ПРОГРАММИРОВАНИЕ, 2004, № 6, с. 1–15

УДК 681.3.06

ВЫСОКОКАЧЕСТВЕННЫЙ АЛГОРИТМ ИНТЕРПОЛЯЦИИ ИЗОБРАЖЕНИЙ В ВИДЕ БАЙЕРОВСКИХ ШАБЛОНОВ

© **2004 г. А. Лукин, Д. Кубасов**

Факультет вычислительной математики и кибернетики

МГУ им. М.В. Ломоносова

119992 Москва, Воробьевы горы

E-mail: lukin@graphics.cs.msu.su, lukin@ixbt.com

Поступила в редакцию

В этой статье мы даем краткий обзор алгоритмов интерполяции изображений, представленных в виде байеровских шаблонов, и предлагаем новый алгоритм интерполяции, значительно повышающий качество изображения. Предложенный алгоритм базируется на методе Киммела и использует ряд улучшений: более качественную интерполяцию зеленого цвета, адаптивное управление числом итераций, проекцию на исходные данные.

1. ВВЕДЕНИЕ

Цифровые камеры и сканеры давно уже перестали быть экзотикой и атрибутом дорогих фотолабораторий, прочно обосновавшись на рынке цифровых устройств для домашнего использования. Стремление занять как можно более широкий сектор рынка толкает производителей на упрощение и, следовательно, удешевление процесса производства с сохранением потребительских качеств и функциональности устройства.

Одним из таких упрощений является использование при формировании изображения одной светочувствительной матрицы вместо трех, соответствующих базисным цветам. В такой одной матрице присутствует несколько видов фотоэлементов, каждый из которых чувствителен к отдельному базисному цвету. Эти элементы в матрице расположены в виде мозаики, называемой обычно байеровским шаблоном (bayer pattern).

Таким образом, в каждой точке матрицы имеется информация только об одной из трех цветовых компонент, в то время как выдаваемое цифровое изображение должно содержать все три компоненты (R, G и B) для каждого пикселя.

Задача интерполяции байеровских шаблонов (demosaicing, demosaicking) – получение "полноцветного" изображения по его байеровскому шаблону. Алгоритм должен проинтерполировать каждую из цветовых плоскостей в тех точках, где значение соответствующей цветовой компоненты неизвестно.

2. ОБЗОР СУЩЕСТВУЮЩИХ МЕТОДОВ

Большинство существующих методов решения данной задачи можно условно разделить на линейные подходы, адаптивные алгоритмы и математические методы восстановления.

2.1. Линейные методы

2.1.1. Независимая интерполяция цветовых плоскостей

Самый простой способ восстановления – применить линейную интерполяцию (например, билинейную или бикубическую) к каждой цветовой плоскости в отдельности. Хотя этот метод достаточно быстр и прост в реализации, применение его приводит к появлению заметных артефактов.

Один из артефактов – цветовой муар – в той или иной степени свойственен почти всем методам и является следствием несовпадения позиций сенсоров разных цветов. Поэтому многие

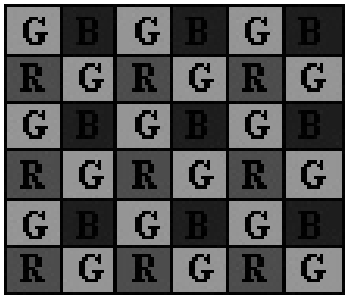


Рис. 1. Байеровский шаблон.

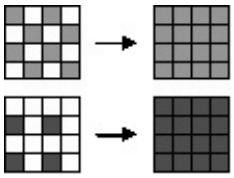


Рис. 2. Интерполяция цветовых плоскостей.

1) Зеленый:

$$G5 = (G2 + G4 + G6 + G8)/4$$

2) Красный:

$$R2 = (R1 + R3)/2$$
$$R4 = (R1 + R7)/2$$
$$R5 = (R1 + R3 + R7 + R9)/4$$

3) Синий: аналогично красному

Рис. 3. Линейная интерполяция пиксела в позиции 5.

методы используют избыточность зеленых пикселей в мозаике для более качественного восстановления высоких частот изображения, а затем интерполируют красный и синий цвета с использованием восстановленного зеленого.

2.1.2. *Интерполяция цветовых отношений*

Восстановление красного и синего цветов на основе восстановленного зеленого базируется на некотором предположении о характере зависимости между цветовыми плоскостями.

Одним из возможных предположений является предположение о постоянстве соотношений цветов (например, отношения красного к зеленому и отношения синего к зеленому) в пределах одного объекта на изображении [2]. Тогда, имея полностью восстановленную зеленую компоненту, можно интерполировать не сами значения красного в соседних точках, а отношения красного к зеленому в этих точках. Это даст лучший результат, чем независимая интерполяция цветовых плоскостей, так как зеленая компонента имеет более высокую частоту дискретизации и может быть восстановлена (даже линейным методом) более аккуратно. Для интерполяции отношений, в свою очередь, может использоваться линейный либо какой-то другой метод.

Слабое место этого метода – области, где мало зеленого цвета. В этом случае отношения других цветов к зеленому резко возрастают и становятся менее адекватными вследствие влияния шума. Эту проблему можно решить, если интерполировать не сами отношения, а их логарифмы, то есть разности цветов.

2.2. *Адаптивные методы*

Адаптивные методы заключаются в выборе конкретного вида линейного фильтра для каждого интерполируемого пикселя в отдельности, в зависимости от эвристических закономерностей, вычисляемых по окрестности данного пикселя.

2.2.1. *Edge-adaptive метод*

При применении интерполяции цветовых отношений (или их разностей) большое влияние на конечный результат оказывает качество первоначальной интерполяции зеленого. Поэтому желательно производить этот шаг как можно более аккуратно. Например, вместо линейной интерполяции использовать интерполяцию в направлении границ.

Самый простой вариант интерполяции в направлении границ работает следующим образом. Для пикселя, который нужно проинтерполировать, по его имеющимся соседям оцениваются вертикальный и горизонтальный градиенты [1] и за направление границы в окрестности данного пикселя принимается то направление, в котором градиент меньше. После этого искомое значение определяется как полусумма двух ближайших соседей в направлении границы.

Этот метод может быть улучшен путем рассмотрения более широкой области и учета градиентов по другим цветам для выбора направления интерполяции [2].

После того, как зеленый цвет восстановлен этим методом, красная и синяя компоненты интерполируются с использованием метода интерполяции цветовых отношений.

2.2.2. Алгоритм Киммела

Восстановление изображения алгоритмом Киммела [3] происходит в три этапа:

1. интерполяция зеленого;
2. интерполяция красного и синего с использованием зеленого;
3. коррекция.

Более подробно опишем каждый этап.

2.2.2.1. Интерполяция зеленого в алгоритме Киммела

Неизвестное значение зеленой компоненты в алгоритме Киммела определяется как линейная комбинация значений четырех ближайших соседей, значения которых известны. Веса E_i в этой линейной комбинации характеризуют вероятность того, что G_i принадлежит тому же объекту, что и G_5.

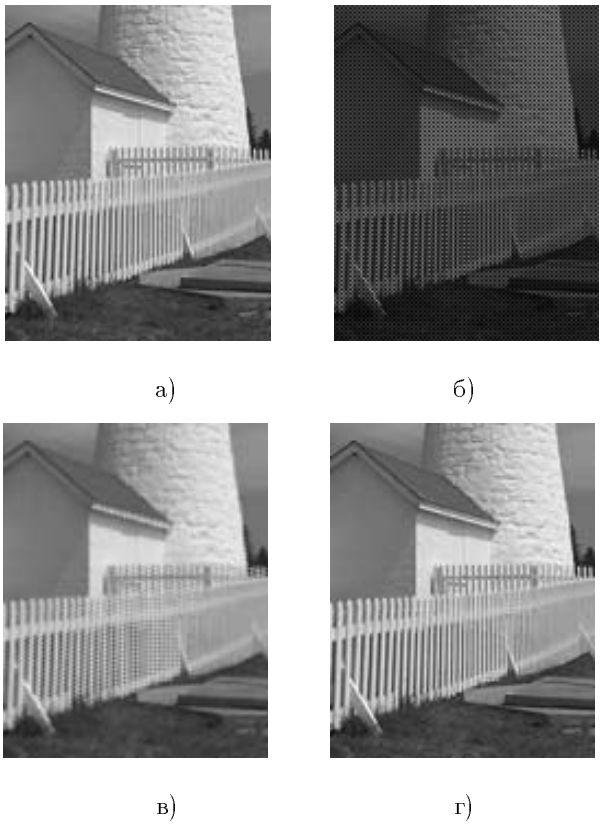
Весовые функции E_i вычисляются следующим образом.

Сначала вводится понятие "производной" по четырем направлениям (вертикальное, горизонтальное и два диагональных) в заданной точке.

Пусть требуется вычислить производные в точке $P5$ (см. рис. 8). P используется, чтобы показать, что производные вычисляются одинаково, независимо от того, какая компонента задана в шаблоне в точке $P5$. Тогда производные вычисляются по формулам:

$$D_x(P_5) = \frac{P_4 - P_6}{2}, \quad D_y(P_5) = \frac{P_2 - P_8}{2},$$
$$D_{xd}(P_5) = \frac{P_3 - P_7}{2\sqrt{2}}, \quad D_{yd}(P_5) = \frac{P_1 - P_9}{2\sqrt{2}},$$

где P_i – значение интенсивности, заданное в шаблоне в этой точке. Обратите внимание, что, какая бы ни была задана компонента в точке $P5$ (зеленая, красная или синяя), разности всегда вычисляются между интенсивностями одного цвета.



а) б) в) г)

Рис. 4. Исходное изображение (а), байеровский шаблон (б), билинейная интерполяция (в) и предложенный метод (г).

Если в $P5$ задано значение зеленого цвета, то мы можем определить производные чуть точнее, положив:

$$D_{xd}(P_5) = \max\left\{\left|\frac{P_3 - P_5}{\sqrt{2}}\right|, \left|\frac{P_7 - P_5}{\sqrt{2}}\right|\right\},$$
$$D_{yd}(P_5) = \max\left\{\left|\frac{P_1 - P_5}{\sqrt{2}}\right|, \left|\frac{P_9 - P_5}{\sqrt{2}}\right|\right\}.$$

Тогда весовая функция определяется так:

$$E_i = \frac{1}{\sqrt{1 + D^2(P_5) + D^2(P_i)}},$$

где $D(P_i)$ – производная в направлении P_i.

2.2.2.2. Интерполяция красного и синего и использованием зеленого

Для восстановления красного и синего цветов используется описанный выше метод интерполяции цветовых отношений. Сами отношения

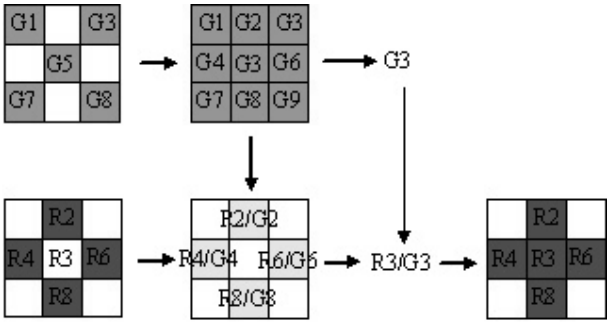


Рис. 5. Интерполяция отношений цветов.

1) Горизонтальный градиент:

$$H = |G2 - G4|$$

2) Вертикальный градиент:

$$V = |G1 - G5|$$

3)
```
If  H>V
      G3=(G1+G5)/2
  Else If V>H
      G3=(G2+G4)/2
  Else
      G3=(G1+G2+G4+G5)/4
```

Рис. 6. Интерполяция вдоль границ. $G1$, $G2$, $G4$, $G5$ – имеющиеся значения зеленого, $G3$ – интерполируемое.

интерполируются таким же образом, как интерполировались зеленые пиксели на первом этапе, с использованием весовой функции E, определенной выше.

Формула для интерполяции синего цвета аналогична.

2.2.2.3. Коррекция в алгоритме Киммела

Это, пожалуй, ключевой этап в алгоритме, так как именно благодаря нему подавляется большая часть артефактов, например, цветовой муар. Основная идея заключается в следующем.

При интерполяции красного (синего) цвета мы предположили, что отношение красного (синего) к зеленому должно быть постоянным в пределах одного объекта. Это значит, что и отношение зеленого к красному (синему) также должно быть постоянным в этой области. Поэтому после интерполяции красного и синего можно скорректировать зеленые пиксели так, чтобы удовлетворить этому предположению. Но это изменит исходные значения отношений красного (синего) к зеленому, так что теперь требуется подкорректировать значения красного (синего). Авторы метода [3] предлагают повторять этот процесс три раза, попеременно корректируя значения зеленого и красного/синего.

Итак, формально этап коррекции выглядит следующим образом:

- Повторять три раза:
- Скорректировать значения зеленого в соответствии со значениями отношений зеленого к синему и красному:

$$G_5^B = B_5 \frac{E_2\frac{G_2}{B_2} + E_4\frac{G_4}{B_4} + E_6\frac{G_6}{B_6} + E_8\frac{G_8}{B_8}}{E_2 + E_4 + E_6 + E_8},$$

$$G_5^R = R_5 \frac{E_2\frac{G_2}{R_2} + E_4\frac{G_4}{R_4} + E_6\frac{G_6}{R_6} + E_8\frac{G_8}{R_8}}{E_2 + E_4 + E_6 + E_8},$$

$$G_5 = \frac{G_5^R + G_5^B}{2}.$$

- Скорректировать значения красного и синего в соответствии со значениями их отношений к зеленому:

$$B_5 = G_5 \frac{\sum E_i \frac{B_i}{G_i}}{\sum E_i}, \quad i \neq 5,$$

$$R_5 = G_5 \frac{\sum E_i \frac{R_i}{G_i}}{\sum E_i}, \quad i \neq 5.$$

- Конец цикла.

2.3. Математические методы

2.3.1. Optimal Recovery

Для качественного восстановления зеленого цвета, от которого во многом зависит качество всей интерполяции, можно применять известные и хорошо себя зарекомендовавшие методы, например, NEDI [7] или метод оптимального восстановления (optimal recovery), являющийся его расширением [4].

Так, авторы работы [5] предлагают заменить в вышеописанном алгоритме Киммела [3] интерполяцию зеленого на интерполяцию методом оптимального восстановления, а интерполяцию цветовых отношений – на интерполяцию цветовых разностей.

Идея метода оптимального восстановления заключается в расширении метода NEDI, описанного ниже, с помощью применения дополнительных функционалов, накладывающих ограничения на получающиеся значения пикселей. В результате удается получить более качественные результаты, чем за счет подавления при применении метода NEDI многих артефактов, свойственных этому методу, с помощью ограничивающих функционалов. Опишем кратко теорию оптимального восстановления.

Пусть есть неизвестный вектор x, линейный функционал F от которого требуется оценить, и пусть известно его значение на наборе L линейных функционалов F_i $(i = 1, \ldots, L)$. Далее, будем предполагать, что искомый вектор x принадлежит некоторому эллипсоидному классу K:

$$K = \{x \in R^n : x^T Q x \le \varepsilon\},$$

где Q – задано.

Заметим, что функционалы F_i задают гиперплоскость X в n-мерном пространстве, а ее пересечение с эллипсоидом K есть гиперкруг. Обозначим его C_x. Задача оптимального восстановления состоит в том, чтобы выбрать в X такой вектор $\hat{x}$ среди всех $x \in C_x$, который минимизирует ошибку аппроксимации искомого функционала:

$$\delta = \max_{x^T Q x} |\hat{y} - y|,$$

где $\hat{y} = (\hat{x})$, $y = F(x)$. Эта минимизация достигается в центре Чебышева – векторе $\bar{u} \in X$ с минимальной Q-нормой. В [4] показано, что искомый вектор $\bar{u}$ может быть представлен в виде линейной комбинации представителей заданных линейных функционалов F_i:

$$\bar{u} = \sum_{i=1}^{L} \alpha_i \phi_i.$$

Применительно к задаче интерполирования шаблона в качестве вектора x берется какое-либо подмножество этого шаблона, содержащее интерполируемый пиксель в позиции j, тогда $F(x) = x_j$, а в качестве известных функционалов F_i $(i = 1, \ldots, L)$ обычно берутся известные значения соседних пикселей: $F_i(x) = x_i$, где x_i – заданы в шаблоне. Тогда $F(\bar{u})$ – искомое значение интерполируемого пикселя.

$$G_5 = \frac{E_2 G_2 + E_4 G_4 + E_6 G_6 + E_8 G_8}{E_2 + E_4 + E_6 + E_8}$$

E_i – весовая функция

Рис. 7. Интерполяция зеленого в алгоритме Киммела.

Рис. 8.

$$R_5 = \frac{E_1 \frac{R_1}{G_1} + E_3 \frac{R_3}{G_3} + E_7 \frac{R_7}{G_7} + E_9 \frac{R_9}{G_9}}{E_1 + E_3 + E_7 + E_9}$$

E_i – весовая функция

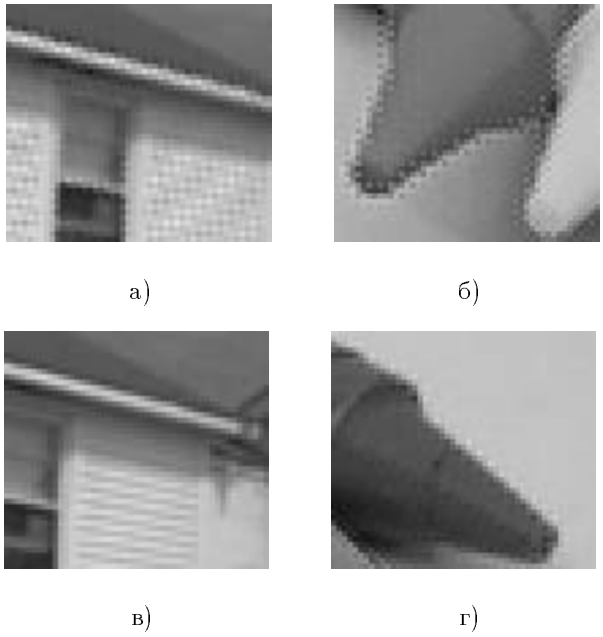
Рис. 9. Интерполяция красного цвета в алгоритме Киммела.

Очевидно, что основная проблема теперь – построить класс K, то есть выбрать такое Q, чтобы он наиболее адекватно представлял изображение в окрестности интерполируемого пикселя. В [4] показано, что $Q = (SS^T)^{-1}$, где S – тренировочный набор, состоящий из m обучающих векторов $x \in R^n$, записанных в качестве столбцов матрицы S.

В [5] восстановление зеленого происходит в два этапа:

- Грубое восстановление. Класс K грубо моделирует изображение в окрестности интерполируемого пикселя, так как обучающие вектора берутся только в заданных точках шаблона, то есть через 1 пиксель.

- Точное восстановление. Обучающие вектора берутся из интерполированного на первом этапе изображения, поэтому имеют тот же масштаб, что и оцениваемый вектор x.



а) б) в) г)

Рис. 10. Цветовой муар (а), эффект молнии (б), потеря четкости (в), зазубренность границ (г).

На обоих этапах тренировочный набор S составляется из всех возможных обучающих векторов, попадающих в окно размером 15×15 с центром в позиции интерполируемого пикселя.

Второй этап во многих случаях может быть опущен. Он дает заметное повышение качества только на высокочастотных областях, т.е. повторяющихся областях мелкой текстуры.

2.3.2. *Alternating Projections*

Метод чередующихся проекций [6] относится к классу так называемых POCS-методов (проектирование на выпуклые множества). Основная идея метода состоит в получении некоего начального приближения интерполированного изображения и последующем итеративном его улучшении путем попеременного удовлетворении двум множествам ограничений.

Первое множество ограничений – это заданные в шаблоне значения цветов: результат интерполяции должен совпадать с этими значениями в тех точках, где они заданы.

Второе множество ограничений выводится из предположения, что высокие частоты (детали) всех трех цветовых компонент в каждой точке близки, и заключается в том, что вейвлет-коэффициенты красного и синего цветов должны отличаться от вейвлет-коэффициентов зеленого в той же точке не больше, чем на заданный порог. Порог может задаваться как для всего изображения в целом, так и для каждой точки в отдельности в виде функции от каких-то локальных характеристик окрестности этой точки и характеризует близость высоких частот различных цветовых плоскостей. Если они очень близки, то порог должен быть низким; если ожидается (допускается) большое различие между ними – то высоким. Для изображений в оттенках серого порог можно задать равным нулю.

Таким образом, схема алгоритма выглядит следующим образом:

- Первоначальная интерполяция зеленого одним из методов, например, Edge-Adaptive.
- Повторять несколько раз:
  - Произвести вейвлет-преобразование каждой из цветовых плоскостей.
  - Изменить вейвлет-коэффициенты красного и синего на вейвлет-коэффициенты зеленого таким образом, чтобы они отличались от них не более чем на заданный порог.
  - Произвести обратное (восстанавливающее) вейвлет-преобразование.
  - Вернуть исходные значения цветов в тех точках, где они изначально были заданы в шаблоне.
  - Конец цикла.

2.4. *Оценка качества алгоритмов*

Широко распространенной мерой качества алгоритмов интерполяции байеровских шаблонов является оценка PSNR. Для ее вычисления используется полноцветное исходное изображение, из него искусственно создается байеровский шаблон, который подается на вход алгоритму интерполяции. Полученное интерполированное изображение сравнивается с исходным полноцветным изображением с помощью PSNR. Существуют и другие меры разницы между двумя изображениями: вычисление цветовых разностей в равномерных к восприятию цветовых

моделях (например, ΔE_{00} [9]), использование моделей человеческого восприятия и маскировки искажений. Для оценки результатов в нашей работе мы будем использовать оценки PSNR и ΔE_{00} наравне с визуальной оценкой артефактов интерполяции.

Основные артефакты интерполированных изображений, важные при визуальной оценке, приведены на рис. 10.

3. ПРЕДЛОЖЕННЫЙ АЛГОРИТМ

Наш алгоритм состоит из следующих стадий:

1. Качественная интерполяция зеленого цвета с использованием методов градиентной интерполяции и NEDI.
2. Применение модифицированного алгоритма Киммела с переменным числом итераций и другими незначительными изменениями.
3. Проекция на исходные данные.
4. Повторение всего алгоритма с использованием интерполированного изображения.

Рассмотрим каждый из этих этапов подробнее.

3.1. Интерполяция зеленого цвета

В нашем методе интерполяции зеленого цвета мы определяем направление и веса интерполяции только на основании зеленой компоненты, без использования красной и синей компонент. Этот выбор объясняется тем, что красная и синяя компоненты в байеровском шаблоне имеют худшее разрешение, и возникающий в них алиасинг может приводить к неверному выбору направления интерполяции.

3.1.1. NEDI

Метод интерполяции NEDI (New Edge-Directed Interpolation) был предложен в работе [7] и развит в работах [8] и [5]. Для NEDI характерна качественная интерполяция границ в изображении, не вызывающая эффекта рваных краев. Однако NEDI имеет ряд недостатков: высокая вычислительная сложность, артефакты акварелизации в областях мелкой текстуры, нестабильность алгоритма на гладких областях изображения [10]. Поэтому мы предлагаем комбинировать метод NEDI с более простыми методами интерполяции, адаптирующимися к краям изображения.

1) Вычисление горизонтального и вертикального градиентов H и V

2)
```
If  H≥ V
      G3=(G2+G8)/2
Else
      G3=(G4+G6)/2
```

Рис. 11. Градиентная интерполяция зеленого (упрощенный вариант).

1) $E_2 = E_8 = 1/(\varepsilon + V^8)$
2) $E_4 = E_6 = 1/(\varepsilon + H^8)$
3) $G3$ вычисляется, согласно рис. 9.

Рис. 12. Градиентная интерполяция зеленого (упрощенный вариант).

В нашем методе мы используем модификацию NEDI, предложенную в работе [8] (размытие карты интенсивности, пространственное расширение апертуры). Также для повышения стабильности операции обращения матрицы при вычислении оптимальных весов к карте интенсивности добавлялся случайный белый шум небольшой амплитуды.

3.1.2. Градиентная интерполяция

В качестве второго, более простого, метода интерполяции мы используем градиентную интерполяцию. В упрощенном варианте градиентной интерполяции мы выбираем между вертикальным и горизонтальным направлением интерполяции. Выбор осуществляется путем сравнения локальных вертикального и горизонтального градиентов зеленой компоненты изображения. Выбирается направление интерполяции, соответствующее меньшему градиенту.

Каждый из градиентов усредняется по окрестности текущего пикселя. На первом шаге градиент H усредняется по небольшой окрест-

ности (формула 1):

$$D[i,j] = |G[i,j] - G[i,j+2]|,$$
$$H = D[i,j-1] + D[i-1,j-2] +$$
$$+D[i+1,j-2] + D[i-1,j] + D[i+1,j] +$$
$$+D[i,j-3] + D[i,j+1] + D[i-2,j-1] +$$
$$+D[i+2,j-1] + (D[i,j-5] + D[i,j+3] +$$
$$+D[i-2,j-3] + D[i+2,j-3] +$$
$$+D[i-2,j+1] + D[i+2,j+1] +$$
$$+D[i-1,j-4] + D[i+1,j-4] +$$
$$+D[i-1,j+2] + D[i+1,j+2])/2.$$

Здесь $D[]$ – модуль одной горизонтальной разности.

Далее сравниваются вертикальный и горизонтальный градиенты, и при близости их значений принимается решение о недостаточной определенности направления интерполяции, и в этом случае градиенты вычисляются в более широкой окрестности:

`If |H - V| <` $\beta \cdot$ `(H + V) then Use bigger window`

Всего мы используем три различных размера окрестности: наименьший, согласно формуле 1, составляет примерно 4×4 пикселя; средняя окрестность имеет размер 10×10 пикселей; большая окрестность имеет размер примерно 22×22 пикселя. После выбора окончательного направления интерполяция производится путем усреднения двух соседних пикселей в данном направлении (согласно рис. 11).

В более сложном методе интерполяции выбирается не одно из двух направлений, а веса интерполяции вычисляются по формуле, приведенной на рис. 12.

Высокая степень градиентов в знаменателе практически сводит на нет смешивание вертикального и горизонтального направлений интерполяции. Это сделано с целью предотвращения цветового муара, который часто появляется в результате такого смешивания.

Следующая модификация улучшает качество интерполяции на гладких областях. Она заключается в уменьшении степени градиентов в знаменателе для гладких областей изображения. В таких областях существенную роль в формировании градиентов играет случайный шум изображения, и результирующее интерполированное изображение может иметь артефакт "акварельности", в то время как обычная билинейная интерполяция такого артефакта не дает. Поэтому уменьшение степени градиентов в знаменателе усиливает смешивание вертикальной и горизонтальной интерполяции и приближает метод к билинейной интерполяции. В нашем алгоритме, в зависимости от гладкости области, степень градиентов в знаменателе уменьшается с 8 до 4 или 2.

В качестве результата градиентной интерполяции мы получаем интерполированную зеленую цветовую компоненту, а также дополнительный массив, хранящий координаты пикселей, для которых использовалось вычисление градиента с максимальным размером окна. Этот массив будет использоваться в дальнейшем для определения сложных, "проблемных" областей изображения, где направление интерполяции определялось неуверенно.

3.1.3. Комбинирование градиентной интерполяции и NEDI

В нашем методе интерполяции зеленой компоненты мы комбинируем результаты NEDI и градиентной интерполяции для устранения артефактов обоих подходов и ускорения NEDI (так как NEDI теперь будет применяться лишь к небольшой части пикселей изображения). Для этого после градиентной интерполяции зеленого цвета мы применяем к результирующему изображению специальный алгоритм классификации, определяющий пропорции смешивания результатов NEDI и градиентной интерполяции для каждого пикселя. Цель алгоритма классификации – разделить пиксели изображения на 2 класса. К первому классу должны относиться края (границы) в изображении. Ко второму классу должны относиться гладкие области изображения, области высоких частот (т.е. повторяющихся мелких шаблонов) и области мелкой текстуры (трава, листва и др.) К первому классу будет применяться алгоритм NEDI, так как он дает хорошие результаты на выраженных границах. Ко второму классу будет применяться градиентная интерполяция, так как NEDI здесь даст артефакты.

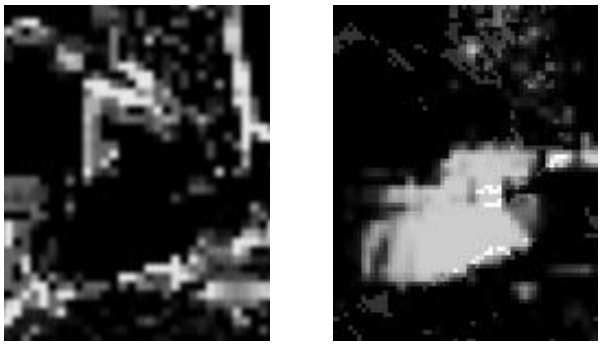
Используемый классификатор строит решение, руководствуясь комбинацией следующих критериев:

1. *Наличие деталей в окрестности данного пиксела.* Критерий вычисляется как линейный фильтр (3×3), аппроксимирующий вторую производную $((0,1,0),\ (1,-4,1),\ (0,1,0))$. Этот критерий позволяет отнести гладкие области в изображении ко второму классу.

2. *Отношение низкочастотной энергии в окрестности данного пиксела к высокочастотной энергии.* Критерий вычисляется с помощью локального оконного двумерного преобразования Фурье (размер окна – 8×8 пикселей). Подсчитывается суммарная энергия коэффициентов для области высоких и низких частот. Низкими частотами считаются частоты от 0 до 0.25 периодов на пиксель (нулевая частота исключается из рассмотрения, так как соответствует общей яркости и не должна влиять на определение границ). Этот критерий позволяет отнести ко второму классу области с преобладающей высокочастотной информацией (мелкие повторяющиеся шаблоны), а к первому классу – области границ (так как в границах преобладает низкочастотная информация).

3. *Сложность двумерной структуры изображения в окрестности данного пиксела.* Критерий вычисляется следующим образом. Сначала область 8×8 вокруг данного пиксела преобразуется в двухцветную палитру (методом K-средних с очень малым числом итераций). Затем в полученном бинарном изображении ищется количество связных областей (рассматривается 8-связность, т.е. у каждого пиксела 8 связных с ним соседей). Этот критерий позволяет отнести к первому классу области с простой структурой (у краев обычно только 2 связные области: по разные стороны от края), а ко второму классу – области с мелкой текстурой (листва, трава и тому подобное).

В результате объединения этих критериев каждому пикселю приписывается "вероятность" отнесения его к первому или второму классу. Это число и определяет пропорции смешивания результатов NEDI и градиентной интерполяции (рис. 13 а).



а) б)

Рис. 13. Классификация пикселей для переключения вида интерполяции (а), "карта проблемности" (б). Исходное изображение см. на рис. 4.

Так как классификацию мы проводим до интерполяции NEDI, то мы можем выполнять NEDI только для точек, где соответствующий вес не равен нулю (для дальнейшего ускорения интерполяции можно поднять порог от нуля до 0.3 без заметного снижения качества).

3.2. Модификация алгоритма Киммела

После интерполяции зеленого цвета мы производим интерполяцию красного и синего цветов с помощью алгоритма Киммела с несколькими изменениями. Основным из них является использование переменного числа итераций.

3.2.1. Переменное число итераций

Стандартная реализация алгоритма Киммела предполагает использование трех итераций для восстановления красного и синего цветов. Мы провели эксперименты для выяснения зависимости результирующего изображения от числа итераций. Эксперименты показали, что с ростом числа итераций улучшается подавление цветового муара за счет более полного взаимного согласования трех цветовых компонент. Однако одновременно ухудшается цветовая насыщенность изображения в местах цветовых переходов [11]. Три итерации обычно приемлемы

для большинства изображений и в среднем дают наилучшее значение PSNR. Однако результаты можно существенно улучшить, управляя числом итераций адаптивно.

Основная задача алгоритма Киммела – устранить цветовой муар. Цветовой муар возникает из-за алиасинга красной и синей цветовых компонент, т.е. в тех местах, где пониженной частоты дискретизации недостаточно для передачи высоких пространственных частот. Идея нашего метода – находить такие области и увеличивать в них число итераций метода Киммела. Поиск областей осуществляется с помощью двух критериев.

Первый из них – это неопределенность нахождения градиента, запоминаемая во вспомогательном массиве (см. раздел 3.1.2).

Второй критерий – это классификатор областей, аналогичный описанному в разделе 3.1.3. Отличие от раздела 3.1.3 заключается в обратном использовании второго критерия (так как мы относим области с преобладающей высокочастотной информацией ко второму классу) и неиспользовании третьего критерия (в практической реализации алгоритма эти классификаторы совмещены, так как работают над одними и теми же данными).

В результате работы классификатора мы получаем карту распределения вероятностей отнесения пикселей ко второму классу. Далее мы называем эту карту "картой проблемности", так как она показывает области, в которых вероятно возникновение цветового муара (рис. 13 б).

На основании "карты проблемности" регулируется число итераций алгоритма Киммела: в областях "отсутствия проблем" число итераций может быть очень малым: 1–2. А в "проблемных" областях число итераций может возрастать до 10–12.

3.2.2. *Расширение окна вычисления градиентов*

При вычислении весов интерполяции в алгоритме Киммела считаются градиенты по нескольким направлениям. Каждый из градиентов находится как разность двух значений пикселей: $Grad = |G[i,j] - G[k,m]|$.

В нашем алгоритме мы модифицировали вычисление градиента, расширив вычисление разностей на большее пространственное окно:

$$Grad = 8 \cdot |G[i,j] - G[k,m]| + |G[i+1,j] - G[k+1,m]| + |G[i-1,j] - G[k-1,m]| + |G[i,j+1] - G[k,m+1]| + |G[i,j-1] - G[k,m-1]|.$$

Это привело к небольшому увеличению PSNR.

3.2.3. *Полурекурсивное обновление цветов*

В оригинальном алгоритме Киммела вычисление новых значений скорректированных цветов происходит над одним массивом, а записываются скорректированные цвета в другой массив. Затем, на следующей итерации, массивы меняются местами. В нашем же алгоритме мы записываем скорректированные значения пикселей в исходный массив и используем их при вычислении взвешенных значений следующих интерполированных пикселей. На нечетных итерациях обход массива пикселей производится слева направо, сверху вниз, а на четных – в обратном порядке. Это небольшая модификация также увеличила результирующее значение PSNR.

3.3. *Проекция на исходные данные*

В результате работы алгоритма Киммела происходит согласование трех цветовых компонент и уменьшается цветовой муар. Однако особенность алгоритма заключается в том, что он изменяет также значения цветовых компонент в тех позициях, где они были изначально заданы. Очевидно, что подстановка исходных значений цветовых компонент в эти позиции полученного изображения улучшит PSNR. Оказывается, визуальное качество изображения при этом также улучшается. Если рассматривать проблему интерполяции как задачу проекции на выпуклые множества ограничений (POCS) [6], то подстановка исходных значений цветовых компонент является проекцией изображения на исходные данные.

В нашем алгоритме мы используем более сложную модель проектирования изображения на исходные данные. Идея метода в том, чтобы с изменением одной подставляемой цветовой компоненты менять и другие цветовые компоненты данного пикселя в соответствии с локальной моделью изменения цветов в изображении.

Средние значения PSNR по тестовым изображениям

Метод	Среднее значение PSNR-RGB	Среднее значение PSNR-CIEDE2000
Bilinear	27.50	35.56
Kimmel	33.50	42.57
AQua-2	34.63	42.99
Alternating Projections	35.24	42.76
Предложенный алгоритм	37.10	44.36

Рис. 14. Тестовый набор изображений.

Локальная модель строится следующим образом. Мы пытаемся аппроксимировать все цвета пикселей из окрестности данного пикселя прямой линией в трехмерном цветовом пространстве. Если рассмотреть цвета этих пикселей в виде облака точек (в трехмерном цветовом пространстве), то требуется провести в пространстве прямую линию так, чтобы она оптимально аппроксимировала это облако. В нашем методе мы решаем эту задачу упрощенно, беря отрезок,

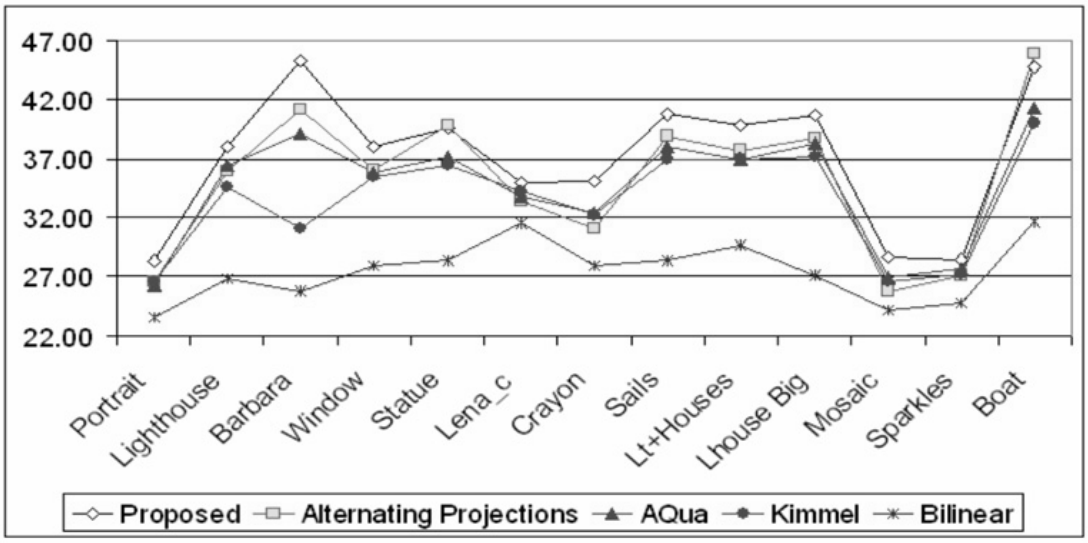


Рис. 15. График PSNR-RGB по изображениям.

соединяющий две наиболее далекие друг от друга точки облака. Данный отрезок мы называем "вектором цветового направления". Он показывает доминантное направление изменения цвета в трехмерном цветовом пространстве в окрестности данного пикселя изображения. Например, если в окрестности данного пикселя находится граница между черным и белым цветами (включающая и промежуточные цвета), то вектор цветового направления будет иметь равные компоненты R, G и B.

В нашем алгоритме мы используем окрестность 3×3 и после нахождения вектора цветового направления определяем также степень отклонения пикселей окрестности от этого вектора. Эта степень отклонения может рассматриваться как показатель консистентности такой локальной одномерной цветовой модели.

Когда вектор цветового направления получен, подстановка исходных цветовых компонент производится по следующему принципу: исходная подставляемая компонента задает величину сдвига цвета пикселя, а направление сдвига задается вектором цветового направления.

Этот простой принцип требует некоторого уточнения, так как он может, например, приводить к очень большим изменениям компонент, если вектор цветового направления перпендикулярен подставляемой цветовой компоненте. Поэтому в нашей реализации мы предусмотрели некоторые ограничения на максимальное изменение цветовых компонент следующего вида (на примере подстановки зеленой цветовой компоненты):

```
Green = OriginalGreen;
if (CDV[Green]>eps)
{
  Amount = 1 - eps/CDV[Green] + GrayscaleMeasure;
  Red += (OriginalGreen - Green) * CDV[Red] /
                    CDV[Green] * Amount;
  Bound(Red, -Limit, +Limit);
  Blue += (OriginalGreen - Green) * CDV[Blue] /
              CDV[Green] * Amount;
  Bound(Blue, -Limit, +Limit);
}
```

Здесь Red, $Green$, $Blue$ – значения цветовых компонент, $OriginalGreen$ – исходное подставляемое значение зеленой компоненты, $\{CDV[Red], CDV[Green], CDV[Blue]\}$ – вектор цветового направления, $Limit$ – ограничение изменения цветовых компонент (может вычисляться исходя из консистентности локальной цветовой модели).

Переменная $Amount$ уменьшает действие подставляемой цветовой компоненты на остальные цветовые компоненты данного пикселя. Ее значение вычисляется исходя из двух критериев:

1. Уменьшение силы воздействия для защиты от нестабильности при перпендикулярности

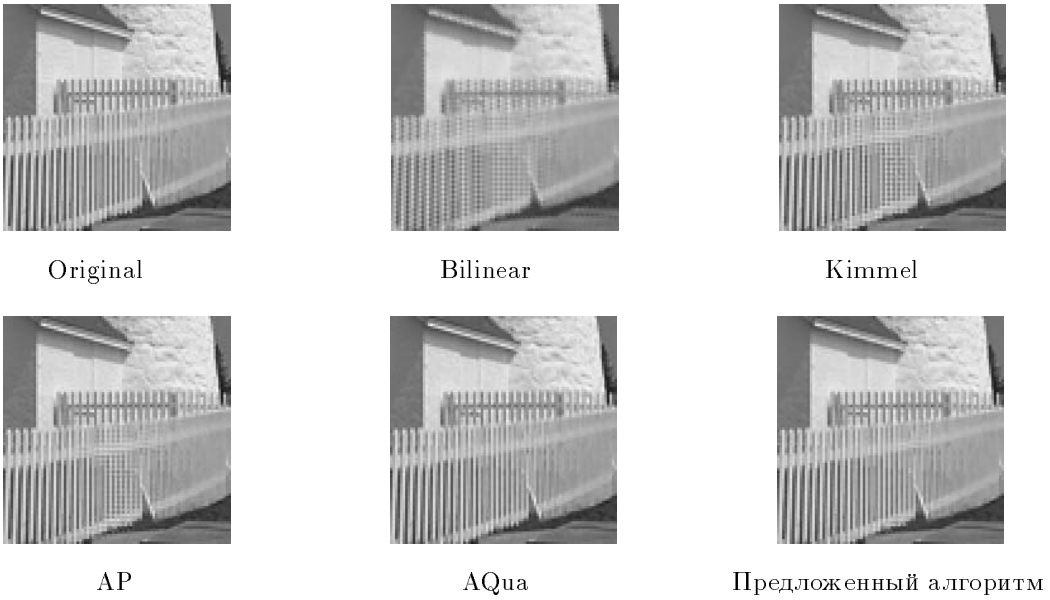


Рис. 16. Результаты работы различных методов на изображении Lighthouse.

вектора CDV зеленой компоненте (член с Eps).

2. Увеличение силы воздействия при близости направления вектора CDV к направлению $\{1, 1, 1\}$. Этот критерий позволяет усилить взаимосвязь цветовых компонент для черно-белых изображений и повысить качество восстановления. Одновременно он ослабляет взаимосвязь цветовых компонент для областей с переходами между цветами различных тонов, что также поднимает качество и уменьшает артефакт "эффект молнии".

3.4. Повторение всего алгоритма

Для дополнительного повышения качества результирующего изображения можно использовать еще одну итерацию всего алгоритма. На этой итерации в интерполяции зеленой компоненты используется изображение с полным разрешением, полученное в результате первой итерации. Таким образом, достигается значительное повышение качества интерполяции зеленого цвета, особенно – в областях изображения с преобладанием высокочастотной информации. На первой итерации направление интерполяции могло быть определено неверно из-за алиасинга, который возникал из-за недостаточной частоты дискретизации зеленого цвета. После проекции на исходные данные на первой итерации нужные высокие частоты отчасти восстанавливаются, и это помогает повторной интерполяции зеленой компоненты на второй итерации алгоритма.

4. РЕЗУЛЬТАТЫ

Нами было проведено сравнение предложенного метода с наиболее качественными из описанных алгоритмов. Для сравнения был подготовлен ряд изображений, состоящий, в основном, из известных и широко используемых в публикациях примеров.

Качество результатов измерялось с помощью меры PSNR, посчитанной в цветовом пространстве RGB и с помощью цветовых разностей ΔE_{00} в цветовом пространстве CIEDE2000 [9] между исходным изображением, из которого затем генерировался байеровский шаблон, и результатом применения алгоритма к этому шаблону.

Для тестирования мы выбрали набор различных изображений, часто используемых при оценке качества алгоритмов (рис. 14). Общие (усредненные) результаты методов интерполяции, примененных к этим изображениям, приведены в таблице.

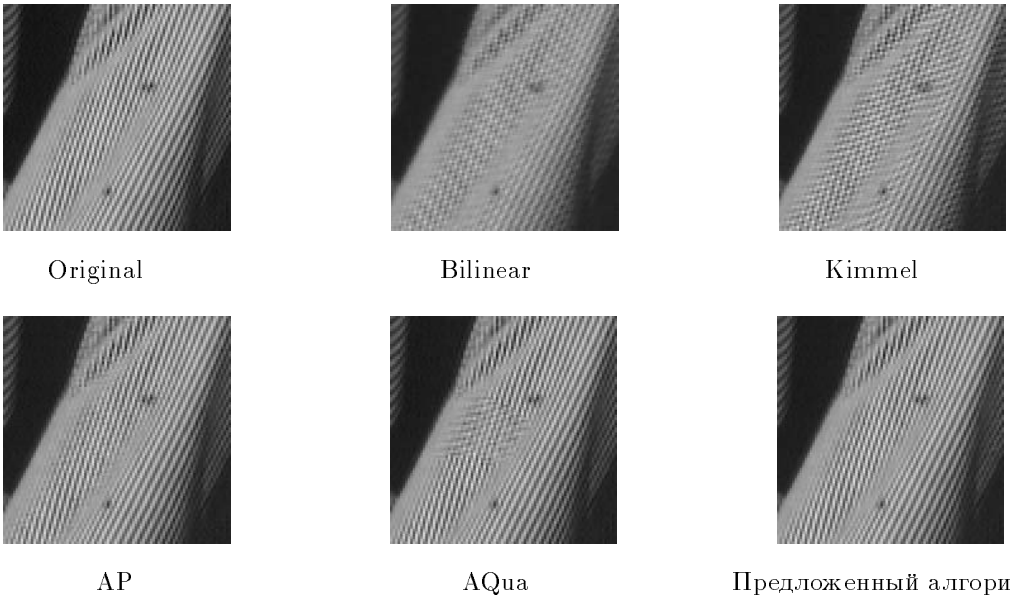


Рис. 17. Результаты работы различных методов на изображении Barbara.

Значения PSNR по каждому изображению в отдельности приведены на графике на рис. 15.

На рис. 16 и 17 приведены примеры работы различных алгоритмов. Другие примеры и иллюстрации можно найти в [11].

Перечислим особенности результирующих изображений для рассмотренных алгоритмов.

Линейные алгоритмы (билинейная и бикубическая интерполяция) показывают наихудшее качество и максимум артефактов.

Алгоритм Киммела показывает хорошие результаты на границах между различными цветами и практически устраняет эффект молнии. Однако он не способен полностью устранить цветовой муар в сложных, высокочастотных областях изображения. При увеличении числа итераций подавление муара улучшается, однако, ухудшается цветовая насыщенность изображения.

Алгоритм Aqua-2, основанный на подходе Optimal Recovery и алгоритме Киммела, демонстрирует лучшее качество за счет более сложной интерполяции зеленого цвета. К недостаткам этого метода можно отнести сильное влияние размера окна OR на вид результирующего изображения (отсутствие адаптивности размера окна) и недостаточную четкость результирующего изображения (нет проекции на исходные данные после алгоритма Киммела).

Метод чередующихся проекций (Alternating Projections) хорошо работает для изображений, в которых вектор цветового направления совпадает с осью серых цветов (в том числе – для изображений в оттенках серого). Когда же изображение перестает удовлетворять этой модели (например, вблизи границ между цветами разного тона), алгоритм производит сильный эффект молнии.

В нашем методе мы постарались объединить положительные качества рассмотренных алгоритмов и адаптивно управлять размером окна вычисления градиентов, числом итераций алгоритма Киммела и проекцией изображения на исходные данные. По результатам замеров значений PSNR и по визуальному качеству предложенный нами алгоритм превосходит все рассмотренные (и известные нам) алгоритмы интерполяции байеровских шаблонов. Временная сложность нашего метода примерно в 3 раза выше, чем у алгоритма Киммела.

СПИСОК ЛИТЕРАТУРЫ

1. *Hibbard R.H.* Apparatus and method for adaptively interpolating a full color image utilizing luminance gradients. U.S. Patent 5,382,976. January 1995.

2. *Laroche C.A., Prescott M.A.* Apparatus and method for adaptively interpolating a full color image utilizing chrominance gradients. U.S. Patent 5,373,322. December 1994.

3. *Kimmel R.* Demosaicing: image reconstruction from CCD samples. Proc. Trans. Image Processing. 1999. V. 8. P. 1221–1228.

4. *Muresan D.D.* Review of Optimal Recovery. http://dsplab.ece.cornell.edu/papers/technical_reports/review_or.pdf.

5. *Muresan D.D., Parks T.W.* Optimal Recovery Demosaicing. IASTED Signal and Image Processing Conference. Hawaii, 2002.

6. *Gunturk B.K.* et al. Color Plane Interpolation using Alternating Projections // IEEE Trans. Image Processing. 2002. V. 11. № 9. P. 997–1013.

7. *Li X., Orchard M.T.* New Edge-Directed Interpolation // IEEE Trans. On Image Processing. 2001. V. 10. № 10.

8. *Leitao J.A., Zhao M., de Haan G.* Content-adaptive video up-scaling for high-definition displays. Proc. IVCP 2003. V. 5022. January 2003.

9. *Luo M.R., Cui G., Rigg B.* The Development of the CIE 2000 Colour Difference Formula: CIEDE2000. Colour & Imaging Institute, University of Derby. UK.

10. *Lukin A.* Image Resampling Algorithms (demo web-page). http://audio.rightmark.org/lukin/graphics/resampling.htm.

11. *Лукин А., Кубасов Д.* Интерполяция байеровских шаблонов (веб-страничка). http://audio.rightmark.org/lukin/graphics/demosaicing.rus.htm.