16-я Международная конференция по компьютерной графике и ее приложениям, Институт вычислительной математики и математической геофизики СО РАН, 1-5 июля 2006 года, стр. 112-119

Алгоритм N-кратного увеличения частоты кадров видео на основе пиксельной компенсации движения с обработкой наложений

С.В. Гришин, Д.С. Ватолин
Московский Государственный Университет им. М.В. Ломоносова,
Москва, Россия
{sgrishin, dmitriy}@graphics.cs.msu.su

Аннотация

Описывается алгоритм преобразования частоты кадров видео потока, позволяющий увеличивать скорость кадров с произвольное целое число раз. Достоинством разработанного метода является его устойчивость к сложности движения, которая достигается за счет маскирования проблемных областей кадра, фильтрации «истинных векторов» и оценки точности векторов смещения в каждой точке кадра. Высокая точность интерполированных кадров также достигается за счет обработки наложений и пиксельного векторного поля. Последнее вычисляется с использованием оригинального метода восьмикратного повышения разрешения векторного поля.

Ключевые слова: *преобразование частоты кадров, наложение, истинные вектора движения, пиксельная оценка движения.*

1. ВВЕДЕНИЕ

Задача преобразования частоты кадров (ПЧК) видео имеет широкий спектр применений. Среди них самыми распространенными являются преобразование между форматами воспроизведения видео и использование в области сжатия видео.

Известно множество подходов к решению задачи ПЧК. Они отличаются друг от друга кратностью увеличения частоты кадров, используемыми алгоритмами оценки движения, механизмом борьбы с артефактами, способом обработки наложений и потому имеют различные области применения.

Существуют алгоритмы решения задачи преобразования частоты кадров с дробной кратностью преобразования. Например, в [1] описан алгоритм увеличения частоты кадров в 1.5 раза. При использовании данного метода нарушается временной интервал между исходными кадрами, что проявляется в появлении артефакта, называемого jerkiness («рывки в движении»). Артефакты такого характера очень заметны при воспроизведении. Артефакты такого же вида свойственны для алгоритма, использующего процедуру глобальной компенсации движения [2]. Важной частью любого алгоритма ПЧК является способ обработки наложений. В серии работ [3-6] для обработки наложений используются различные варианты медианной фильтрации, применяемой как к векторам движения, так и к скомпенсированным пикселям. Одним из недостатков такого подхода является неявный способ обработки наложений, который использует предположения верные достаточно редко. Это часто приводит к появлению артефактов в областях со сложным движением.

Некоторые из существующих методов [7-10] имеют ограниченную область применения и могут использоваться только на стороне декодера. Каждый из этих методов по-своему использует дополнительную доступную информацию. Алгоритм, описанный в [7], считывает из закодированного видео потока информацию о движении и сегментации кадров, используя ее для более точной обработки наложений. Алгоритм из работы [8] использует M-кадры, закодированные в потоке, для передачи информации о нелинейном движении между интерполируемыми кадрами. Таким образом, этот метод, в отличие от стандартных алгоритмов ПЧК, не использует предположения о равномерности и прямолинейности движения между опорными кадрами. Основным недостатком метода, описанного в [11], является использование очень грубой оценки вектора фона в областях наложений. Предлагается использовать нулевой вектора движения фона для обработки областей наложений, в то время как во многих случаях фон не является статичной областью.

Решение задачи увеличения числа кадров может выполняться и другими способами. Например, с помощью метода super-resolution во временной области [12]. Однако существенным ограничением данного метода является условие наличия нескольких видео последовательностей, снятых с разных камер с различными характеристиками. Это условие, как правило, не выполнено в случае задачи преобразования частоты кадров и потому описанный в [12] метод мало применим для ее решения.

В данной работе описан алгоритм увеличения частоты кадров видео потока с кратностью равной произвольному целому числу, что расширяет его область применения. Пиксельная компенсация, использование векторного поля движения высокого разрешения, метод определения истинных векторов движения позволяют достичь очень высокой точности передачи движения и свести к минимуму или полностью устранить рывки в движении (jerkiness) на преобразованной видео последовательности. Явная схема детектирования и обработки наложений, а также определение областей со сложным движением и особой обработкой пикселей в них позволяют повысить точность обработки наложений и уменьшить уровень артефактов на сценах со сложным движением.

2. ОПИСАНИЕ АЛГОРИТМА

Разработанный алгоритм позволяет увеличивать частоту кадров видео в произвольное целое число раз.

16-я Международная конференция по компьютерной графике и ее приложениям, Институт вычислительной математики и математической геофизики СО РАН, 1-5 июля 2006 года, стр. 112-119

Преобразование частоты производится за счет увеличения общего числа кадров видео последовательности, т.е. добавления новых (интерполированных) кадров. Новые кадры вставляются в исходную видео последовательность таким образом, что число интерполированных кадров между любыми двумя соседними исходными кадрами одно и то же и пропорционально кратности увеличения частоты кадров (см. Рис. 1). Преобразованный видео поток можно описать следующим образом:

$$\tilde{I}(p,n)=\begin{cases} I(p,k), & n=N\cdot k, \quad k\in\{0,1,2,...\} \\ \tilde{I}(p,l), & n=N\cdot k+l, \quad l\in\{1,..,N-1\} \end{cases},$$

$$p=\begin{pmatrix} x \\ y \end{pmatrix}, \; n\in\{0,1,...\}, \quad N\in\{2,...\},$$

где $\tilde{I}(p,n)$ – пиксель с координатами p на кадре n преобразованной видео последовательности;

$I(p,k)$ – пиксель с координатами p на кадре k исходной видео последовательности;

$\tilde{I}(p,l)$ – пиксель с координатами p на интерполированном кадре l;

p – вектор координат пикселя на кадре;

N – кратность преобразования частоты кадров.

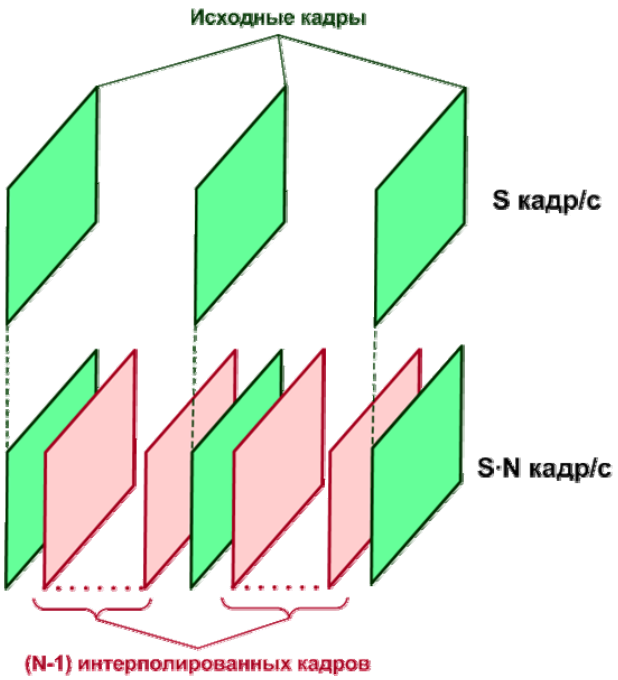

Рис. 1. Схема N-кратного увеличения частоты кадров

Преобразование частоты кадров в разработанном алгоритме представляет собой итерационный процесс, на каждой итерации которого вычисляются все интерполированные кадры между двумя соседними исходными кадрами. Далее будем называть их опорными кадрами. Опорный кадр, предшествующий блоку интерполированных кадров, будем называть предыдущим. Аналогичным образом определяется следующий опорный кадр. Обработка на каждой итерация производится по одному алгоритму, состоящему из четырех основных шагов:

1. оценка движения;
2. вычисление карты доверия векторов;
3. сегментация;
4. вычисление интерполированных кадров.

Следующие разделы содержат подробное описание указанных этапов алгоритма за исключением этапа оценки движения. В виду объемности изложения и специфичности описание этого этапа выходит за рамки данной статьи. Стоит лишь заметить, что для задачи преобразования частоты кадров критическим является свойство «истинности» векторов движения. Это означает, что направление векторов смещения должно соответствовать реальному направлению движения в каждом блоке кадра. Важно отметить, что в разработанном алгоритме ПЧК использовался метод, основанный на разбиении кадров на блоки размером 8x8.

3. ВЫЧИСЛЕНИЕ КАРТЫ ДОВЕРИЯ ВЕКТОРОВ

При вычислении значения доверия для каждого блока формируется набор из 9 векторов. Набор состоит из векторов блоков, лежащих в окрестности 3x3 блока с центром в текущем блоке. Затем вычисляется количество векторов из набора, слабо отличающихся от вектора из текущего блока. Критерием отличия служит евклидово расстояние в двумерном пространстве. Вектора считаются слабо отличающимися, если евклидово расстояние между ними не превосходит фиксированного порога *Th*:

$$\left|mv^1-mv^2\right|=\sqrt{\left(mv_x^1-mv_x^2\right)^2+\left(mv_y^1-mv_y^2\right)}\le Th,$$

$$mv^1=\begin{pmatrix} mv_x^1 \\ mv_y^1 \end{pmatrix}, \quad mv^2=\begin{pmatrix} mv_x^2 \\ mv_y^2 \end{pmatrix},$$

где mv^1, mv^2 – два вектора движения.

Если вычисленное число векторов, похожих на вектор в текущем блоке, больше заданного порога, то значение доверия равно единице, иначе – нулю. Вектора со значением доверия 1 будем называть «верными», со значением 0 – «неверными».

4. СЕГМЕНТАЦИЯ

Сегментация выполняется для блоков следующего опорного кадра на основе векторного поля, вычисленного на предыдущем этапе алгоритма. Для каждого блока вычисляется его тип. Для типа возможны три значения: «сложный», «наложение» или «обычный». Тип «обычный» определяет блоки из областей с простым движением, «сложный» – из областей со сложным движением и «наложение» соответствует блокам из областей наложений. Наложения возникают на краях движущихся объектов в случае, если вектора движения объектов не параллельны. Сегментация выполняется в три этапа:

1. определение «сложных» блоков;

2. определение блоков с типом «наложение»;
3. коррекция маски сегментации;

Таким образом, сегментация выполняется за три прохода векторного поля для следующего опорного кадра. В следующих подразделах эти этапы рассмотрены более подробно.

4.1 Определение «сложных» блоков

Для каждого блока вычисляется вектор *D* по следующим формулам:

$$D(x,y)=\begin{pmatrix}D_x\\D_y\end{pmatrix}=\frac{\sum_{i=y-1}^{y+1}\sum_{j=x-1}^{x+1}\left|mv(i,j)-m(x,y)\right|}{9},$$

$$m(x,y)=\begin{pmatrix}m_x\\m_y\end{pmatrix}=\frac{\sum_{i=y-1}^{y+1}\sum_{j=x-1}^{x+1}mv(i,j)}{9},\quad mv(i,j)=\begin{pmatrix}mv_x\\mv_y\end{pmatrix},$$

где $mv(i,j)$ – вектор смещения для блока с координатами *i* и *j*;

(x,y) – координаты текущего блока.

Вектор *D* характеризует степень различия между вектором текущего блока и векторами в соседних блоках. Чем сильнее вектор в текущем блоке отличается от векторов соседних блоков, тем больше будет модуль вектора *D*. Определение «сложных» блоков производится на основе этого вектора. Если хотя бы одна из координат вектора *D* больше заданного порога, то текущий блок считается «сложным».

4.2 Определение блоков с типом «наложение»

Вычисление типа производится только для блоков, которые не были отнесены к «сложным». Определение наличия наложения производится на основе сравнения двух векторов из соседних блоков методом, аналогичным методу из [5]. Единственным отличием используемого метода является то, что проверка на наличие наложения выполняется только в случае, если оба рассматриваемых вектора являются «верными». Важно заметить, что существует два типа наложений: «открытие» и «закрытие», классификация которых также описана в [5].

Для всех блоков с типом «наложение» сохраняются два вектора из рассматриваемых соседних блоков (для горизонтального наложения – из горизонтальных соседей, для вертикального – вертикальных). Эти векторы будут использоваться на этапе вычисления интерполированных кадров. Будем их называть векторами наложения.

4.3 Коррекция маски сегментации

Маска сегментации представляет собой двумерный массив, каждое значение которого определяет тип блока с соответствующими координатами на кадре. Коррекция маски сегментации производится с целью удаления изолированных блоков с типом «сложный» или «наложение», а также совокупностей небольшого числа блоков этих типов. Другими словами, на данном этапе производится удаление «шума» с маски сегментации (увеличение точности сегментации). В качестве алгоритма фильтрации может быть использована любая модификация традиционных алгоритмов расширения (dilation) и сужения (erosion), выполняемых последовательно.

5. ВЫЧИСЛЕНИЕ ИНТЕРПОЛИРОВАННЫХ КАДРОВ

На каждой итерации алгоритма вычисляется несколько интерполированных кадров, число которых зависит от кратности увеличения частоты кадров. Алгоритм вычисления каждого интерполированного кадра один и тот же, поэтому далее описывается способ вычисления одного интерполированного кадра.

В процессе описания алгоритма часто будет встречаться метод линейного усреднения. Этот метод описывает способ вычисления пикселей интерполированного кадра. Согласно этому способу каждый пиксель кадра вычисляется по формуле:

$$I(p,t+k)=w\cdot I(p,t)+(1-w)\cdot I(p,t+N),$$
$$w=\frac{N-k}{N},\ p=\begin{pmatrix}x\\y\end{pmatrix},\ k\in\{1,..,N-1\},\qquad(1)$$

где $I(p,t)$ – яркость пикселя с координатами *p* на кадре в момент времени *t*;

k – номер интерполированного кадра.

Вычисление интерполированного кадра производится в три этапа:

1. вычисление «точных» пикселей;
2. пост-обработка вычисленных пикселей;
3. вычисление оставшихся пикселей.

В следующих подразделах указанные этапы будут рассмотрены более детально. Важной особенностью алгоритма является способ оценки точности векторов для пикселей, описанию этого способа посвящен отдельный подраздел «Вычисление локальных ошибок векторов пикселей».

5.1 Вычисление локальных ошибок векторов пикселей

Для каждого пикселя интерполированного или опорного кадра возможно оценить точность заданного вектора. Способ вычисления этой оценки различен для пикселей из блоков с типами «обычный», «наложение закрытие» и «наложение открытие». Оценка точности вектора для пикселя из «обычного» блока вычисляется по формулам:

$$Error(p(t+N),mv(p(t+N)))=$$
$$\sum_{i=-1}^{1}\sum_{j=-1}^{1}\left|I\left(p(t+N)+\begin{pmatrix}i\\j\end{pmatrix},t+N\right)-I\left(p(t)+\begin{pmatrix}i\\j\end{pmatrix},\mathrm{t}\right)\right|,\qquad(2)$$

где $p(t),p(t+N)$ – вектора координат пикселя на предыдущем и следующем опорных кадрах соответственно.

Для вычисления ошибки вектора для пикселя на интерполированном кадре, перед тем как использовать данную формулу, необходимо вычислить координаты данного пикселя на предыдущем и следующем опорных

16-я Международная конференция по компьютерной графике и ее приложениям, Институт вычислительной математики и математической геофизики СО РАН, 1-5 июля 2006 года, стр. 112-119

кадрах. Оценку (2) будем называть локальной ошибкой вектора в пикселе $p(t+N)$.

Вычисление оценок точности векторов для пикселей из блоков с типом «наложение» полностью аналогично за тем исключением, что выражение (1) вычисляется с использованием иной пары опорных кадров, а не предыдущего и следующего. Схема вычисления этих оценок изображена на **Рис. 2**. Локальные ошибки векторов для пикселей из блоков с типом «наложение» будем называть локальными ошибками открытия и закрытия.

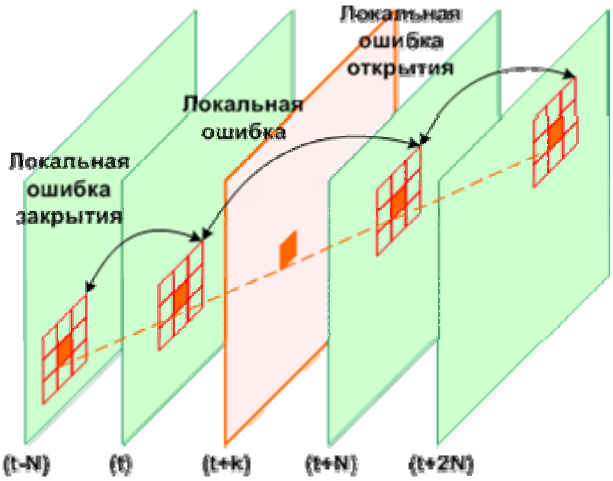

Рис. 2. Схема вычисления локальных ошибок для различных типов пикселей

5.2 Вычисление «точных» пикселей

На данном этапе вычисляются те пиксели интерполированного кадра, для которых вектора движения были найдены с высокой точностью.

Обработка производится по пикселям следующего опорного кадра сверху вниз, слева направо. Алгоритм обработки каждого пикселя следующий:

1. Если тип текущего блока – «сложный» или вектор этого блока – «неверный», то обработка переходит к следующему пикселю.
2. Вычисляется локальная ошибка для вектора текущего блока в обрабатываемом пикселе. Вычисленное значение ошибки сохраняется в специальном массиве.
3. Если локальная ошибка больше заданного порога, то обработка переходит к следующему пикселю.
4. Вычисляются координаты точки текущего пикселя на интерполированном кадре с учетом вектора движения:

$$p(t+k) = \left]p(t+N) + w \cdot mv(p(t+N))\right[$$
$$w = \frac{N-k}{N},\ k \in \{1,..,N\text{-}1\}$$

где $p(t+k)$ – координаты пикселя на интерполированном кадре номер k;

$\left]\bullet\right[$ – операция округления;

$p(t+N)$ – координаты текущего пикселя на следующем опорном кадре;

$mv(p(t+N))$ – вектор смещения текущего пикселя (равен вектору блока, содержащего текущий пиксель).

5. Если пиксель интерполированного кадра еще не был вычислен, то его значение вычисляется на следующем шаге. Иначе, сравниваются локальные ошибки вектора текущего пикселя и уже вычисленного. Если ошибка текущего пикселя меньше, то выполняется следующий шаг, иначе – обработка переходит к следующему пикселю.
6. Вычисляется значение пикселя на интерполированном кадре. Для вычисления используются формулы:

$$\begin{aligned} I(p(t+k),t+k) &= w \cdot I\big(p(t),t\big) + \\ &+ (1-w) \cdot I\big(p(t+N),t+N\big), \\ w &= \frac{N-k}{N},\ k \in \{1,..,N-1\}, \\ p(t) &= p(t+N) + mv(p(t+N)), \end{aligned} \quad (3)$$

где $p(t+k)$ – координаты пикселя на интерполированном кадре номер k, вычисленные на шаге 4. После этого, использованный вектор смещения пикселя сохраняется в специальном массиве.

5.3 Пост-обработка вычисленных пикселей

На данном этапе производится удаление небольших отдельно стоящих групп пикселей с интерполированного кадра. Значения в точках интерполированного кадра, находящихся в позициях удаленных пикселей, будут вычислены на следующих этапах. Цель данной операции – удаление пикселей, вычисленных с использованием неправильных векторов движения. Такие пиксели обычно расположены небольшими группами, окруженными пустыми пикселями. Этот этап является модификацией обычной морфологической операции, называемой сужением.

5.4 Вычисление оставшихся пикселей

Обработка производится по пикселям интерполированного кадра сверху вниз, слева направо. Для каждого пустого пикселя вычисляется его значение. Способ вычисления зависит от типа блока, в котором находится пиксель следующего опорного кадра с теми же координатами, что и текущий пиксель. В следующих подразделах рассмотрены алгоритмы вычисления значений пикселей для каждого типа блока.

5.4.1 Вычисление пикселей с типом «обычный»

Для вычисления значения пикселя необходимо определить его вектор смещения. Поиск вектора смещения производится в несколько этапов. Сначала поиск выполняется среди векторов из малой окрестности данного пикселя. Затем, если подходящий вектор не был найден, поиск производится среди векторов движения более удаленных пикселей. Если после этого подходящий вектор

движения не был найден, используется альтернативный вектор. Более детально алгоритм вычисления значения пикселя выглядит следующим образом:

1. **Формирование локального набора кандидатов:** формируется набор векторов смещения, каждый вектор которого является вектором-кандидатом текущего пикселя. Вектора данного набора выбираются из пикселей, расположенных по отношению к текущему в позициях, указанных на **Рис. 3**. Очевидно, что для формирования набора кандидатов используются только вычисленные пиксели.

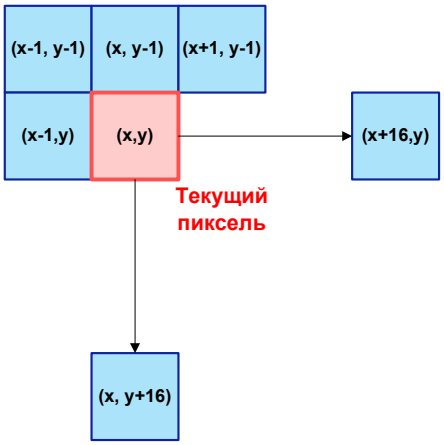

Рис. 3. Схема расположения локальных соседей пикселя

2. **Поиск вектора в наборе кандидатов:** из набора кандидатов выбирается вектор с наименьшей локальной ошибкой. Если ошибка этого вектора меньше заданного порога, то данный вектор выбирается в качестве текущего и выполняется шаг 5.
3. **Добавление глобальных кандидатов в набор:** в набор кандидатов добавляются вектора вычисленных пикселей, отстоящих от текущего на заданном расстоянии R (см. Рис. 4). Это расстояние увеличивается при каждом выполнении данного шага. Если данное расстояние становится больше заданного порога, то обработка переходит на шаг 4, иначе – на шаг 2.
4. **Вычисление альтернативного вектора**: если набор кандидатов пуст, то в качестве текущего вектора выбирается нулевой, иначе – вектор из набора с наименьшей локальной ошибкой.
5. **Вычисление значения пикселя:** вычисление значения пикселя производится по формуле (3) с использованием вектора, выбранного в качестве текущего. Текущий вектор и его локальная ошибка сохраняются в специальных массивах, и обработка переходит к следующему пикселю.

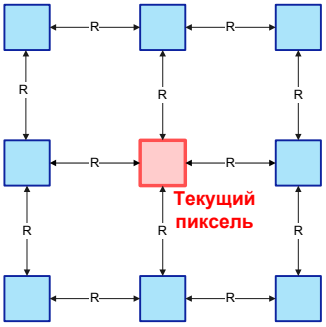

Рис. 4. Схема расположения глобальных соседей пикселя

Стоит заметить, что в каждый момент времени набор кандидатов содержит только «верные» вектора движения, что значительно увеличивает точность передачи движения всего алгоритма.

5.4.2 Вычисление пикселей с типом «сложный»

Способ вычисления значений пикселей этого типа очень похож на способ, описанный в предыдущем подразделе. Первым отличием является использование другой формулы в случае, если найден вектор движения с допустимой локальной ошибкой. Для вычисления значений «сложных» пикселей используется соотношение:

$$I(p(t+k),t+k) = \alpha \cdot V_i + (1-\alpha)\cdot V_{la},$$

$$\alpha = \frac{Th - Error}{Th + Error},$$

где $p(t+k)$ – координаты пикселя на интерполированном кадре номер k;

V_i – значение пикселя, вычисленное по формуле (3);

V_{la} – значение пикселя, вычисленное методом линейного усреднения по формуле (1);

Th – фиксированный порог, определяющий степень размытия в зависимости от точности найденного вектора движения для пикселя;

$Error$ – локальная ошибка найденного вектора смещения (см. формулу (2)).

Вторым отличием от предыдущего метода является способ выбора альтернативного вектора. Сначала поиск альтернативного вектора осуществляется среди локальных кандидатов. Если же локальных кандидатов нет, то поиск осуществляется во всем наборе кандидатов. В обоих случаях выбирается кандидат с минимальной локальной ошибкой. В случае если набор кандидатов пуст, то для вычисления значения пикселя используется метод линейного усреднения (формула (1)).

5.4.3 Вычисление пикселей с типом «наложение»

Вычисление значений пикселей с типом «наложение» производится согласно следующему алгоритму:

1. **Попытка вычислить текущий пиксель как «обычный»:** если текущий пиксель расположен не

возле края кадра и в его окрестности 3x3 есть хотя бы один вычисленный «обычный» пиксель, то выполняется поиск вектора методом, описанным в подразделе 5.4.1. Если допустимый вектор был найден, то значение пикселя вычисляется способом из подраздела 5.4.1.

2. **Обработка наложений возле краев кадра:** если текущий пиксель расположен возле края кадра и один из векторов наложения равен нулевому вектору, то ненулевой вектор наложения выбирается в качестве текущего и выполняется шаг 4.
3. **Выбор текущего вектора:** в качестве текущего вектора выбирается тот вектор наложения, который имеет меньшую локальную ошибку наложения соответствующего типа (см. раздел «Вычисление локальных ошибок векторов пикселей»).
4. **Вычисление значения пикселя:** способ вычисления значения пикселя зависит от типа наложения. Для пикселей с наложением «закрытие» используется формула:

$$I(p(t+k),t+k) = \alpha \cdot V_{prev} + (1-\alpha)\cdot V_{la},$$
$$\alpha = \frac{Th - Error}{Th + Error}, V_{prev} = I(p(t),t),$$
$$p(t) = p(t+k) + (1-w)\cdot mv,$$
$$w = \frac{N-k}{N}, k \in \{1,..,N-1\},$$

а в случае наложения «открытие»:

$$I(p(t+k),t+k) = \alpha \cdot V_{next} + (1-f)\cdot V_{la},$$
$$f = \frac{Th - Error}{Th + Error}, V_{next} = I(p(t+N),t+N),$$
$$p(t+N) = p(t+k) - w\cdot mv,$$
$$w = \frac{N-k}{N}, k \in \{1,..,N-1\},$$

где $p(t+k)$ – координаты текущего пикселя на интерполированном кадре;

V_{prev}, V_{next} – пиксели предыдущего и следующего кадра соответственно;

$p(t), p(t+N)$ – координаты того же пикселя на предыдущем и следующем опорных кадрах соответственно;

Th – заданный порог, определяющий степень размытия в зависимости от локальной ошибки наложения найденного вектора;

$Error$ – локальная ошибка наложения соответствующего типа (см. раздел «Вычисление локальных ошибок векторов пикселей»).

6. РЕЗУЛЬТАТЫ

С целью проверки эффективности разработанного метода было произведено его сравнение с тремя методами [4] (см. Табл. 1) на нескольких последовательностях (см. Табл. 2). В последовательностях 1, 3 и 4 преобладает слабое движение, остальные отличаются более сильным движением. Для обеспечения возможности использования объективной метрики исходные последовательности прореживались в два раза, а затем производилось двукратное увеличение скорости кадров с использованием какого-либо метода. Эта операция делает возможным сравнение интерполированных и исходных кадров с использованием метрики PSNR яркостной компоненты (Y-PSNR). Для построения графиков и во всех расчетах использовались значения метрики только для интерполированных кадров.

Название метода	Используемое сокращение
Motion Compensated Averaging	MC AVG
Static Median	SM
Dynamic Median	DM

Табл. 1. Методы, использованные в сравнении

Нетрудно видеть (см. Рис. 5), что на большинстве кадров разработанный метод достигает заметно более высоких показателей метрики по сравнению с другими алгоритмами.

№	Название	Разрешение	Частота кадров (кадр/с)
1	waterfall	352x288	15
2	bus	352x288	15
3	foreman	352x288	15
4	susi	352x288	15
5	table tennis	352x288	15
6	flower garden	352x240	15

Табл. 2. Последовательности, использованные в сравнении

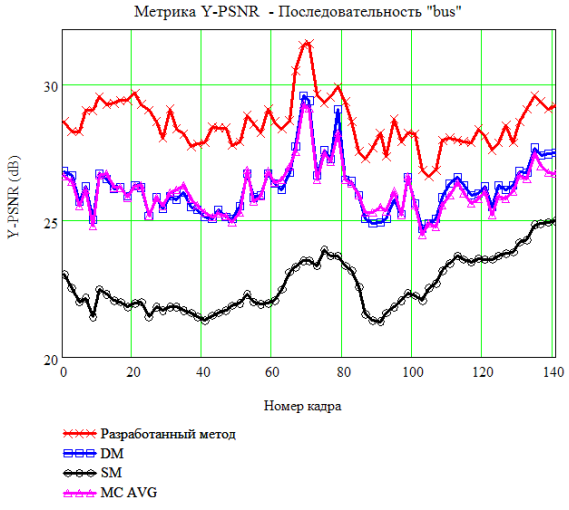

Рис. 5. Метрика интерполированных кадров для последовательности «bus»

На Рис. 6 изображен график сравнения методов с алгоритмом SM на всех последовательностях. По оси ординат этого графика отложена разница усредненных значений метрики каждого метода с методом SM. Такой способ построения графика позволяет легко оценивать величину разницы показателей метрики для каждого метода на каждой последовательности. Легко видеть, что разработанный метод достигает наибольшего преимущества в почти 3дБ на последовательностях с сильным движением.

На Рис. 7 - Рис. 10 очевидно визуальное преимущество разработанного алгоритма. В областях со сложным движением из-за неверно найденных векторов все рассмотренные алгоритмы, кроме разработанного, дают видимые артефакты.

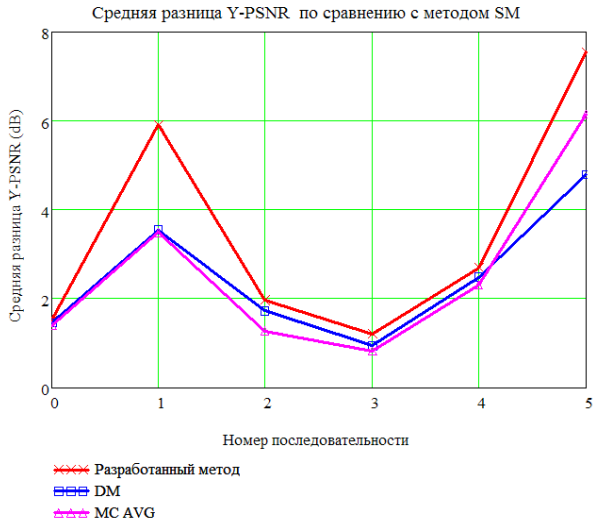

Рис. 6. Разница усредненных значений метрики по отношению к методу SM

7. ЗАКЛЮЧЕНИЕ

Разработанный алгоритм позволяет увеличивать частоту кадров в видео с кратностью, равной произвольному целому числу. Результаты сравнения разработанного алгоритма с другими методами показали его значительное преимущество в визуальном качестве. Преимущество разработанного метода в терминах объективной метрики Y-PSNR достигает на некоторых последовательностях 3дБ. При этом скорость алгоритма при 4-кратном увеличении частоты кадров последовательности с разрешением 352x288 составляет 30 кадр/с на процессоре Pentium IV 2.4 ГГц.

8. БИБЛИОГРАФИЯ

[1] R. Castagno, P. Haavisto, G. Ramponi. *“A Method for Motion Adaptive Frame Rate Up-Conversion”. IEEE Transactions on circuits and Systems for Video Technology, Vol 6, No. 5, October 1996.*

[2] Sanjeev Kumar, Mainak Biswas and Truong Q Nguyen. *“Global Motion Estimation in Frequency and Spatial Domain”, IEEE International Conference on Speech, Acoustics, and Signal Processing, March 19-23, 2004.*

[3] Gerard de Haan, Paul W.A.C. Biezen, Henk Huijgen, and Olukayode A. Ojo. *“Graceful Degradation in Motion-Compensated Field-Rate Conversion”, Signal Processing of HDTV, V,L.Stenger, L. Chiariglione and M. Akgun (Eds.), Elsevier,1994.*

[4] Olukayode Anthony Ojo and Gerard de Haan. *“Robust motion-compensated video upconversion”, IEEE Transactions on. Consumer Electronics, Volume 43 4, November 1997, Pages 1045-1056.*

[5] A. Pelagotti and G. de Haan. *“High quality picture rate up-conversion for video on TV and PC”, Proc. Philips Conf. on Digital Signal Processing, Nov. 1999, paper 4.1, Veldhoven (NL).*

[6] A. Pelagotti and G. de Haan. *“A New Algorithm for High Quality Video Format Conversion”, Proceedings of the International Conference on Image Processing (ICIP2001), Oct. 2001, Thessaloniki, Greece, pp. 375-378.*

[7] Soo-Chul Han and John W. Woods. *“Frame-rate Up-conversion Using Transmitted Motion and Segmentation Fields for Very Low Bit-rate Video Coding”, Proc. IEEE International Conference on Image Processing,vol. 1, pp. 747-750, Oct. 1997.*

[8] Tien-ying Kuo, JongWon Kim and C.-C. Jay Kuo. *“Motion-Compensated Frame Interpolation Scheme for H.263 Codec”. Proc. ISCAS 99, May 1999. 27.*

[9] Shan Liu, JongWon Kim and C.-C. Jay Kuo. *“Non-Linear Motion-Compensated Interpolation for Low Bit Rate Video”, in SPIE Proceeding of International Symposium on Optical Science, Engineering, and Instrumentation, Applications of Digital Image Processing XXIII, San Diego, CA, July, 2000.*

[10] Ravi Krishnamurthy, John W. Woods and Piere Moulin. *“Frame Interpolation and Bidirectional Prediction of Video using Compactly-Encoded Optical Flow Fields and Label Fields”, IEEE Trans. Circ. Syst. Video Tech, Vol. 9, No. 5, pp. 713-726, Aug. 1999.*

[11] Y.-K. Chen, A. Vetro, H. Sun, and S. Y. Kung. *"Frame-rate up-conversion using transmitted true motion vectors," in Proc. IEEE Second Workshop on Multimedia Signal Processing, pp. 622-627, Dec. 1998.*

[12] E. Shechtman, Y. Caspi, and M. Irani. *"Increasing space-time resolution in video," in Proc. of the 7th European Conference on Computer Vision (ECCV), ser. Lecture Notes in Computer Science, vol. 2350. Springer-Verlag Heidelberg, 2002, pp. 753-768.*

Об авторах

Ватолин Дмитрий Сергеевич — специалист в области сжатия изображений и видео с двенадцатилетнем стажем. К.ф.-м.н., IEEE Member, старший научный сотрудник лаборатории компьютерной графики ВМиК МГУ, руководитель ВидеоГруппы. Автор книг "Методы сжатия изображений" (Д. Ватолин) и "Методы сжатия данных" (Д. Ватолин, А. Ратушняк, М. Смирнов, В. Юкин). Основатель серверов «Все о сжатии» http://compression.ru/ и “Compression Links” http://www.compression-links.info/.

E-mail: dmitriy@graphics.cs.msu.su

Гришин Сергей Викторович – аспирант, сотрудник лаборатории компьютерной графики ВМиК МГУ.

16-я Международная конференция по компьютерной графике и ее приложениям, Институт вычислительной математики и математической геофизики СО РАН, 1-5 июля 2006 года, стр. 112-119

E-mail: sgrishin@graphics.cs.msu.su

Адрес: Москва, 119899, Воробьевы горы, МГУ, 2-й учебный корпус, факультет ВМиК, лаборатория компьютерной графики.

N-times Video Frame-rate Up-conversion Algorithm based on Pixel Motion Compensation with Occlusions Processing

Abstract

Frame rate up-conversion algorithm is described which allows increasing the frame rate with factor equal to any integer number. Advantage of this method is its robustness with respect to motion complexity. This robustness is achieved by applying complex areas masking, true motion vectors detection and vectors accuracy estimation at each point of the frame. Further interpolated frame accuracy gain is achieved by occlusions processing and using of pixel motion vector field, which is calculated by original method of eight-times motion vector field scaling.

Keywords: *frame rate conversion, true motion vectors, pixel motion estimation*.

About the author(s)

Dmitriy Vatolin is an expert in image, video and data compression with more than 12 years experience. Ph.D. in graphics compression, IEEE Member. Books: "Image compression algorithms" (D.Vatolin), "Data compression methods" (D.Vatolin, A.Ratushniak, M.Smirnov, V.Yukin). Senior researcher at the Laboratory of Computer Graphics and Multimedia at Moscow State University, Department of Computational Mathematics and Cybernetics. Head of VideoGroup. Founder of “The Compression Project” http://compression.ru/ and “Compression Links” http://www.compression-links.info/.

E-mail: dmitriy@graphics.cs.msu.su

Sergey Grishin is a post-graduate student, collaborator of Graphics&Media Lab at Moscow State University, Department of Computational Mathematics and Cybernetics. His contact email is sgrishin@graphics.cs.msu.su.

Рис. 7. Кадр разработанного метода

Рис. 8. Кадр MC AVG

Рис. 9. Кадр DM

Рис. 10. Кадр SM