



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 «Информатика и системы управления»

КАФЕДРА «Программное обеспечение ЭВМ и информационные технологии»

## Расчетно-пояснительная записка к курсовому проекту.

Тема Обработка Raw видео.

Студент Нитенко М.Ю.

Группа ИУ7-53Б

Преподаватели (научный руководитель?)

# Оглавление

<b>Введение</b>	<b>3</b>
<b>1 Аналитическая часть</b>	<b>5</b>
1.1 Процесс обработки кадра. . . . .	5
1.2 Формат файла. . . . .	6
1.2.1 DNG . . . . .	6
1.3 Алгоритмы демозаики. . . . .	7
1.3.1 Билинейная интерполяция . . . . .	8
1.3.2 Malvar-He-Cutler . . . . .	8
1.4 Цветовая модель. . . . .	10
1.4.1 CIE XYZ. . . . .	10
1.4.2 ProPhoto RGB. . . . .	11
1.5 Преобразования цветовой модели. . . . .	12
1.6 Настройка изображения. . . . .	12
1.6.1 Яркость. . . . .	13
1.6.2 Контрастность. . . . .	13
1.6.3 Насыщенность. . . . .	14
1.6.4 Баланс цветов. . . . .	15
<b>2 Конструкторская часть</b>	<b>18</b>
2.1 Требования к программному обеспечению . . . . .	18
2.1.1 Алгоритм билинейной интерполяции . . . . .	18
2.1.2 Алгоритм интерполяции Malvar-He-Cutler . . . . .	20
2.1.3 Диаграмма классов . . . . .	22
2.1.4 IDEF0 . . . . .	23
<b>3 Технологическая часть</b>	<b>24</b>
3.1 Средства реализации . . . . .	24
3.2 Реализация алгоритмов . . . . .	24
<b>4 Исследовательская часть</b>	<b>26</b>

<b>Заключение</b>	<b>27</b>
<b>Литература</b>	<b>28</b>

# Введение

RAW видео — это видео содержащее необработанную информацию об изображении с сенсора камеры.

Главный элемент цифровых камер — сенсор, при попадании света на сенсор на нем накапливается заряд. Из этих зарядов формируется изображение.

Однако без дополнительных средств любой свет воспринимается сенсором одинаково, и на выходе получается черно-белое изображение. Наиболее распространенными способами записи цветного изображения в одну экспозицию являются: фильтр Байера, над одной матрицей или разделение изображения на три цвета, красный, зеленый и синий, и обработка каждого из них отдельной матрицей. [1]

Несмотря на то что метод разделения на три матрицы дает наиболее качественный результат, в большинстве камер среднего ценового сегмента установлена одна матрица с фильтром Байера.

Фильтр Байера состоит из 25% красных элементов, 25% синих и 50% зеленых элементов, как показано на рисунке 1.

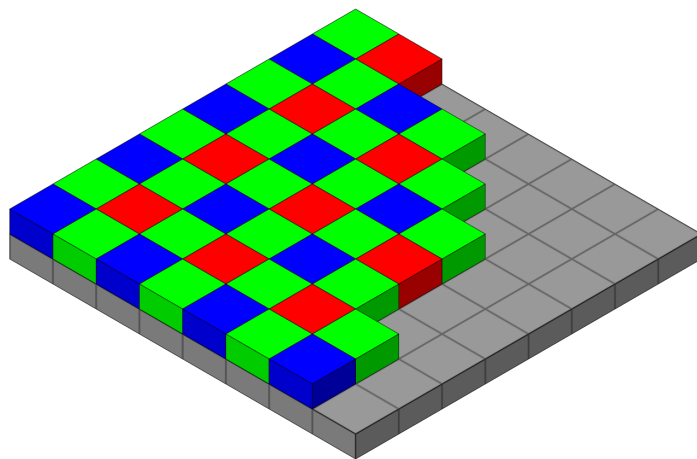


Рисунок 1 – Фильтр Байера.

Изображение с такого фильтра дает возможность создания цветного изображения, однако без обработки оно не будет таковым. Поэтому необходимо произвести процесс демозаики, который приведет изображение к корректному виду.

После этого можно приступать к остальным настройкам изображения, таким как: преобразование цвета, настройка баланса белого, тональных

кривых, контрастности, насыщенности и так далее.

Так как речь идет об обработке видео, необходимо использовать быстрые алгоритмы, чтобы обеспечивать корректную частоту кадров.

Таким образом, цель данной работы — реализовать ПО позволяющее просматривать, обрабатывать и сохранять RAW видео.

Чтобы достигнуть поставленной цели, требуется решить следующие задачи:

- реализовать открытие и отображение RAW файлов;
- реализовать инструменты для обработки видео;
- реализовать возможность сохранения модифицированного видео;
- реализовать пользовательский интерфейс.

# 1 Аналитическая часть

В данном разделе описаны необходимые для обработки данных с сенсора алгоритмы.

## 1.1 Процесс обработки кадра.

Raw-кадр является набором значений с матрицы, поэтому для показа без обработки не пригоден.

Типичная обработка включает в себя:

- декодинг данных, например если каждому пикселю соответствуют 14 бит информации они, скорее всего, лежат последовательно и их придется декодировать;
- демозаика, то есть устранение фильтра Байера;
- преобразование цвета из пространства цвета камеры в общепринятое;
- изменение гаммы и прочих параметров изображения.

На изображении 1.1 показан ожидаемый результат обработки.

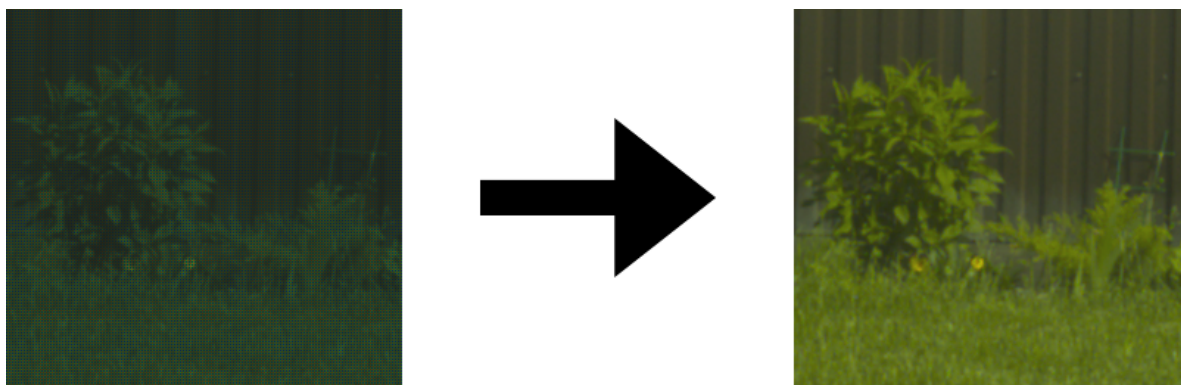


Рисунок 1.1 – Ожидаемый результат обработки. Изображение интерполировано при помощи билинейной интерполяции, проведена коррекция цвета, увеличена яркость и контрастность изображения.

## 1.2 Формат файла.

В таблице 1.2 указана необходимая информация о изображении для каждого из этапов.

Таблица 1.1 – Используемая информация о изображении.

Этап обработки	Необходимая информация
Распаковка	BitsPerSample
Предварительная обработка	WhiteLevel, BlackLevel
Демозаика	CFALayout
Коррекция цвета	ColorMatrix, CameraCalibration

### 1.2.1 DNG

Digital Negative (DNG) — формат хранения raw-изображений, основан на формате изображений TIFF. CinemaDNG является форматом хранения raw-видео и представляет собой набор DNG файлов.

DNG хранит в себе изображение и набор метаданных, для обработки кадра используются следующие поля (TIFF tags):

- BitsPerSample — количество битов для описания каждого сэмпла (пикселя), поддерживаемые значения от 8 до 32 бит на семпл. Если BitsPerSample не равен 8 или 16 или 32, тогда биты должны быть упакованы в байты с использованием стандартного порядка для TIFF FillOrder 1 (big-endian);
- BlackLevel — уровень черного, все значения меньше или равные ему считаются минимальными. Тип может быть SHORT, LONG или RATIONAL, Tag = 50714;
- WhiteLevel — уровень белого, все значения больше или равные ему считаются максимальными. Тип может быть SHORT или LONG, Tag = 50717;

- CFALayout — поле описывающее поддерживаемые форматы фильтра Байера. Тип — SHORT, Tag = 50711;
- ColorMatrix — матрица для преобразования из цветового пространства XYZ в цветовое пространство камеры. Тип — SRATIONAL, Tag = 50721;
- CameraCalibration — матрица для преобразования из идеального цветового пространства камеры в цветовое пространство конкретной камеры. Тип — SRATIONAL, Tag = 50723. [2]

Для обработки файлов можно использовать библиотеку «Tiny DNG Loader», эта библиотека небольшая по размеру и поддерживает необходимые поля.

### 1.3 Алгоритмы демозаики.

Одной из главных задач обработки RAW видео является устранение эффектов фильтра Байера. Схематичное изображение фильтра Байера показано на рисунке 1.2, а на изображении 1.3 показано изображение с сенсора камеры, содержащее этот эффект.

1	2	3	4	5
6	7	8	9	10
11	12	13	14	15
16	17	18	19	20
21	22	23	24	25

Рисунок 1.2 – Пронумерованный фильтр Байера.



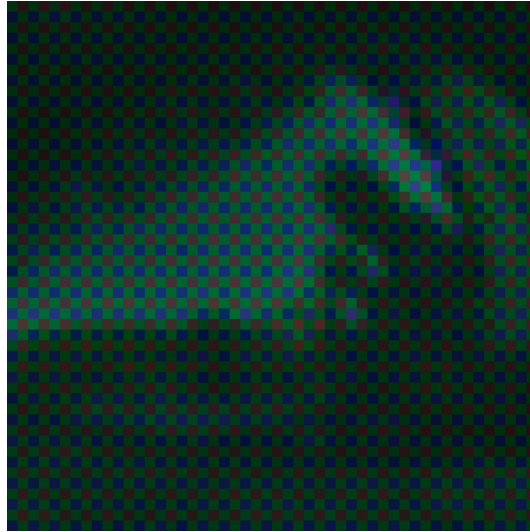


Рисунок 1.3 – Изображение с сенсора камеры.

### 1.3.1 Билинейная интерполяция

Билинейная интерполяция использует среднее значение четырех соседних пикселей соответствующего цвета, например: значения зеленого пикселя для красных или синих пикселей находятся по 1.1:

$$\hat{G}^{bl}(i, j) = \frac{1}{4}(G(i-1, j) + G(i+1, j) + G(i, j-1) + G(i, j+1)) \quad (1.1)$$

где  $G(x, y)$  — значение зеленого цвета в пикселе  $x, y$ . [4]

Билинейная интерполяция красного и синего канала похожи на интерполяцию зеленого, но используют пиксели лежащие по диагонали от интерполируемого.

Данный алгоритм считается одним самых быстрых и часто используется для интерполяции видео в реальном времени. [3]

### 1.3.2 Malvar-He-Cutler

Метод является улучшением билинейной интерполяции.

Улучшение достигается при помощи использования Laplacian cross-channel correction (<– еще не придумал как перевести, если поможете буду благодарен).

Зеленый канал для красного пикселя вычисляется как 1.2

$$\hat{G}(i, j) = \hat{G}^{bl}(i, j) + \alpha \Delta_R(i, j) \quad (1.2)$$

где  $\Delta_R$  — дискретный лапласиан красного канала по 5 точкам. По формуле 1.3.

$$\Delta_R(i, j) = R(i, j) - \frac{1}{4}(R(i-2, j) + R(i+2, j) + R(i, j-2) + R(i, j+2)) \quad (1.3)$$

Красный канал для зеленого пикселя вычисляется как 1.4

$$\hat{R}(i, j) = \hat{R}^{bl}(i, j) + \beta \Delta_G(i, j) \quad (1.4)$$

где  $\Delta_G$  — дискретный лапласиан зеленого канала, по 9 точкам.

Красный канал для синего пикселя вычисляется как 1.5

$$\hat{R}(i, j) = \hat{R}^{bl}(i, j) + \gamma \Delta_B(i, j) \quad (1.5)$$

где  $\Delta_B$  — дискретный лапласиан синего канала по 5 точкам.

Синие компоненты высчитываются так же, как и для красного.

Параметры  $\alpha$ ,  $\beta$  и  $\gamma$  отвечают за силу корректировки, оптимальные значения рассчитаны авторами алгоритма [4]:

$$\alpha = \frac{1}{2}, \quad \beta = \frac{5}{8}, \quad \gamma = \frac{3}{4} \quad (1.6)$$

Демозаика производится при помощи линейных фильтров, существуют 8 различных фильтров, они показаны на рисунке 1.4

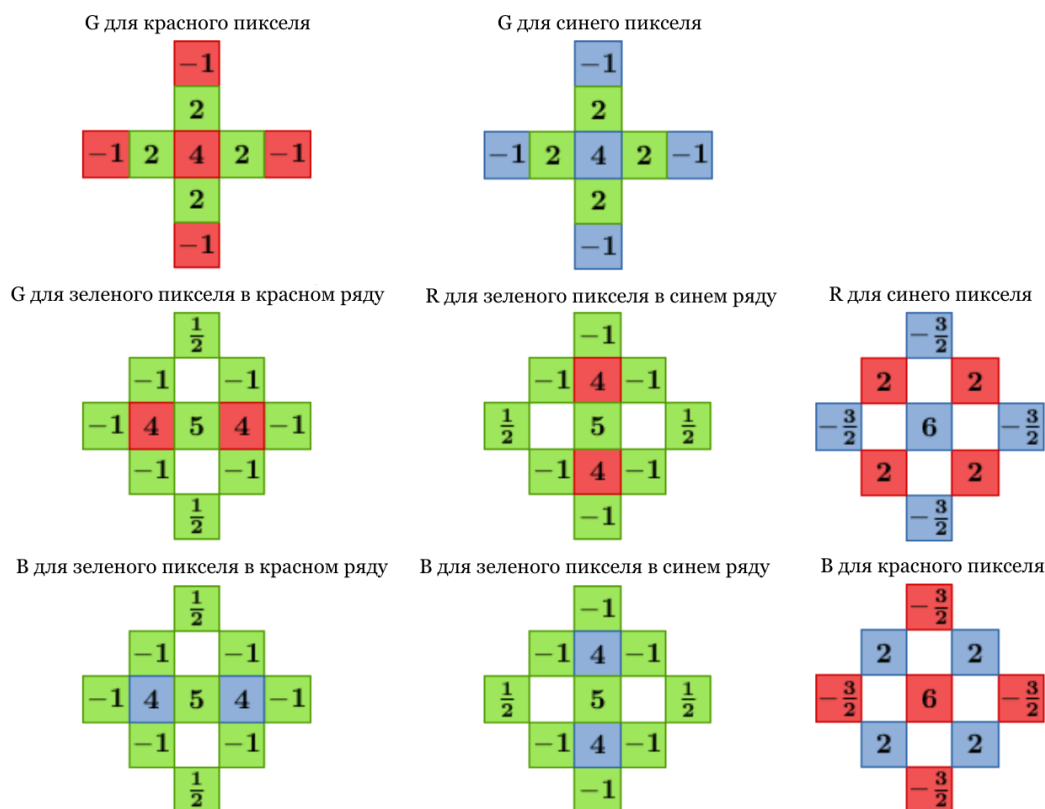


Рисунок 1.4 – Используемые фильтры, коэффициенты домножены на 8.

## 1.4 Цветовая модель.

Цветовая модель — это математическая модель описания представления цветов в виде кортежей чисел, называемых цветовыми компонентами или цветовыми координатами. Изображение с матрицы находится в цветовой модели камеры и для правильного представления картинки необходимо преобразование цветов. Например, формат DNG хранит в себе матрицу для преобразования изображения из цветовой модели камеры в CIE XYZ D50. Однако не все камеры используют свое цветовое пространство, например для камеры Canon 650D эта матрица — единичная.

### 1.4.1 CIE XYZ.

В цветовой модели CIE XYZ каждый элемент кортежа примерно соответствует одной из колбочек человеческого глаза: X — длинноволновым,

$Y$  — средневолновым и  $Z$  — коротковолновым. На рисунке 1.5 показана хроматическая диаграмма модели.

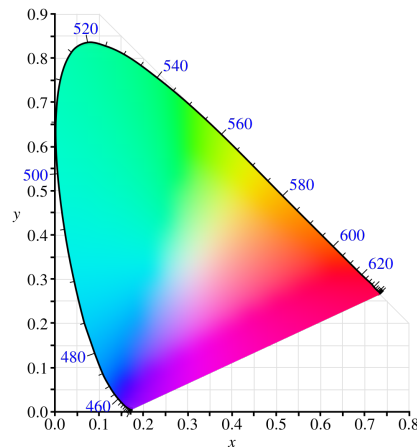


Рисунок 1.5 – Хроматическая диаграмма модели CIE XYZ.

### 1.4.2 ProPhoto RGB.

Цветовая модель в которой значения кортежа означают значения основных цветов: красного, зеленого и синего. Остальные цвета получаются сочетанием базовых. Цвета такого типа называются аддитивными.

Цветовая модель ProPhoto RGB покрывает 90% возможных цветов модели CIELAB и является рекомендованной в спецификации DNG цветовой моделью. [2]

На рисунке 1.6 показано цветовое покрытие, по сравнению с CIE XYZ.

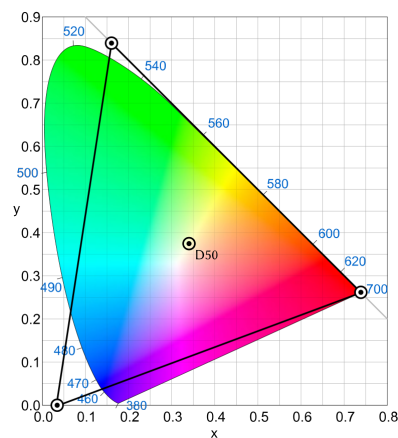


Рисунок 1.6 – Хроматическая диаграмма модели ProPhoto RGB.

## 1.5 Преобразования цветовой модели.

Для преобразования часто используются матрицы. Пусть  $CM$  — матрица преобразующая XYZ D50 в цветное пространство камеры, тогда  $CM^{-1}$  будет матрицей переводящей цветное пространство матрицы в XYZ D50. Пусть  $XTP$ :

$$XTP = \begin{bmatrix} 1.3460 & -0.2556 & -0.0511 \\ -0.5446 & 1.5082 & 0.0205 \\ 0.0 & 0.0 & 1.2123 \end{bmatrix} \quad (1.7)$$

матрица преобразующая XYZ D50 в ProPhoto RGB. Тогда для преобразования изображения из цветного пространства камеры в ProPhoto RGB необходимо произвести умножение:

$$T_{ProPhotoRGB} = XTP * CM * T_{CC} \quad (1.8)$$

где  $T_{CC}$  — кортеж с значениями цвета в пространстве камеры.

$T_{ProPhotoRGB}$  находится в цветном пространстве ProPhoto RGB, но яркость все еще закодирована линейно, для правильного отображения необходимо применить гамма-коррекцию:

$$\gamma(u) = \begin{cases} 16u, & u \leq 0.001953 \\ u^{1/1.8} & \end{cases} \quad (1.9)$$

где  $u$  — одна из компонент цвета. [5]

## 1.6 Настройка изображения.

Значением яркости в моделях RGB считается среднее значение основных цветов:

$$brv = \frac{R + G + B}{3} \quad (1.10)$$

где  $R, G, B$  — красная, зеленая и синяя компоненты пикселя соответственно. [6]

Контрастность определяется как:

$$C_{ip} = \frac{Li_{max} - Li_{min}}{D} \quad (1.11)$$

где  $Li_{max}$  — максимальное, а  $Li_{min}$  — минимальное значение яркости на изображении.  $D$  — максимальное значение разности  $Li_{max} - Li_{min}$ . [7]

Насыщенностью определяет насколько цвета различаются друг от друга, влияет на красочность изображения. Находится на промежутке от чистого цвета (100%) до серого (0%). [8]

Баланс цветов, в RGB, это соотношение между основными цветами. В модели RGB у серых цвета компоненты цветов должны быть равны ( $R = G = B$ ), то есть быть сбалансированы. В случае если они не равны изображение будет иметь оттенок. [9]

Выбранные параметры позволяют адекватно настроить световое (яркость, контрастность) и цветовое (насыщенность, баланс цветов) отношение между пикселями.

### 1.6.1 Яркость.

Преобразование яркости определяется как:

$$br(a) = a + N \quad (1.12)$$

где  $a$  — значение яркости пикселя, а  $N$  — желаемое увеличение в яркости. [10]

На рисунке 1.7 показан пример увеличения яркости.

Написать типа на сколько все увеличивается.

### 1.6.2 Контрастность.

Преобразование контрастности определяется как:

$$con(a) = a * N \quad (1.13)$$

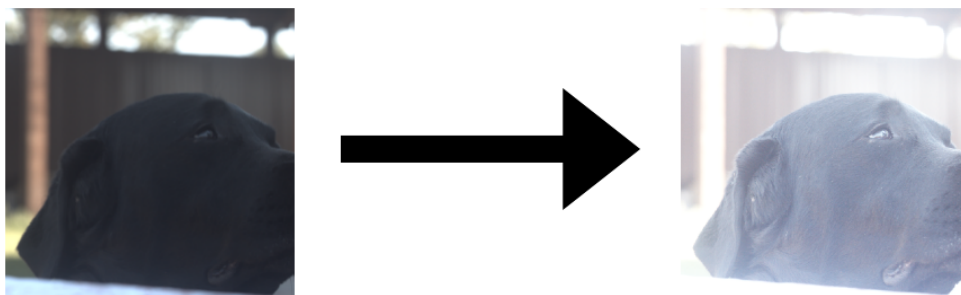


Рисунок 1.7 – Пример увеличения яркости.

где  $a$  — значение яркости пикселя, а  $N$  — желаемое увеличение в контрастности (чтобы, например, увеличить контрастность на 50% необходимо умножить на 1.5). [10]

На рисунке 1.8 показан пример увеличения контрастности.

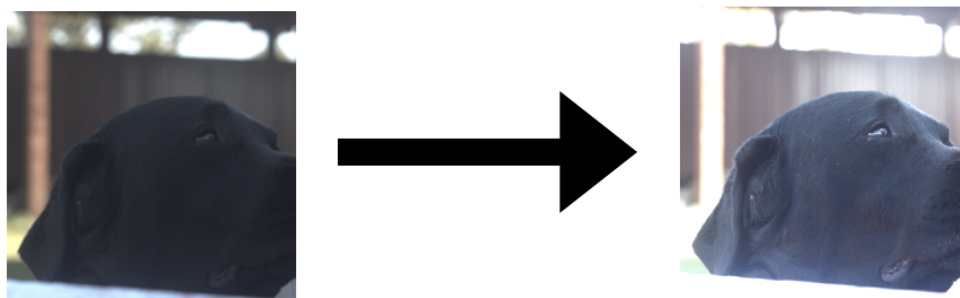


Рисунок 1.8 – Пример увеличения контрастности.

### 1.6.3 Насыщенность.

Для изменения насыщенности изображения в цветовой модели RGB можно воспользоваться умножением матриц.

Пусть  $F(x, y)$  — вектор:

$$F(x, y) = [f_R, f_G, f_B, 1]^T \quad (1.14)$$

где  $f_R, f_G, f_B$  — значения цвета в точке  $x, y$ .

Тогда  $G(x, y)$  — вектор содержащий значения цвета с иной насыщенно-

СТЬЮ:

$$G(x, y) = [g_R, g_G, g_B, g_w]^T \quad (1.15)$$

где  $g_R, g_G, g_B$  — новые значения цвета в точке  $x, y$ , а  $g_w$  не используется.

Вычислить  $G(x, y)$  можно по формуле 1.16:

$$G(x, y) = T * F(x, y) \quad (1.16)$$

где  $T$  — матрица преобразования:

$$T_{sat}(s) = \begin{bmatrix} \alpha + s & \beta & \gamma & 0 \\ \alpha & \beta + s & \gamma & 0 \\ \alpha & \beta & \gamma + s & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (1.17)$$

где  $\alpha = 0.3086(1 - s)$ ,  $\beta = 0.6094(1 - s)$  и  $\gamma = 0.0820(1 - s)$ .

Значения  $s < 1$  приводят к уменьшению насыщенности, значения  $> 1$  — к увеличению. [11]

На рисунке 1.9 показан пример увеличения насыщенности.

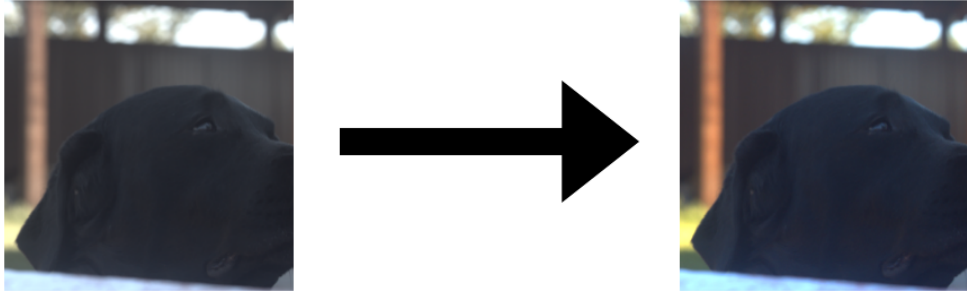


Рисунок 1.9 – Пример увеличения насыщенности.

#### 1.6.4 Баланс цветов.

Для изменения баланса цветов в цветовой модели RGB можно воспользоваться умножением матриц.



Пусть  $F(x, y)$  — вектор:

$$F(x, y) = [f_R, f_G, f_B, 1]^T \quad (1.18)$$

где  $f_R, f_G, f_B$  — значения цвета в точке  $x, y$ .

Тогда  $G(x, y)$  — вектор содержащий значения цвета с иным балансом цвета:

$$G(x, y) = [g_R, g_G, g_B]^T \quad (1.19)$$

где  $g_R, g_G, g_B$  — новые значения цвета в точке  $x, y$ .

Высчитать  $G(x, y)$  можно по формуле 1.20:

$$G(x, y) = T * F(x, y) \quad (1.20)$$

где  $T$  — матрица преобразования:

$$T_{sat}(s) = \begin{bmatrix} R_{wb} & 0 & 0 \\ 0 & G_{wb} & 0 \\ 0 & 0 & B_{wb} \end{bmatrix} \quad (1.21)$$

где  $R_{wb}, G_{wb}$  и  $B_{wb}$  — коэффициенты для каждого из основных цветов.

На рисунке 1.10 показан пример исправления баланса цветов.

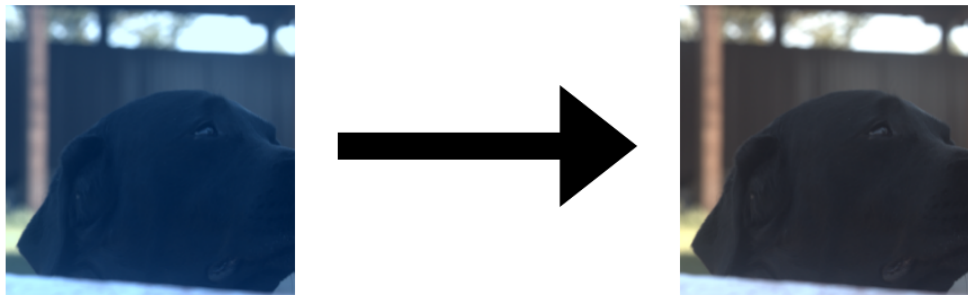


Рисунок 1.10 – Пример восстановления баланса цветов.

## Вывод

В данном разделе был проведен обзор необходимых для реализации алгоритмов. В таблице 1.2 проведена оценка сложности каждого из этапов.

Таблица 1.2 – Используемая информация о изображении.

Этап обработки	Оценка сложности (лучший случай — худший случай)
Распаковка	$2 * \text{высота} + 9N - 2 * \text{высота} + 41N$
Предварительная обработка	$2 * \text{высота} + 14N$
Демозаика	$2 * \text{высота} + 75N - 2 * \text{высота} + 729N$
Коррекция цвета	$2 * \text{высота} + 74N$

## 2 Конструкторская часть

В данном разделе представлены требования к программному обеспечению, а также схемы выбранных для решения поставленной задачи алгоритма.

### 2.1 Требования к программному обеспечению

Программа должна предоставлять доступ к функционалу:

- открытие DNG последовательностей;
- настройка параметров изображения;
- выбор показываемого кадра
- проигрывание кадров;
- сохранение обработанного видео (<— не успею наверное...).

К программе предъявляются следующие требования:

- время обработки кадра должно быть менее 1/30 секунды для проигрывания основных частот кадров.

#### 2.1.1 Алгоритм билинейной интерполяции

---

**Алгоритм 1** Билинейная интерполяция

---

```
1:  $in \leftarrow$  вектор с данными с сенсора камеры
2:  $height \leftarrow$  высота изображения  $DomColor$ 
3:  $width \leftarrow$  ширина изображения
4:  $out \leftarrow$  вектор с интерполированными цветами
5: for  $y \leftarrow 0$  to  $height$  do
6:   for  $x \leftarrow 0$  to  $width$  do
7:      $C \leftarrow$  значение цвета в точке  $x, y$ 
8:     if пиксель красный then
9:        $out_{3*(y*width+x)+0} \leftarrow C$ 
10:       $out_{3*(y*width+x)+1} \leftarrow (in_{x,y-1} + in_{x,y+1} + in_{x-1,y} + in_{x+1,y})/4$ 
11:       $out_{3*(y*width+x)+2} \leftarrow (in_{x-1,y-1} + in_{x-1,y+1} + in_{x+1,y-1} +$ 
 $in_{x+1,y+1})/4$ 
12:     end if
13:     if пиксель зеленый И соседи красные then
14:        $out_{3*(y*width+x)+0} \leftarrow (in_{x-1,y} + in_{x+1,y})/2$ 
15:        $out_{3*(y*width+x)+1} \leftarrow C$ 
16:        $out_{3*(y*width+x)+2} \leftarrow (in_{x,y-1} + in_{x,y+1})/2$ 
17:     end if
18:     if пиксель зеленый И соседи синие then
19:        $out_{3*(y*width+x)+0} \leftarrow (in_{x,y-1} + in_{x,y+1})/2$ 
20:        $out_{3*(y*width+x)+1} \leftarrow C$ 
21:        $out_{3*(y*width+x)+2} \leftarrow (in_{x-1,y} + in_{x+1,y})/2$ 
22:     end if
23:     if пиксель синий then
24:        $out_{3*(y*width+x)+0} \leftarrow (in_{x-1,y-1} + in_{x-1,y+1} + in_{x+1,y-1} +$ 
 $in_{x+1,y+1})/4$ 
25:        $out_{3*(y*width+x)+1} \leftarrow (in_{x,y-1} + in_{x,y+1} + in_{x-1,y} + in_{x+1,y})/4$ 
26:        $out_{3*(y*width+x)+2} \leftarrow C$ 
27:     end if
28:   end for
29: end for
```

---

## 2.1.2 Алгоритм интерполяции Malvar-He-Cutler

---

### Алгоритм 2 Malvar-He-Cutler

---

```
1:  $in \leftarrow$  вектор с данными с сенсора камеры
2:  $height \leftarrow$  высота изображения  $DomColor$ 
3:  $width \leftarrow$  ширина изображения
4:  $out \leftarrow$  вектор с интерполированными цветами
5:  $\alpha \leftarrow \frac{1}{2}$ 
6:  $\beta \leftarrow \frac{5}{8}$ 
7:  $\gamma \leftarrow \frac{3}{4}$ 
8: for  $y \leftarrow 0$  to  $height$  do
9:   for  $x \leftarrow 0$  to  $width$  do
10:      $C \leftarrow$  значение цвета в точке  $x, y$ 
11:      $lapl \leftarrow C - (in_{x-2,y} + in_{x+2,y} + in_{x,y-2} + in_{x,y+2})/4$ 
12:     if пиксель красный then
13:        $out_{3*(y*width+x)+0} \leftarrow C$ 
14:        $out_{3*(y*width+x)+1} \leftarrow (in_{x,y-1} + in_{x,y+1} + in_{x-1,y} + in_{x+1,y})/4 + \alpha * lapl$ 
15:        $out_{3*(y*width+x)+2} \leftarrow (in_{x-1,y-1} + in_{x-1,y+1} + in_{x+1,y-1} + in_{x+1,y+1})/4 + \alpha * lapl$ 
16:     end if
17:     if пиксель синий then
18:        $out_{3*(y*width+x)+0} \leftarrow (in_{x-1,y-1} + in_{x-1,y+1} + in_{x+1,y-1} + in_{x+1,y+1})/4 + \gamma * lapl$ 
19:        $out_{3*(y*width+x)+1} \leftarrow (in_{x,y-1} + in_{x,y+1} + in_{x-1,y} + in_{x+1,y})/4 + \gamma * lapl$ 
20:        $out_{3*(y*width+x)+2} \leftarrow C$ 
21:     end if
22:      $lapl \leftarrow C - (in_{x-1,y-1} + in_{x-1,y+1} + in_{x,y-2} + in_{x,y+2} + in_{x+1,y-1} + in_{x+1,y+1} + in_{x-2,y} + in_{x+2,y})/8$ 
```

---

---

```

23:      if пиксель зеленый И соседи красные then
24:           $out_{3*(y*width+x)+0} \leftarrow (in_{x-1,y} + in_{x+1,y})/2 + \beta * lapl$ 
25:           $out_{3*(y*width+x)+1} \leftarrow C$ 
26:           $out_{3*(y*width+x)+2} \leftarrow (in_{x,y-1} + in_{x,y+1})/2 + \beta * lapl$ 
27:      end if
28:      if пиксель зеленый И соседи синие then
29:           $out_{3*(y*width+x)+0} \leftarrow (in_{x,y-1} + in_{x,y+1})/2 + \beta * lapl$ 
30:           $out_{3*(y*width+x)+1} \leftarrow C$ 
31:           $out_{3*(y*width+x)+2} \leftarrow (in_{x-1,y} + in_{x+1,y})/2 + \beta * lapl$ 
32:      end if
33:  end for
34: end for

```

---

## 2.1.3 Диаграмма классов

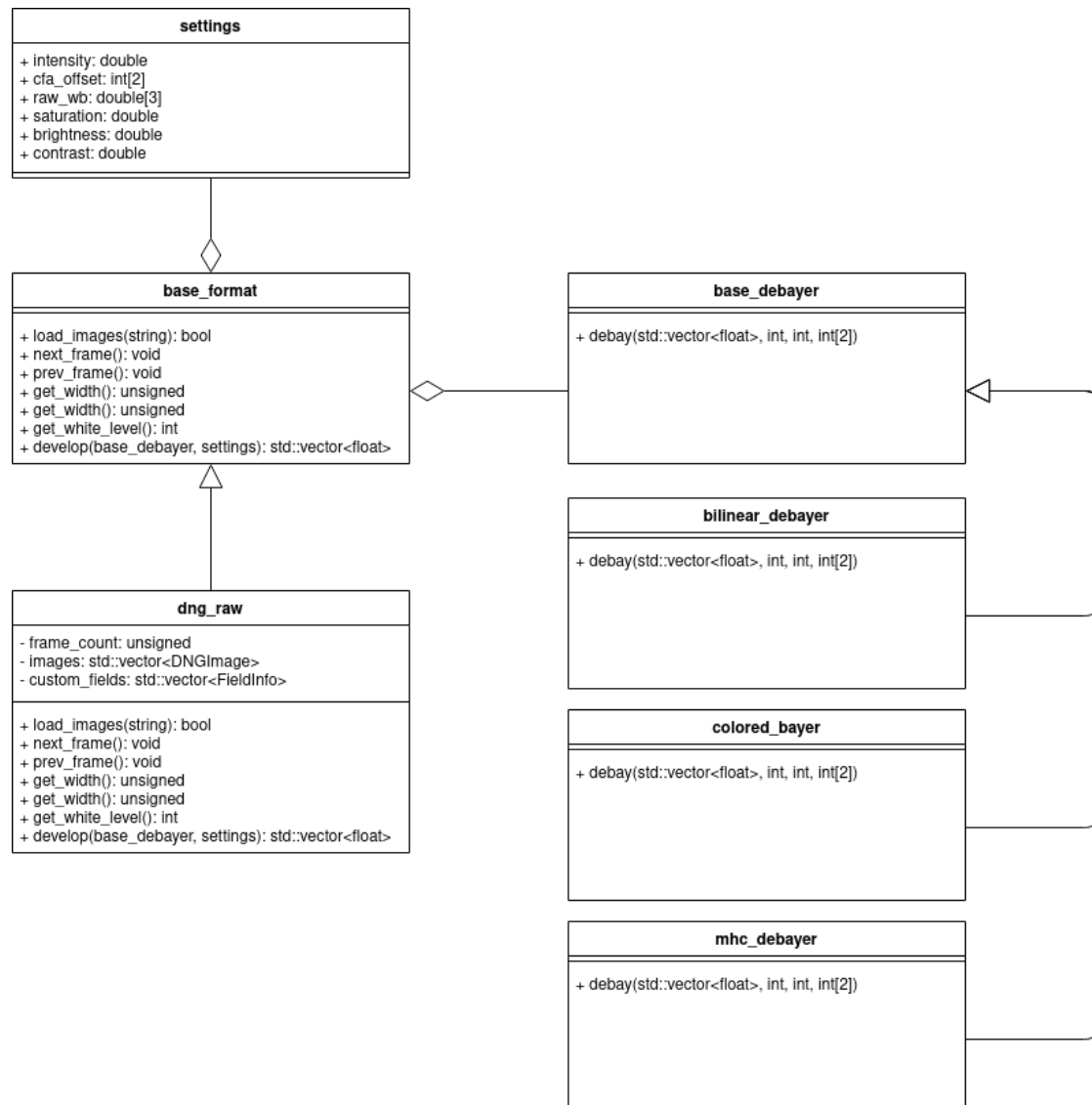


Рисунок 2.1 – Диаграмма классов.

# 2.1.4 IDEF0

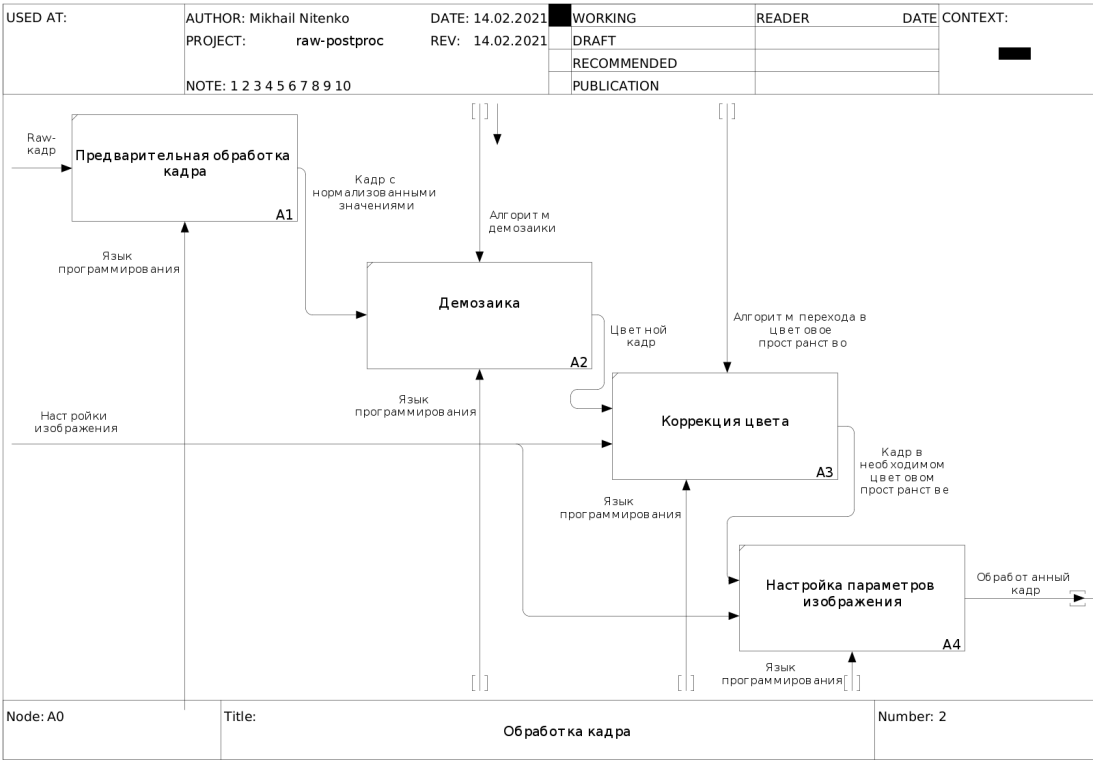


Рисунок 2.2 – Схема декомпозиции задачи обработки кадра.

# Вывод

В данном разделе были представлены требования к программному обеспечению и описаны реализуемые алгоритмы.



## 3 Технологическая часть

В данном разделе представлены средства разработки программного обеспечения, детали реализации и тестирование функций.

### 3.1 Средства реализации

В качестве языка программирования, на котором будет реализовано программное обеспечение, выбран язык программирования C++. Выбор языка обусловлен тем, что на нем написана библиотека Qt, следовательно наиболее полный функционал доступен для этого языка.

Библиотека Qt была выбрана потому что я имею опыт работы с ней, а так же она имеет достаточно обширную возможность работы с изображениями (QImage).

Для работы с DNG-файлами была использована библиотека TinyDNGLoader, она предоставляет доступ ко всем необходимым полям, при этом не является полноценной библиотекой для обработки raw-изображений как, например, libraw.

Для обеспечения качества кода был использован инструмент clang-tidy, позволяющий во время процесса написания исходных кодов программного обеспечения контролировать наличие синтаксических и логических ошибок.

В качестве среды разработки выбран текстовый редактор Qt Creator, он позволяет удобно проектировать Qt интерфейсы, в него интегрирован gdb, а так же он отображает предупреждения clang-tidy прямо во время редактирования.

### 3.2 Реализация алгоритмов

В листинге ?? представлен объект реализующий алгоритм билинейной интерполяции. В листинге ?? представлен объект реализующий алгоритм интерполяции Malvar-He-Cutler.

(ну да давай) (а тести)

## Вывод

## 4 Исследовательская часть

короче фотошоп, там крутим фотку сейвим. потом у меня крутим сейвим, ну и сравниваем получается. на процентик или как.. свой софт штол Е?

или я свой дебайер сделаю  $n^2$  (smooth hue + mhc) и сравниваю типа с этими которые я уже написал.

### Вывод

# Заключение

В ходе выполнения лабораторной работы была проделана следующая работа:

- замерено время выполнения алгоритмов;

# Литература

- [1] Сердце цифровой фотокамеры: ПЗС-матрица (часть четвёртая) [электронный ресурс]. Режим доступа: <https://www.ferra.ru/review/multimedia/71885.htm> (дата обращения: 05.11.2020).
- [2] ADOBE SYSTEMS INCORPORATED. Digital Negative (DNG) Specification.
- [3] Color Properties / Terminology [электронный ресурс]. Режим доступа: <http://www.workwithcolor.com/color-properties-definitions-0101.htm> (дата обращения: 05.02.2021).
- [4] Getreuer Pascal. Malvar-He-Cutler Linear Image Demosaicking // Image Processing On Line. 2011. Т. 1. С. 83–89. [https://doi.org/10.5201/ipol.2011.g\\_mhcd](https://doi.org/10.5201/ipol.2011.g_mhcd).
- [5] ANSI, USA. Specification of ROMM RGB.
- [6] Color FAQ - Frequently Asked Questions Color [электронный ресурс]. Режим доступа: [https://poynton.ca/notes/colour\\_and\\_gamma/ColorFAQ.html#RTFTtoC36](https://poynton.ca/notes/colour_and_gamma/ColorFAQ.html#RTFTtoC36) (дата обращения: 03.02.2021).
- [7] Vladimir Kovalevsky. Modern Algorithms for Image Processing. Springer International Publishing, 2019. Vol. 1. p. 43.
- [8] Color Properties / Terminology [электронный ресурс]. Режим доступа: <http://www.workwithcolor.com/color-properties-definitions-0101.htm> (дата обращения: 05.02.2021).
- [9] Janglin Chen, Wayne Cranton, Mark Fihn. Handbook of Visual Display Technology. Springer International Publishing, 2016. Vol. 1. P. 528–529.
- [10] Wilhelm Burger, Mark James Burge. Principles of Digital Image Processing: Fundamental Techniques. Springer International Publishing, 2009. Vol. 1. P. 55–59.

- [11] Janglin Chen, Wayne Cranton, Mark Fihn. Handbook of Visual Display Technology. Springer International Publishing, 2016. Vol. 1. P. 461–462.