# Алгоритмы детектирования и анализа движения для системы DVR

Альберт Ахриев Александр Болтнев

27 июля 2006

## 1 Введение

В данной работе от нас требовалось создать алгоритм детектирования движения, способный обнаруживать достаточно мелкие области движения ($8 \times 8$ пикселей) и выдавать предупреждения в виде рамок вокруг группы изменившихся точек.

Отличительной особенностью системы DVR является крайне низкая скорость скорость кадров (1.5-2 fps), что наложило существенные ограничения на алгоритмы детектирования движения. Главная проблема состоит в том, что при такой низкой скорости видео, объект может появиться только в одном кадре. Следовательно, мы должны либо сразу принять область изменения в качестве потенциального объекта, либо отвергнуть эту гипотезу, не имея зачастую возможности наблюдать движение сколько-нибудь продолжительное время.

Другой круг проблем, которые пришлось решать при создании алгоритма, связан с качеством видео. Само же качество видео резко ухудшилось с тех пор, как мы начали работу с платами видео-захвата `Techwell`. По нашим предварительным оценкам, уровень помех на этой плате примерно в 2 раза выше чем на плате `BT`.

Также сложным моментами являются изменения освещенности в комнате и постоянная подстройка камеры. Для борьбы с последними был разработан алгоритм коррекции изменения яркости (см. далее). После применения алгоритма коррекции яркости пришлось также усложнить детектор движения, так как помехи все же пробивали заданные пороги. Было замечено что на более темных участках изображения помехи оказывались более значительными и чаще превышали порог. Поэтому была подобрана эмпирическая зависимость порога от яркости, уменьшающая нежелательные эффекты и не требующая повышения порога по всему изображению.

Метод детектирования изложенный в этом отчете условно можно разбить на две части. В первой части мы обнаруживаем точки двух типов – точки сильного и слабого изменения яркости. Главная задача на этом этапе максимально избавиться от помех. Во второй части эти точки объединяются в области изменения (потенциального движения). Области прошедшие отбор согласно заданным критериям помечаются как области движения. Кроме того, производится анализ простых событий (типа появления области движения) на предмет объединения их в более сложные тревожные ситуации (*alarms*).

## 2 Детектирование движения (*А.Ахриев, А.Болтнев*)

Для каждой точки кадра видеопоследовательности требуется определить, есть ли в ней движение или нет.

1. Сначала производится понижения разрешения изображения в $k = 1$ (разрешение не понижается),$k = 2$ или $k = 4$ раза (переменная DOWNSCALE). Это повышает производительность алгоритма и несколько снижает уровень шума.

2. Далее происходит процедура выравнивания яркости. Для данной точки, сначала вычисляется средние значения яркости $\overline{y}$ (канал Y) и цветности $\overline{u}$, $\overline{v}$ (каналы U и V) по квадрату $d \times d$ пикселов, а затем производится преобразование по формуле

$$\begin{aligned} y &\leftarrow \frac{\mathrm{g} \cdot (d^2 \cdot y)}{\overline{y}} \\ u &\leftarrow \frac{\mathrm{g} \cdot (d^2 \cdot u - \overline{u})}{\overline{y}} \\ v &\leftarrow \frac{\mathrm{g} \cdot (d^2 \cdot v - \overline{v})}{\overline{y}} \end{aligned}$$

   Здесь g – константа, задающая диапазон значений в полученном изображении. Данная процедура может быть реализована очень эффективно.

   Вычитание $d^2 \cdot u - \overline{u}$ и $d^2 \cdot v - \overline{v}$ связано с представлением цветного изображения в цветовом пространстве Y, U и V.

3. Если текущий кадр – первый в видеопоследовательности, то устанавливаются начальные значения фона (текущим кадром) и порогов (некоторым значением, сейчас это 5). Начальные значения порогов влияют лишь конечное, достаточно непродолжительное время, поэтому значение 5 подобрано экспериментально.

4. Далее в каждой точке для каждого канала Y, U, V вычисляется разность фона и текущего кадра $\Delta$ (для каждого канала свой $\Delta$)и оценивается движение. Конкретнее, если $\Delta > T_c/2$, точка помечается как подозрительная, если $\Delta > T_c$, точка помечается как движущаяся. *Порог коррекции яркости* $T_c$ зависит от значения яркости $\overline{y}$:

$$\overline{y}^* = \frac{\overline{y} + 0.5 \cdot dim^2 \cdot k}{dim^2 \cdot k},$$

$$T_c = C \cdot \mathrm{f}(\overline{y}^*),$$

$$\mathrm{f}(x) = 1.3 + \frac{0.7 \cdot s^3}{x^3 + s^3},$$

   Здесь $4 < C < 6$, $s = 128$. Чем меньше $C$, тем чувствительней алгоритм. Этот параметр может варьироваться извне алгоритма, через метод

`SetData()` класса `CameraAnalyzer`. Для этого используется класс `TAlgoSensitivity`. Значение 0 соответствует $C = 6$, значение 1 соответствует $C = 4$. Диапазон изменения $C$ можно легко изменить.

Такая зависимость учитывает, что после коррекции яркости в областях с более низкой яркостью усиливаются шумы и помехи и в более темных местах порог ставится выше.

5. Для каждой движущейся точки считается время в движении. Если оно превышает некоторое значение HOLE_TIMEOUT (задается в параметрах), то точка считается дыркой, и пометка о движении снимается, счетчик времени движения обнуляется.

6. Если точка $(x, y)$ не помечена как движущаяся, то в ней происходит обновление фона и значений порогов.

$$B(x, y) = \alpha \cdot B(x, y) + (1 - \alpha) \cdot F(x, y), \tag{1}$$

где $B(x, y)$, $F(x, y)$ – значения интенсивности в точке $(x, y)$ фона (*background*) и текущего кадра (*frame*) соответственно.

Значения порогов обновляются несколько хитрее, в зависимости от того, растет ли порог или уменьшается:

$$\begin{aligned}
&\text{if } (d_{x,y} < T_{x,y}) \\
&\qquad T_{x,y} = \alpha_{slow} \cdot T_{x,y} + (1 - \alpha_{slow}) \cdot d_{x,y} \\
&\text{else} \\
&\qquad T_{x,y} = \alpha \cdot T_{x,y} + (1 - \alpha) \cdot d_{x,y},
\end{aligned}$$

причем значение порога ограничено снизу величиной девиации шума

$$T_{x,y} = \min\left(T_{x,y}, \sigma_{noise}\right).$$

Здесь $\alpha_{slow} = 0.5\alpha$, ($0 < \alpha < 1, \alpha$, сейчас $\alpha = 0.02$), $\sigma_{noise}$ является оценкой шума, считается как усредненная по всем не движущемся точкам разности значения пиксела на текущем и предыдущем кадре (Для каждого канала свое значение).

7. Далее по точкам движения строится прямоугольники движения, при этом учитываются и подозрительные точки. Последовательно перебираются точки движения. Для каждой из этих точек рекурсивно во всех направлениях перебираются точки движения и подозрительные точки на движение, во и все они помечаются как движущиеся. То есть подозрительные точки помечаются как движущиеся, если касаются точек нормального движения. Затем строятся описывающие прямоугольники полученных областей и выдаются как рамки движения.

# 3 Анализ прямоугольников движения (*А.Болтнев*)

1. На каждом кадре выбирается $K$ (сейчас $K = 3$) самых больших прямоугольника (по площади). Затем каждый их этих К больших прямоугольни-

ков объединяется с оставшимися маленькими прямоугольниками, пересекающимися с ним(под объединением понимается минимальный прямоугольник, содержащий данные). Затем маленькие прямоугольники отбрасываются. Данная процедура позволяет сосредоточить внимание на самом мощном движении.

2. Из потока прямоугольников формируются *события движения*. Событие движения может быть открытым или закрытым. Оно состоит из времени начала движения, времени конца движения и некоторых прямоугольников, характеризующих движение. Прямоугольники содержат время своего создания, с точностью 1 секунда (все используют зачем-то time_t). Координаты прямоугольников представлены в формате с плавающей запятой, состоят из чисел от 0 до 1 (в долях размера кадра).

   Событие движения соответствует своей камере.

   Как только поступают рамки движения, открывается событие движения. Оно существует открытым, пока движение не прервется на TMaxPause секунд (задается в параметрах, сейчас равно 1с). Событие может существовать лишь конечное время TMaxLife (задается в параметрах, сейчас равно 7с.) Если же движение прерывается или существует слишком долго, то оно становится закрытым и данные о нем сохраняются в базе данных. Это является аналогом аларма в Orwell. На данный момент эта часть алгоритма используется слабо (в поток пишут все подряд рамки). По началу она использовалась более активно, но в какой-то момент перестали писать рамки в базу данных и стали писать их прямо в видео последовательность.

3. *Короткой вспышкой* считается закрытое событие, все рамки которого имеют одно и то же время. Во времена когда видео было совсем неустойчивым (пропадали или прыгали с камеры на камеру кадры, иногда появлялись огромные полосы) короткие вспышки были некоторым шагом к решению проблемы. Сейчас можно от этого отказаться, но пока такие события не выводятся наружу.

   О том какие рамки содержит событие движения.

   Событие движения содержит не все рамки, поступающие с каждой камеры. Если камер немного и фреймрейт достаточно выскокий, их может оказаться достаточно много (сейчас до $K = 3$ рамок с каждого кадра, см. выше). Время в системе дискретизировано с точностью до 1 секунды. Это достаточно большое время, поэтому естественно отбрасывать лишние рамки в каждом периоде дискретизациии.

   Сейчас отбрасывание происходит следующим способом. Рамки сортируются по площади. Затем, начиная с самой большой рамки, смотрится, как она пересекается с остальными, попавшими в тот же период дискретизации времени (1 сек). Если пересечение занимает достаточно много (сейчас этот параметр равен 50 процентов) от большей рамки, то она заменяется на более новую меньшую (объект немного сместился). В противном случае, меньшая рамка удаляется, при условии что она попала в пересечение более чем на 50 процентов.

# 4 Приложение 1. Список параметров алгоритма

## 4.1 Параметры, изменяемые пользователем

1. Чувствительность к движению. Число от 0 до 1, 0 – низкая чувствительность, 1 – высокая чувствительность.

## 4.2 Параметры, недоступные для пользователя (доступны для разработчиков)

Примечание: по английским комментариям легко обнаружить переменные с этими параметрами в программе.

1. Скорость обновления фона (background update rate). Число от 0.001 до 1; значение по умолчанию 0.05.

2. Время существования дырки (timeout for a hole existence in seconds). Число от 1 до 100; значение по умолчанию 3.

3. Уровень сигнал/шум (signal/noise threshold ration for change detection). Число от 1 до 100; значение по умолчанию 3. Этот параметр – константа из пункта 4 раздела 2.

4. Минимальный периметр региона движения(the minimal perimeter of detected motion region). Число от 1 до 1000; значение по умолчанию 10.

5. Минимальная площадь региона движения (the minimal area of detected motion region). Число от 1 до 10000; значение по умолчанию 64.

6. Максимальное количество рамок на одном кадре(maximal number of rectangles on one frame). Число от 1 до 16; значение по умолчанию 3.

7. Максимальное время прерывания события движения, в секундах (maximal time to motion pause for one event, in seconds). Число от 0.1 до 5, значение по умолчанию 1.

8. Максимальное время жизни события движения, в секундах (maximal time to event to live, in seconds). Число от 0.1 до 60, значение по умолчанию 7.