\_144931.jpg превращается в with0001.jpg. Используется бесплатное ПО FastStone Image Viewer, имеющее инструменты пакетной обработки фотографий.
2. Для тренировки алгоритма, как правило, используются изображения небольшого разрешения (1280 х 720 пикселей, 640 х 480 пикселей или меньше). В противном случае обучение займёт чрезвычайно много времени. В этом же ПО после переименования можно отмасштабировать изображения с уменьшением разрешения до указанных выше значений.
3. С помощью утилиты opencv\_createsamples.exe (входит в пакет поставки фреймворка OpenCV) на позитивных изображениях необходимо вручную указать объект(-ы). Запук утилиты производится командой:

opencv\_annotation.exe --annotations=good\_2.dat

--images=./extracted\_images/with\_original

Ключ «annotations» с параметром «good\_2.dat» задаёт выходной файл, в который будет сохранена информация о разметке, ключ «images» определяет папку с изображениями, содержащими требуемый объект. На рисунке Рисунок 2 показан процесс аннотации изображения, содержащего макет донной зарядной станции с нанесёнными на неё маркерами специалльного вида. Предполагается осуществлять навигацию автономного необитаемого подводного аппарата по системе технического зрения в процессе его стыковки с зарядной станцией.

О том, как взаимодействовать с программой разметки [2], после запуска напоминает подсказка в командной строке:

\* mark rectangles with the left mouse button,

\* press 'c' to accept a selection,

\* press 'd' to delete the latest selection,

\* press 'n' to proceed with next image,

\* press 'esc' to stop.

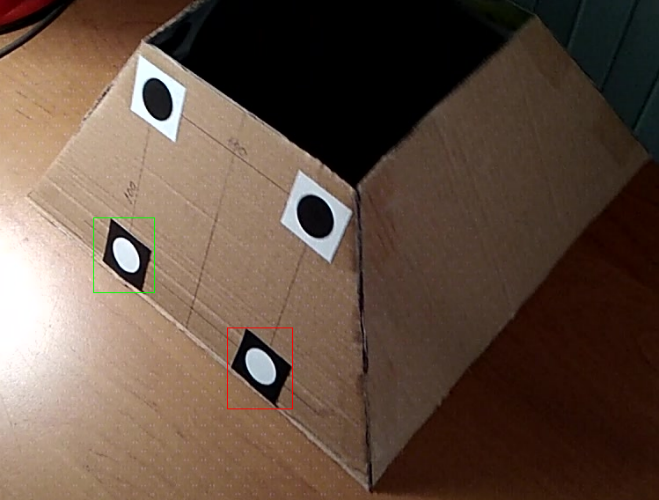


Рисунок – Процесс разметки изображений. Подлежащие детектированию объекты выделены рамками. Слева – подтверждённая аннотация (после нажатия клавиши «с»), справа – не подтверждённая

Программа не будет откликаться на действия пользователя, если выбрана отличная от английской раскладка клавиатуры. По нажатии клавиши «Escape» приложение завершит работу и по указанному пути появится файл с содержимым, показанным на рисунке Рисунок 3.

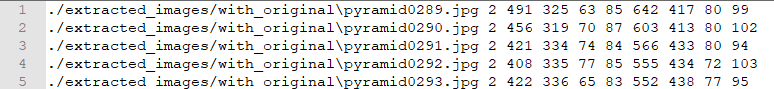


Рисунок – Вид содержимого файла аннотаций изображений с объектом

Каждая строчка соответствует отдельной аннотации, сначала указан путь к файлу (относительный или абсолютный, с зависимости от указанных при запуске программы разметки аргументов); следующий параметр – количество объектов на изображении, в данном примере на каждом кадре по два маркера одного вида; оставшиеся цифры обозначают координаты прямоугольников: левый верхний угол, затем правый нижний. Порядок описания прямоугольников соответствует порядку разметки пользователем.

Вообще говоря, файл негативных примеров может содержать любые изображения, включая и те, которые роботу не доведётся встретить в процессе выполнения задачи. Однако детектор будет функционировать намного лучше, если в качестве негативных примеров использовать изображения обстановки, в которой предстоит вести обнаружение. На рисунке Рисунок 4 показан фрагмент такого файла. Он не содержит никакой лишней информации, кроме относительного пути к изображениям.

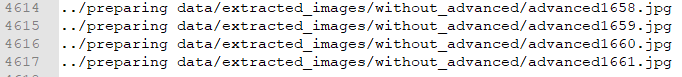


Рисунок – Вид содержимого файла с описанием фоновых изображений

## 1.2 Этап обучения каскада

Для обучения каскада [2] требуется передать последнему файл-вектор, представляющий собой бинарный набор сжатых изображений объекта. Рассмотрим подробнее команду создания вектора:

opencv\_createsamples.exe -info good\_1\_fixed.dat -vec samples.vec -num 1274 -w 24 -h 24 -show

Данное действие выполняет программа opencv\_createsamples.exe, также входящая в пакет поставки фреймворка компьютерного зрения. С ключём «info» указывается путь к файлу с аннотациями, полученному ранее. Ключ «vec» обозначает имя выходного вектор-файла, «num» - это количество позитивных изображений, а параметры «w» и «h» специфицируют ширину и выстоу впикселях. Чем больше размеры объекта, тем дольше будет проходить обучение, но тем точнее будет детектирование. Важно, чтобы отношение указанных величин соответствовало пропорциям реального объекта: к примеру, для поиска на изображении конфеты или болта стоит указать -w 40 -h 10, а для обнаружения маркера как на рисунке 2 лучше оставить квадратные пропорции.

Запуск обучения каскада выполняется командой

opencv\_traincascade.exe -data haar\_output -vec samples.vec -bg bad.dat -numStages 16 -numThreads 12 -w 24 -h 24 -numPos 1100 -numNeg 2955 -mode ALL,

где

data – путь к каталогу куда будут сохранены результаты;

vec – входной вектор-файл;

bg – файл фоновых изображений (background);

numStages – количество итераций (каскадов);

numThreads – количество задействованных потоков процессора;

w, h – ширина и высота изображений вектора в пикселях;

numPos – количество позитивных примеров, необходимых для обучения каскада. Это число обязательно должно быть меньше числа изображений в vec-файле. Причина будет объяснена ниже;

numNeg – количество фоновых изображений, чем больше, тем лучше;

mode – режим на стандартном наборе примитивов Хаара или же на расширенном (включая повёрнутые на 90º).

Программа отобразит следующий вывод:

PARAMETERS:

cascadeDirName: haar\_output

vecFileName: samples.vec

bgFileName: bad\_navigation.dat

numPos: 1100

numNeg: 2955

numStages: 16

precalcValBufSize[Mb] : 1024

precalcIdxBufSize[Mb] : 1024

acceptanceRatioBreakValue : -1

stageType: BOOST

featureType: HAAR

sampleWidth: 24

sampleHeight: 24

boostType: GAB

minHitRate: 0.995

maxFalseAlarmRate: 0.5

weightTrimRate: 0.95

maxDepth: 1

maxWeakCount: 100

mode: ALL

Number of unique features given windowSize [24,24] : 261600

Здесь важными параметрами являются minHitRate и maxFalseAlarmRate. Первый обозначает точность определения, которую должен достичь каскад на каждой стадии. Значение 0,995 означает, что из 1000 изображений 5 будут ложноположительными срабатываниями. Второй – какое количество изображений, не содержащих объект, будет отсеяно на каждой стадии. К примеру, для обученного каскада из 8 стадий будет отсеиваться 1 – 0,58 = 0,996 = 99,6% всех негативных изображений.

По завершении обучения каждой стадии программа будет выдавать подобные сообщения:

===== TRAINING 12-stage =====

<BEGIN

POS count : consumed 1100 : 1195

NEG count : acceptanceRatio 2955 : 2.89746e-05

Precalculation time: 4.876

+----+---------+---------+

| N | HR | FA |

+----+---------+---------+

| 1| 1| 1|

+----+---------+---------+

| 2| 1| 1|

+----+---------+---------+

| 3| 1| 1|

+----+---------+---------+

| 4| 1| 1|

+----+---------+---------+

| 5| 0.999333| 0.836887|

+----+---------+---------+

| 6| 0.999333| 0.843316|

+----+---------+---------+

| 7| 0.998667| 0.649746|

+----+---------+---------+

| 8| 0.996667| 0.581726|

+----+---------+---------+

| 9| 0.996667| 0.461591|

+----+---------+---------+

END>

Training until now has taken 0 days 1 hours 23 minutes 3 seconds.

Третий столбец – значение FalseAlarmRate, переход к следующей стадии происходит, когда оно становится менее 0,5. Фраза «Required leaf false alarm rate achieved. Branch training terminated.» сообщит об успешном завершении обучения классификатора [4]. Работа обученного детектора по распознаванию маркеров специального вида паказана на рисунке Рисунок 5.



Рисунок – Работа каскадного классификатора Хаара. Слева видно одно ложноположительное срабатывание

На каждом этапе каскад совершает ошибки, принимая некоторые фоновые изображения за изображения с объектом. Для тренировки каждой последующей стадии берутётся количество изображений, соответвтующих параметру numPos плюс все ошибочные с предыдущей стадии. В примере вывода выше для 12-й стадии обучения присутствует строчка POS count : consumed 1100 : 1195,сообщающая о том, что для тренировки нынешней стадии было взято 1195 изображений, 1100 из которых изначально переданы в vec-файле. Следовательно, на прерырущем этапе ошибочно были распознаны 95 изображений.

Приблизительно количество позитивных изображений, необходимых для создания vec-файла, можнно оценить по формуле:

V ≥ numPos + (numStages - 1) \* (1 - minHitRate) \* numPos + S,

где

V – число изображений, использованных для создания вектора;

S – суммарное количество нераспознанных изображений на всех стадиях.

По приведённой выше формуле также можно оценить необходимое число объектов, передаваемых opencv\_traincascade.exe в параметре -numPos 1100. При меньшем числе пользователь получит ошибку «Bad argument (Can not get new positive sample. The most possible reason is insufficient count of samples in given vec-file).»

## 1.3 Сравнение каскдов Хаара и LBP

Утилита «opencv\_traincascade.exe» позволяет обучать как каскады Хаара, так и LBP (Local Binary Patterns). В открытых источниках [5] упоминается, что 2-й тип каскадного детектора в несколько раз быстрее каскада Хаара и на 15-20% менее точен. Было проведено собственное сравнение, результаты которого представлены в таблице 1.

Таблица 1 - Сравнение производительности детектирования на основе каскадов Хаара и LBP

|  |  |  |
| --- | --- | --- |
| Тип | Источник видеоряда | Скорость, кардов/с |
| LBP | Тестовое видео, 1280х720 пикселей | 12,5 |
| HAAR | 11 |
| LBP | Потоковое видео с домашней веб-камеры, 640х480, разрешение по умолчанию для OpenCV | 17 |
| HAAR | 15 |
| LBP | Тестовое видео, 1280х720 пикселей, детектируется только один тип маркера (1-й тип - чёрный круг на белом фоне) | 17 |
| HAAR | 15 |
| LBP | Потоковое видео с домашней веб-камеры, 640х480, детектируется только один тип маркера (1-й тип - чёрный круг на белом фоне) | 20 |
| HAAR | 19 |

По данным таблицы 1 ожидаемой разницы в производительности не наблюдается. Однако в выводе программы «opencv\_traincascade.exe» при равных размерах поискового окна количество уникальных комбинаций примитивов, их размеров и положений сильно различается:

|  |  |
| --- | --- |
| stageType: BOOST  featureType: HAAR  sampleWidth: 24  sampleHeight: 24  boostType: GAB  minHitRate: 0.995  maxFalseAlarmRate: 0.5  weightTrimRate: 0.95  maxDepth: 1  maxWeakCount: 100  mode: ALL  Number of unique features given windowSize [24,24] : **261600** | stageType: BOOST  featureType: LBP  sampleWidth: 24  sampleHeight: 24  boostType: GAB  minHitRate: 0.995  maxFalseAlarmRate: 0.5  weightTrimRate: 0.95  maxDepth: 1  maxWeakCount: 100  Number of unique features given windowSize [24,24] : **8464** |

Сравнете значения, выделенные жирным. Автор связывает подобное поведение с особенностью разработанных маркеров: каскадам необходимо детектировать плоское контрастное изображение из простых геометрических фигур, в результате чего большое количество примитивов отсеивается на ранних стадиях. Таким образом, для детектирования достаточно весьма ограниченного набора примитивов и соответствующих им слабых классификаторов. Однако время полного обучения каскадов различается значительно:

|  |  |
| --- | --- |
| featureType: HAAR | featureType: LBP |
| ===== TRAINING 5-stage =====  <BEGIN  POS count : consumed 1100 : 1113  NEG count : acceptanceRatio 4617 : 3.13382e-05  Precalculation time: 5.091  +----+---------+---------+  | N | HR | FA |  +----+---------+---------+  | 1| 1| 1|  +----+---------+---------+  | 2| 1| 1|  +----+---------+---------+  | 3| 1| 0.565735|  +----+---------+---------+  | 4| 1| 0.565735|  +----+---------+---------+  | 5| 1| 0.286766|  +----+---------+---------+  END>  Training until now has taken 0 days **0 hours 41 minutes 35 seconds**.  ===== TRAINING 6-stage =====  <BEGIN  POS count : consumed 1100 : 1113  NEG count : acceptanceRatio 2 : 1.01725e-05  Required leaf false alarm rate achieved. Branch training terminated. | ===== TRAINING 5-stage =====  <BEGIN  POS count : consumed 1100 : 1107  NEG count : acceptanceRatio 4617 : 8.73515e-05  Precalculation time: 0.33  +----+---------+---------+  | N | HR | FA |  +----+---------+---------+  | 1| 1| 1|  +----+---------+---------+  | 2| 1| 1|  +----+---------+---------+  | 3| 1| 0.347412|  +----+---------+---------+  END>  Training until now has taken 0 days **0 hours 2 minutes 8 seconds**.  ===== TRAINING 6-stage =====  <BEGIN  POS count : consumed 1100 : 1107  NEG count : acceptanceRatio 0 : 0  Required leaf false alarm rate achieved. Branch training terminated. |

# 2 Применение каскадного детектора для определения координат

## 2.1 Разработка маркеров специального вида

Для решения задачи навигации были разработаны 2 типа маркеров специального вида, показанных на рисунке Рисунок 6. Все они имеют размеры 30х30 мм.

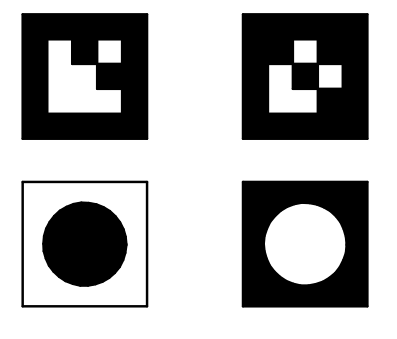


Рисунок – Различные варианты опорных меток

Первый ряд представляет собой ARuCo-подобные маркеры с укрупнёнными внутренними структурными элементами. Стандартный модуль “aruco”, входящий в состав библиотеки OpenCV предлагает пользователю на выбор предопределённые словари меток размерами 4х4, 5х5 или 6х6 клеток. Больший размер внутренних элементов предполагает более устойчивое детектирование или же возможность уменьшить габариты маркера при неизменном расстоянии. Однако же нет никакой необходимости равняться на алгоритмы, заложенные в известную библиотеку, тем более что поставлена задача использовать машинное обучение.

Маркеры второго ряда состоят из ещё более простых геометрических фигур и их комбинаций, именно эти маркеры было решено использовать в дальнейшем.

## 2.2 Описание макета донной зарядной станции

Для удобства тестирования и отладки разрабатываемых алгоритмов был собран небольшой макет донной зарядной станции, показанный на рисунке Рисунок 7. Размеры станции приблизительно соответвуют реальному объекту: усечённая пирамида со стороной нижнего основания в 250 мм, верхнего – в 125 мм и высотой 125 мм.

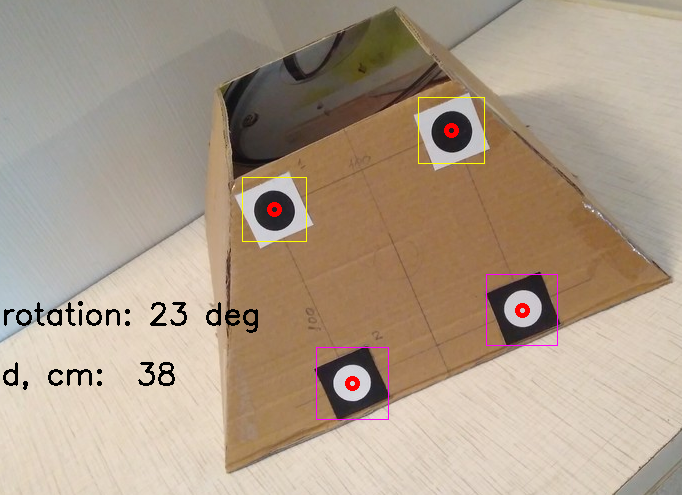


Рисунок – Макет донной зарядной станции

Использованы маркеры размерами 30х30 мм, представленные ранее на рисунке Рисунок 6, они образуют кварат со сторой 100 мм. Каждый каскад (первый - для верхних, второй - для нижних) возвращает массив прямоугольников, внутри которых с определённой вероятностью на изображении находится искомый объект. Как видно из рисунка выше, границы прямоугольников не всегда соответствуют границам объекта, к тому же, объект может иметь сложную форму или быть искажённым в зависимости от угла зрения.

Преимущество данног типа маркеров заключается в наличии круга в центре опорной метки.

## 2.3 Калибровка камеры по расстоянию

Для измерения расстояния по видеопотоку требуется произвести калибровку камеры по расстоянию. С этой целью был изготовлен стенд, повторяющий конфигурацию маркеров ДЗС, и отснята серия изображений, показанных на рисунке 6.

|  |  |
| --- | --- |
|  |  |
|  |  |

Рисунок 6 – Изображения для калибровки камеры. Расстояние на фотографиях сверху составляет 500 мм, снизу – 1000 мм.

Плоскость с маркерами расположена строго перпендикулярно лучу, соединяющему центр объектива и центр квадрата, сторона квадрата составляет 100 мм. У детектированных каскадами маркеров находился геометрический центр, представляющий собой пересечение диагоналей прямоугольника. Расстояния между каждой парой вершин были усреднены, результаты измерения приведены в таблице . Исходные изображения имели высокое разрешение (3840х2160) для уменьшения погрешностей измерения.

Таблица 2 – Результаты калибровки камеры по расстоянию

|  |  |
| --- | --- |
| Расстояние, мм | Значение, пикселей |
| 500 | 600 |
| 1000 | 300 |

В пересчёте по ширине на разрешение 640х480 пикселей имеем 100 px при 500 мм и 50 px при 1000 мм.

## 2.4 Определение относительных смещений

## 2.5 Определение поворота относительно нормали

# СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Метод Виолы-Джонса (Viola-Jones) как основа для распознавания лиц [электронный ресурс]. URL: <https://habr.com/ru/post/133826/> (дата обращения: 10.05.2020).
2. Документация фреймворка OpenCV [электронный ресурс]. URL: <https://docs.opencv.org/4.1.1/dc/d88/tutorial_traincascade.html> (дата обращения: 12.05.2020).
3. Официальная страница разработчиков [электронный ресурс]. URL: <https://www.python.org/> (дата обращения: 14.05.2020).
4. Страница форума stackoverflow.com [электронный ресурс]. URL: <https://stackoverflow.com/questions/50186866/opencv-cascade-training-fast-fails-required-leaf-false-alarm-rate-achieved-br> (дата обращения: 16.05.2020).
5. Страница форума stackoverflow.com [электронный ресурс]. URL: <https://stackoverflow.com/questions/8791178/haar-cascades-vs-lbp-cascades-in-face-detection> (дата обращения: 16.05.2020).

# Приложение А. Сценарий для администрирования большого количества изображений в процессе обучения каскадного классификатора

abs\_path = 'Your\_absolute\_path\_to\_base\_folder/'

which\_file = 2

if which\_file == 1:

f = open(abs\_path+'good\_1.dat')

f\_fixed = open(abs\_path+'good\_1\_fixed.dat', 'a')

else:

f = open(abs\_path + 'good\_2.dat')

f\_fixed = open(abs\_path + 'good\_2\_fixed.dat', 'a')

data = f.readlines()

print(type(data))

for line in data:

first\_part = line.strip().split()[0]

#print(first\_part)

file\_name = first\_part.split("\\")[-1]

print(file\_name)

if which\_file == 1:

try:

shutil.move(abs\_path+"from\_folder/"+file\_name,

abs\_path+" /to\_folder/"+file\_name)

line = line.strip().split('\\')

new\_line = line[0] + "/to\_folder/" + line[1] + "\n"

print(new\_line, end='')

f\_fixed.write(new\_line)

except FileNotFoundError:

print(file\_name, " does not exist")

input()

else:

try:

shutil.move(abs\_path+"from\_folder\_2/"+file\_name,

abs\_path+"to\_folder\_2/"+file\_name)

line = line.strip().split('\\')

new\_line = line[0] + "/to\_folder\_2/" + line[1] + "\n"

print(new\_line, end='')

f\_fixed.write(new\_line)

except FileNotFoundError:

print(file\_name, " does not exist")

input()

# Приложение Б. Сценарий для наполнения файла фоновых изображений

import os

f = open('bad.dat', 'w')

for i in range(1, 2956):

s = './extracted\_images/without\_800/advanced' + '{:04d}'.format(i) + ".jpg\n"

f.write(s)

# print(s, end = '')