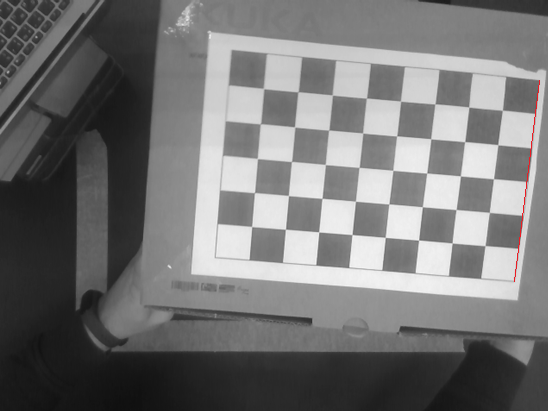
# Распознавание объектов по цвету

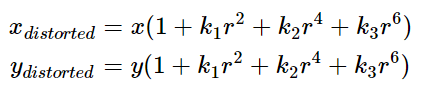
**Калибровка камеры**

Некоторые камеры вносят значительные искажения в изображения. Двумя основными видами искажений являются радиальное искажение и тангенциальное искажение.

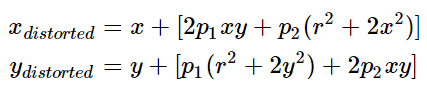
Из-за радиального искажения прямые линии выглядят изогнутыми. Радиальное искажение тем больше, чем дальше точки от центра изображения. Например, на рис.1.1 показано одно изображение, на котором два края шахматной доски отмечены красными линиями. Но вы можете видеть, что граница шахматной доски не является прямой линией и не совпадает с красной линией.

  
Рис.1.1 – Искажение камеры

Радиальное искажение можно представить следующим образом:



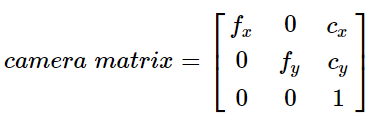
Точно так же тангенциальное искажение возникает из-за того, что объектив, принимающий изображение, не выровнен идеально параллельно плоскости изображения. Таким образом, некоторые области изображения могут выглядеть ближе, чем ожидалось. Величину тангенциального искажения можно представить следующим образом:



Нам нужно найти пять параметров, известных как коэффициенты искажения, определяемые как:



В дополнение к этому нам нужна другая информация, например, внутренние и внешние параметры камеры. Внутренние параметры специфичны для камеры. Они включают такую ​​информацию, как фокусное расстояние () и оптические центры (). Матрица камеры уникальна для конкретной камеры, поэтому после расчета ее можно повторно использовать для других изображений, снятых той же камерой. Она выражается в виде матрицы 3x3:



Внешние параметры соответствуют векторам вращения и перемещения, которые переводят координаты 3D-точки в систему координат.

Чтобы найти эти параметры, необходимо предоставить несколько образцов изображений четко определенного шаблона (например, шахматной доски). Находим некоторые определенные точки, относительные положения которых уже известны (например, квадратные углы на шахматной доске). Мы знаем координаты этих точек в реальном пространстве и знаем координаты на изображении, поэтому можем вычислить коэффициенты искажения. Для лучших результатов нам нужно как минимум 10 тестовых шаблонов.

Для калибровки камеры нам потребуется не менее 10 тестовых шаблонов. Рассмотрим изображение шахматной доски. Важными входными данными, необходимыми для калибровки камеры, являются набор трехмерных точек реального мира и соответствующие двухмерные координаты этих точек на изображении. Точки 2D-изображения в порядке, которые можно легко найти на изображении. (Эти точки изображения представляют собой места, где два черных квадрата касаются друг друга на шахматной доске)

Нужно знать X,Y,Z, но для простоты можно сказать, что шахматная доска оставалась неподвижной в плоскости XY (поэтому Z=0 всегда) и камера перемещалась соответственно. Это соображение помогает нам найти только значения X, Y. Теперь для значений X, Y можно просто передавать точки как (0,0), (1,0), (2,0),... что обозначает расположение точек. В этом случае полученные результаты будут в масштабе размера квадрата шахматной доски. Зная размер квадрата (скажем, 30 мм), можно передать значения как (0,0), (30,0), (60,0),... . Таким образом, получить результаты в мм.

Точки 3D называются точками объекта, а точки 2D-изображения называются точками изображения.

Таким образом, чтобы найти шаблон на шахматной доске, можно использовать функцию cv.findChessboardCorners(). Необходимо указать, какой шаблон ищем, например, сетка 8x8, сетка 5x5 и т. д. Обычно шахматная доска имеет квадраты 8x8 и внутренние углы 7x7. Он возвращает угловые точки и retval, которые будут True, если шаблон получен. Эти углы будут расположены в порядке (слева направо, сверху вниз)

Как только углы будут найдены, можно повысить их точность, используя cv.cornerSubPix() и нарисовать шаблон, используя cv.drawChessboardCorners().

Одно изображение с нарисованным на нем узором показано на рис.1.2:

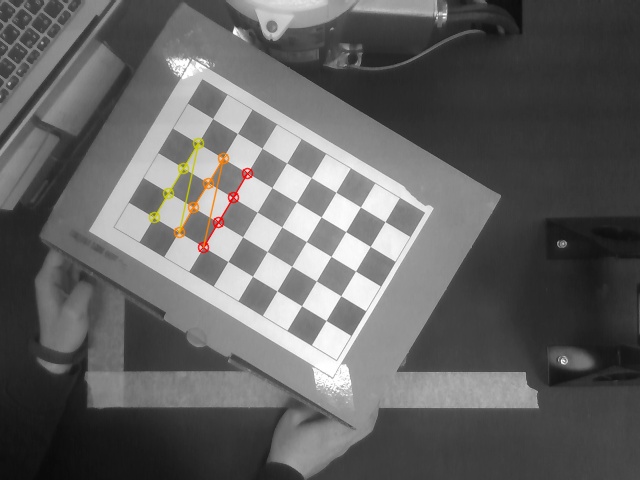


Рис.1.2 – Отображение углов шахматки

**Калибровка**

Следующим этапом необходимо произвести калибровку. Для этого используется функция cv.calibrateCamera(), которая возвращает матрицу камеры, коэффициенты искажения, векторы поворота и перемещения и т. д.

В результате в окне вывода появляются значения, как на рис.1.3.

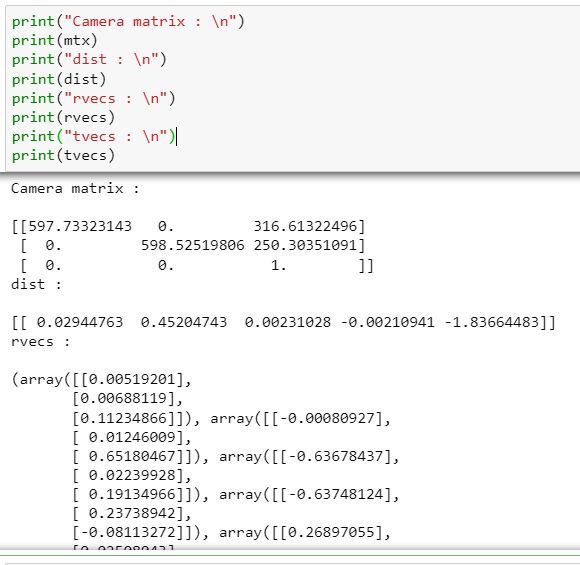


Рис.1.3 – Значения калибровки камеры

# Реализация детектирования объекта по цвету

OpenCV позволяет детектировать объект используя либо его форму, либо цвет. В данной работе реализуется способ детектирования объектов по цвету. В качестве объектов используются разноцветные кубики 40х40х40 мм.

Цветовым пространством по умолчанию в OpenCV является RGB. Сегментация изображений в цветовом пространстве RGB сложная задача, т.к. цвета пересекаются и часто перекрывают друг друга. На рис. 15 представлен график цветового рассеивания рисунка 1.4 в формате RGB.



Рис.1.4 - Исходное изображение полученное с камеры

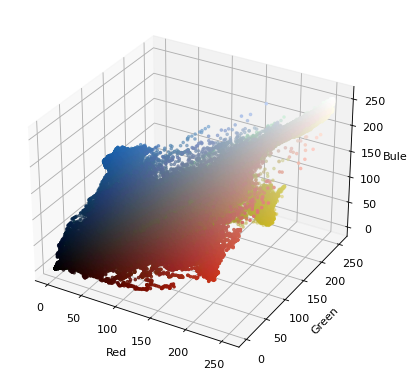


Рис.1.5 - График цветового рассеивания RGB

На графике видно, что оранжевые элементы изображения распределены по всем значениям красного, зеленого и синего. Таким образом сегментация в пространстве RGB не целесообразна. В этой связи принято решение осуществлять сегментацию изображений в цветовом пространстве HSV (Hue, Saturation и Value).

HSV - цилиндрическое цветовое пространство. Цвета, или оттенки, меняются при движении по кругу цилиндра. Вертикальная ось определяет яркость: от темного (0 в нижней части) до светлого сверху. Третья ось - насыщенность, определяет тени оттенков при движении от центра к краю вдоль радиуса цилиндра (от менее к более насыщенному).

График в цветовом пространстве HSV того же изображения представлен на рис 1.6.

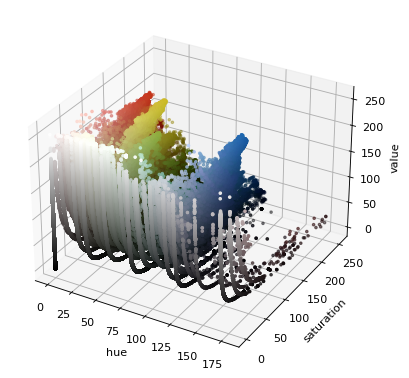


Рис.1.6 - График цветового рассеивания HSV

В пространстве HSV цвета кубиков более локализованы и визуально различимы.

Чтобы корректно разделить цветовой диапазон для детектирования каждого цвета написана программа, позволяющая в реальном времени на видео-потоке настраивать диапазон пространства HSV под каждый цвет. Пример детектирования объекта розового цвета в программе представлен на рис.1.7.

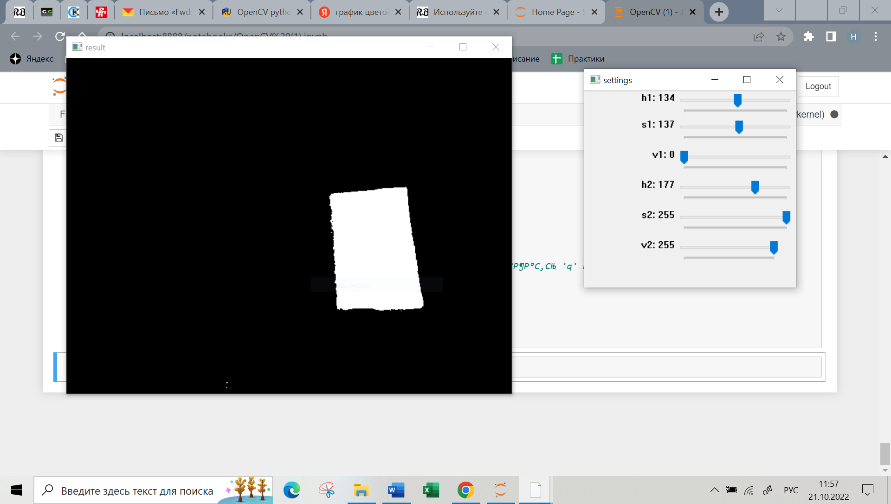


Рис. 1.7 - Детектирование объекта розового цвета в цветовом пространстве HSV

В результате определены диапазоны для детектирования каждого цвета. Полученные результаты, представленные в таблице 1.1.

Таблица 1.1 - Распределение цветов в цветовом пространстве HSV

| Цвет | Hue | Saturation | Value |
| --- | --- | --- | --- |
| Красный | 0-16 | 158-255 | 0-255 |
| Желтый | 17-35 | 158-255 | 0-255 |
| Зеленый | 36-52 | 158-255 | 0-255 |
| Синий | 66-108 | 158-255 | 0-255 |

Для обеспечения детектирования объектов всех вышеперечисленных цветов необходимо сложить полученные битовые маски.

### **Распознавание объектов по цвету**

#### **Вариант 1: "Калибровка и детектирование красных объектов"**

**Задача:** Откалибровать камеру и написать программу для обнаружения красных кубиков.

**Специфика:**

* Целевой цвет: **Красный** (Hue: 0-16, Saturation: 158-255, Value: 0-255).
* Реализовать вывод на изображении bounding box и координат центра каждого красного объекта.

#### **Вариант 2: "Детектирование желтых объектов и фильтрация по площади"**

**Задача:** Настроить детектирование желтых объектов с фильтрацией по минимальной площади.

**Специфика:**

* Целевой цвет: **Желтый** (Hue: 17-35).
* Реализовать отсев шумов путем фильтрации контуров по площади (игнорировать объекты площадью менее 500 пикселей).
* Вывести на изображение только отфильтрованные объекты с их нумерацией.

#### **Вариант 3: "Система сортировки по цветам"**

**Задача:** Реализовать одновременное детектирование объектов двух цветов с разной маркировкой.

**Специфика:**

* Целевые цвета: **Зеленый** (Hue: 36-52) и **Синий** (Hue: 66-108).
* Обозначать зеленые объекты прямоугольником, синие — кругом.
* Выводить рядом с каждым объектом текст с его цветом и координатами в пикселях.

#### **Вариант 4: "Поиск самого крупного объекта заданного цвета"**

**Задача:** Найти самый большой красный объект в кадре.

**Специфика:**

* Целевой цвет: **Красный**.
* Реализовать поиск контура с максимальной площадью.
* Выделить его на изображении особым цветом (например, белым) и вывести его координаты и площадь.

### ***Вариант 5: "Анализ влияния освещения на цветовое детектирование"***

**Задача:** Исследовать устойчивость цветового детектирования к изменению условий освещения.

**Специфика:**

* Целевой цвет: **Зеленый** (Hue: 36-52, Saturation: 158-255, Value: 0-255)
* Провести серию экспериментов с различными условиями освещения:
  1. Стандартное освещение
  2. Затемненная сцена
  3. Яркая засветка
  4. Измененный цветовой фон (теплый/холодный свет)
* Для каждого условия:
  1. Зафиксировать количество ложноположительных и ложноотрицательных срабатываний
  2. Измерить стабильность координат детектируемого объекта
  3. Сделать скриншоты результатов

#### **Вариант 6: "Подсчет количества объектов определенного цвета"**

**Задача:** Реализовать подсчет количества зеленых кубиков в кадре.

**Специфика:**

* Целевой цвет: **Зеленый**.
* Выводить в углу кадра текст с количеством обнаруженных объектов.

#### **Вариант 7: "Определение центра масс группы объектов"**

**Задача:** Найти среднюю координату для всех синих объектов в кадре.

**Специфика:**

* Целевой цвет: **Синий**.
* Рассчитать среднее арифметическое координат X и Y всех обнаруженных синих объектов.
* Отобразить на изображении этот "центр масс" большой красной точкой.

#### **Вариант 8: "Детектирование в режиме реального времени с записью координат"**

**Задача:** Написать программу, которая записывает координаты красных объектов в файл.

**Специфика:**

* Целевой цвет: **Красный**.
* При обнаружении объекта записывать его координаты (X, Y в пикселях) и время обнаружения в текстовый файл.
* Реализовать функцию для запуска и остановки записи по нажатию клавиш.

#### **Вариант 9: "Сравнение эффективности RGB и HSV"**

**Задача:** Реализовать детектирование желтого цвета в RGB и HSV для сравнения.

**Специфика:**

* Реализовать два режима детектирования **желтого** цвета: по RGB-диапазону и по HSV-диапазону.
* Выводить результаты обоих методов рядом.
* Сделать вывод о эффективности каждого метода на основе визуального сравнения.

### ***Вариант 10: "Система контроля наличия объектов на конвейере"***

**Задача:** Разработать систему мониторинга наличия объектов определенного цвета на движущейся ленте.

**Специфика:**

* Целевой цвет: **Красный** (Hue: 0-16, Saturation: 158-255, Value: 0-255)
* Реализовать детектирование объектов в зоне контроля (заранее определенной области кадра)
* При появлении/исчезновении объекта из зоны контроля:
  + Формировать звуковой сигнал (beep)
  + Выводить текстовое сообщение в консоль ("Объект появился", "Объект отсутствует")
  + Вести лог событий с временными метками