УДК 519.6

Эффективный метод отслеживания перемещений головы в видеопотоке на основе трехмерной(3D) модели

Середенко Евгений Сергеевич
Московский Государственный Университет им. М.В. Ломоносова

Введение

Задача отслеживания перемещений лица в видео (*face tracking*) является одной из важнейших задач машинной графики и компьютерного зрения (*computer vision*). Постановка задачи: необходимо располагая данными о координатах лица на первом кадре получить информацию о перемещениях лица на всех последующих кадрах. Данная задача является основой для многих современных алгоритмов сжатия видеоинформации. Ее реализации являются важной частью современных интеллектуальных интерфейсов пользователя.

Большинство алгоритмов решения данной задачи опираются на использование данных о форме, цвете, размерах лица. Как правило, используется информация о допустимых изменениях положения лица в пространстве. Результатом работы подавляющего большинства алгоритмов является набор двумерных векторов перемещения лица на изображении от кадра к кадру [2].

Целью данной работы было построение надежного алгоритма, малочувствительного к природным особенностям лица, малочувствительного к окружающей среде и дефектам входных данных. В качестве результата алгоритм должен возвращать шестимерный вектор перемещения лица (3 координаты в пространстве перемещений и 3 координаты в пространстве поворотов).

Описание алгоритма слежения за лицом

Представленный алгоритм базируется на использовании текстурированной трехмерной модели лица, построенной на основе первого кадра. Далее на каждом кадре алгоритм ищет такие вектора перемещения модели, чтобы разница между очередным кадром и перемещённой моделью была минимальна.

Изображением (кадром) называется двумерный набор троек чисел (каждая тройка – точка, каждое число – цвет точки в палитре RGB)

$$\mathrm{I}(u, v) = (r_{uv},\ g_{uv},\ b_{uv});\ (u, v) \in R \times R$$

Черной (нулевой) назовём точку, у которой все цветовые компоненты равны нулю. Нулевым изображением назовем изображение состоящее только из чёрных точек.

Заметим, что изображение является двумерным пространством, а значит необходимо иметь отображение из трехмерного пространства лица в пространство изображения (обозначим оператор проектирования *Pr*)

$$Pr: R^3 \to R^2$$

В качестве модели лица используется полигональное приближение эллипсоида (а вернее его выпуклой части относительно наблюдателя), т.е. сетка точек удовлетворяющих

$$\begin{cases} x = x_0 \ldots x_m;\ \ x_i = x_{i-1} + \Delta_x;\ \ x_0 = -1;\ x_m = 1 \\ y = y_0 \ldots y_n;\ \ y_i = y_{i-1} + \Delta_y;\ y_0 = -1;\ y_n = 1 \\ z = z_0 \ldots z_k;\ \ z_i = z_{i-1} + \Delta_z;\ \ z_0 = 0;\ z_k = 1 \\ \dfrac{x^2}{a^2} + \dfrac{y^2}{b^2} + \dfrac{z^2}{c^2} = 1 \end{cases}$$

Приближение головы человека эллипсоидом является оправданным из-за простоты и универсальности модели. Проекция сетки эллипсоида на исходное изображение даст текстурные координаты для сетки. Таким образом,

результатом обработки первого кадра является трехмерная текстурированная модель лица. При чём проекция модели лица на текстуру в точности совпадает с самой текстурой в области лица [1, 3].

Разницей между двумя изображениями назовём изображение, полученное вычитанием по модулю всех значений цветов точек одного изображения из соответствующих значений другого изображения.

Если на очередном кадре обнаружилась ненулевая разница между проекцией модели лица и самим кадром, то можно утверждать, что снимаемое лицо изменило своё положение. Алгоритм пытается так переместить модель, чтобы минимизировать разницу между кадром и проекцией.

Перемещение модели, это шестимерный вектор τ. При чём целью является минимизировать значение

$$\begin{cases} \sum_{u,v} \rho(L \times Pr \times R(\tau) \times T(\tau) \times M,\ L \times I_{current}) \rightarrow min \\ \tau = [x, y, z, \alpha, \beta, \gamma] \end{cases},$$

Где R – матрица вращения, T – матрица перемещения, M – модель лица, $I_{current}$ – текущий кадр, $\rho(I_1(u,v), I_2(u,v))$ – разность между точками изображений I_1 и I_2 (сумма квадратов разностей между цветовыми компонентами), L – оператор выравнивания освещённости.

Поиск минимума многомерной функции ведётся при помощи «*жадного*» алгоритма на основе *квадродерева*, т.е. на каждом шаге выбирается двумерное сечение (т.е. фиксируются четыре из шести координат, по которым идёт поиск). В оставшемся двумерном пространстве просчитывается и сравнивается значение минимизируемой функции в текущей точке (p, q) и в точках ($p \pm step$, $q \pm step$), после чего текущая точка перемещается в точку минимума среди просчитанных точек, а *step* уменьшается вдвое и процедура повторяется ещё раз. Начальное значение точки – это расположение модели на предыдущем кадре. Начальное значение *step* выбирается равным максимальному из радиусов эллипсоида.

Выравнивание освещенности

Выравнивание освещённости происходит за счёт особенностей одного из представлений цветовой палитры. Так, любой цвет из палитры RGB можно однозначно представить в виде трёх чисел (*h, s, i*), где *h[hue]* – чистый цвет в палитре, *s[saturation]* – чистота цвета и *i[intensity]* – интенсивность (яркость) цвета. При чём возможно и обратное однозначное отображение. В результате исследований выяснилось, что изменение освещения влияет в основном именно на компоненту *i* hsi-представления.

Был разработан и реализован следующий алгоритм: в каждой строке интересующей области вычисляется средняя интенсивность цвета, а также вычисляется средняя интенсивность цвета по всем точкам интересующей области. После чего значение интенсивности в каждой точке каждой строки сдвигается таким образом, чтобы среднее значение внутри строки стало равно общему среднему значению. После этого аналогичная операция выполняется для всех столбцов области. Таким образом, в основном сохраняя разницу между соседними точками, почти полностью ликвидируется неравномерность освещения.

Результаты

Алгоритм неплохо показал себя в разнообразных ситуациях.

Основную свою задачу – отслеживание перемещения лица в видеопотоке алгоритм успешно выполняет. Реализация алгоритма показала интересную его особенность – даже если на очередном кадре алгоритм нашёл перемещение не совсем точно, то спустя всего несколько кадров он найдёт правильное перемещение и продолжит работу без каких-либо проблем. Это проявления минимизации области перебора за счёт построения двумерных разрезов и поиска движения только внутри данного разреза на очередном шаге. Но на очередном кадре алгоритм начинать поиск сначала, а значит, имеет

возможность улучшить найденные вектора во всех направлениях.

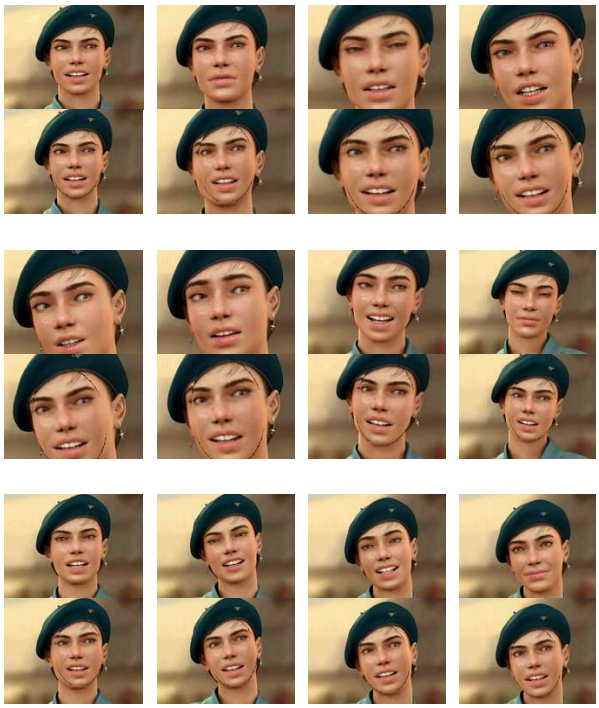

Рис.1. Работа алгоритма на динамичном движении

Алгоритм хорошо проявил себя на видеопоследовательностях с высоким уровнем шума. Деструктивные действия пользователя, например появление в кадре посторонних предметов, не сильно мешают корректной работе алгоритма. Алгоритм может работать на очень низких скоростях потока кадров (*low framerate*). Так скорость потока в один кадр в секунду является вполне достаточной для устойчивой работы алгоритма. Кроме того, алгоритм устойчив к долговременному прерыванию сигнала и способен продолжать работу без повторной инициализации. Это достигается за счёт того огромного объёма информации, который несёт в себе текстурированная модель.

Рис.2. Устойчивость алгоритма к качеству исходных данных

Алгоритм выравнивания освещенности тоже показал себя достойно и пракически полностью решает проблему неравномерного и динамического освещения.

Рис.3. Выравнивание освещенности

Список литературы

1. *Bottino A.* Real-Time Head Tracking From Uncalibrated Monocular Views // Proceedings of 5th Asian Conference on Computer Vision (ACCV 2002), Melbourne, Australia January 23-25, 2002
2. *Vezhnevets V.* Face and facial feature tracking for natural Human-Computer Interface // Proceedings Graphicon - 2002, September 16 - September 21, 2002, pp. 86-90
3. *Colmenarez A., Lopez R., Huang, T.S.* 3D model-based head tracking // Proceedings Visual Communications and Image Processing, San Jose, 1997