

Исправление стерео видео

Вячеслав Нападовский

Video Group

CS MSU Graphics & Media Lab



Содержание

- **Введение**
- Цветокоррекция
- Исправление геометрии
- Временная синхронизация
- Заключение

Цветокоррекция

Пример проблемного видео



левый ракурс



правый ракурс

Цветокоррекция

Актуальность

Причины:

- Различия в матрицах камер
- Разное положение камер относительно источника света

Проблемы, возникающие в стерео видео:

- Глобальное несоответствие цветов между ракурсами
- Наличие/отсутствие бликов на объектах на разных ракурсах

Исправление геометрии

Причины:

- Сложность корректной установки камер
- Сложность грамотной настройки камер
- Трудность сохранения их фиксированного положения

Исправление геометрии

Проблемы

Вертикальный параллакс



левый ракурс



правый ракурс

Исправление геометрии

Проблемы (2)

Поворот



левый ракурс



правый ракурс

Исправление геометрии

Проблемы (3)

Искажение вертикальной перспективы

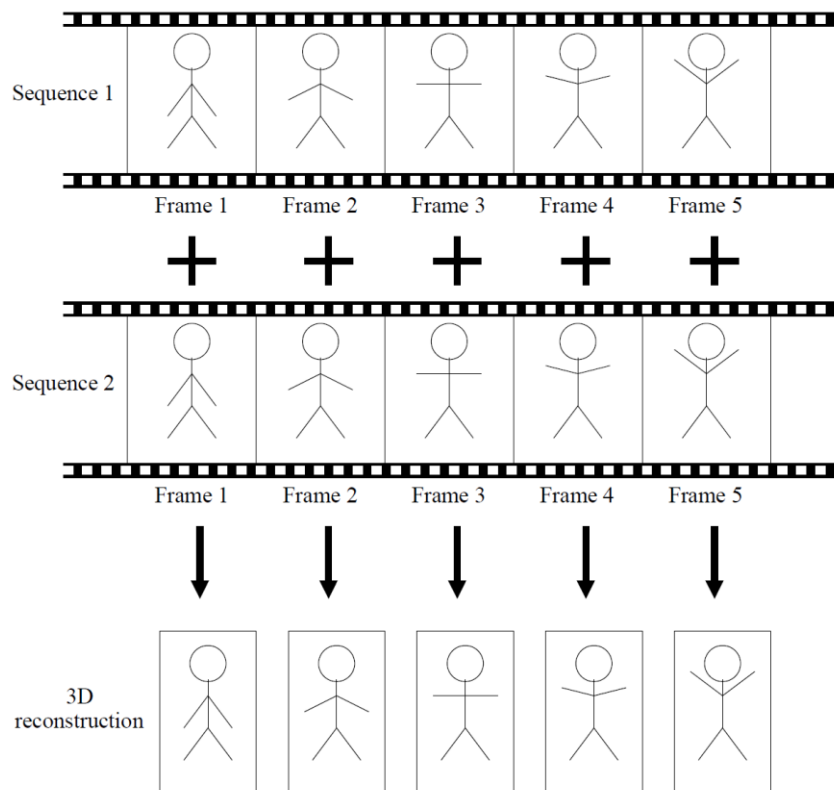


левый ракурс

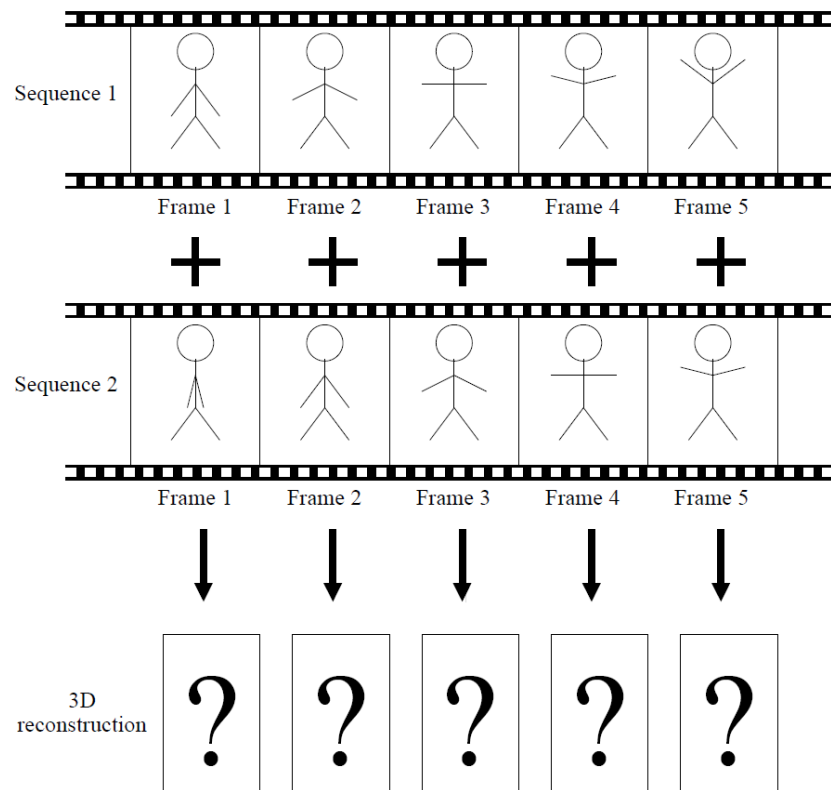


правый ракурс

Временная синхронизация



Синхронизированные ракурсы



Несинхронизированные ракурсы

Временная синхронизация

Актуальность

- Запустить две камеры одновременно довольно сложно
- В процессе записи камеры могут пропускать кадры
- Потери данных при передаче



Содержание

- Введение
- Цветокоррекция
 - **Постановка задачи**
 - Цветокоррекция с использованием размытого изображения
 - Hybrid color correction
- Исправление геометрии
- Временная синхронизация
- Заключение



Цветокоррекция

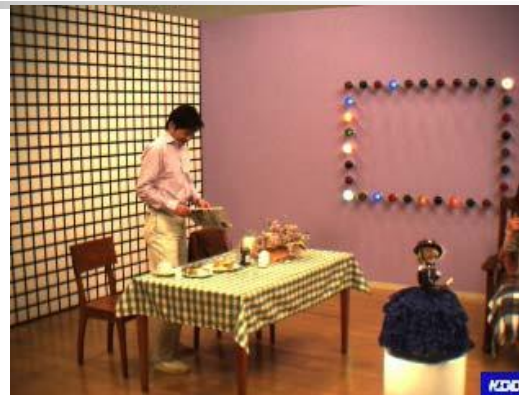
Постановка задачи:

- Найти преобразование, приводящее один ракурс к другому, чтобы объекты в соответствующих ракурсах имели один и тот же цвет
- Найти среднее преобразование для двух ракурсов

Цветокоррекция



reference image



input image



результат применения одного преобразования по всем пикселям



результат с учётом пространственной принадлежности пикселей



Содержание

- Введение
- Цветокоррекция
 - Постановка задачи
 - **Цветокоррекция с использованием размытого изображения**
 - Hybrid color correction
- Исправление геометрии
- Временная синхронизация
- Заключение

Gaussian Blur

Алгоритм

1. Поиск пар feature points с помощью SIFT (Scale-Invariant Feature Transform)
2. Гауссово размытие левого и правого изображений
3. Выделение цветов feature points на размытых и неразмытых изображениях
4. Построение и применение функции преобразования цветов

Gaussian Blur

Функция преобразования

$$M(c'_{i,j} = i) = c_{i,j}$$

$$D_{i,j,u} = -\frac{1}{2\pi\sigma^2} \exp\left(-\frac{(c_{i,j} - q'_{u,j})^2 + (c_{i,j} - q''_{u,j})^2}{2\sigma}\right)$$

$$E_{i,j} = \frac{1}{K} \sum_{u=0}^{K-1} D_{i,j,u}, \quad R_{i,j} = \lambda \left| \frac{(c_{i+1,j} + c_{i-1,j})}{2} - c_{i,j} \right|$$

$$F_{i,j} = R_{i,j} + E_{i,j}$$

$c'_{i,j}$, $c_{i,j}$ – старое и новое значение интенсивности j -ой цветовой компоненты

$q'_{u,j}$, $q''_{u,j}$ – значения интенсивности j -ой цветовой компоненты u -ой feature point для левого и правого ракурса соответственно

K – количество обрабатываемых feature points на размытых и неразмытых изображениях

λ , σ – коэффициенты сглаживания

M – искомая функция

$F_{i,j}$ – функция, для которой выполняется минимизация

Gaussian Blur

Результат

Входные данные



reference image



input image

Gaussian Blur

Результат

Выходные данные



reference image



output image



Выводы

Достоинства:

- Исправление цвета на зашумлённых изображениях
- Возможность эффективной параллельной реализации

Недостаток:

- Не учитывается разная освещённость отдельных объектов



Содержание

- Введение
- Цветокоррекция
 - Постановка задачи
 - Цветокоррекция с использованием размытого изображения
 - **Hybrid color correction**
- Исправление геометрии
- Временная синхронизация
- Заключение

Алгоритм

1. Поиск, сопоставление и отслеживание feature points у двух ракурсов
 - Scale-Invariant Feature Transform (David Lowe, 1999)
 - Pyramid Optical Flow (Jean-Yves Bouguet, 2002)
2. Оценка преобразования
 1. Histogram matching
 2. Локальная цветокоррекция
3. Применение итогового преобразования к входному изображению

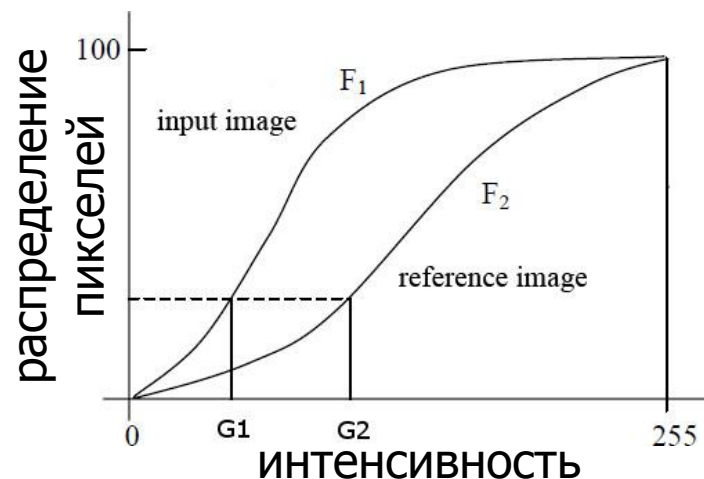
Histogram matching

Для каждой из компонент RGB:

1. Строим интегральные гистограммы
2. Для каждого пикселя с интенсивностью G_2 находим такое G_1 , что $F_1(G_1) = F_2(G_2)$.

Получаем функцию $M(G_1) = G_2$

3. Функцию M применяем к каждому пикселю изображения



Histogram matching (2)

$$Lum_{r,i,j} = \frac{1}{3} * (R_{r,i,j} + G_{r,i,j} + B_{r,i,j})$$

$$LUT_D[k] = LUT[k] - k$$

$$R'_{r,i,j} = R_{r,i,j} + \left(\frac{R_{r,i,j}}{R_{r,i,j} + G_{r,i,j} + B_{r,i,j}} \right) * LUT_D[Lum_{r,i,j}]$$

LUT – гистограмма яркости для правого ракурса

$R_{r,i,j}$, $G_{r,i,j}$, $B_{r,i,j}$ – цветовые компоненты

$R'_{r,i,j}$ – полученный цвет

Hybrid color correction

Обработка feature points

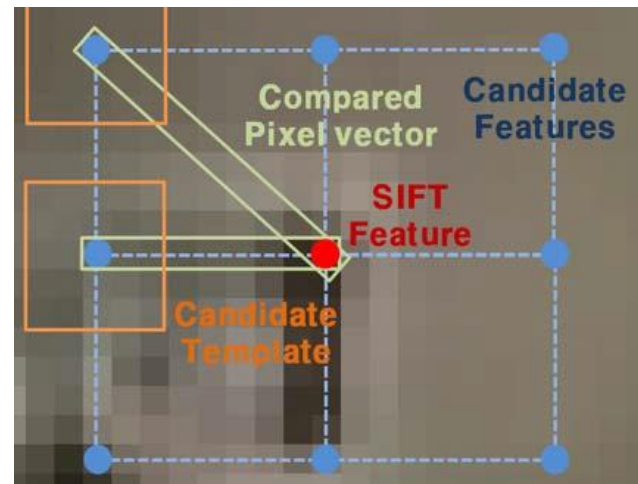
if $CC(CPV_{l,k}, CPV_{r,k}) > w_1$ then

$$m_{l,k} = \frac{1}{h w} \sum_{i=1}^h \sum_{j=1}^w Lum_{l,i,j}$$

$$m_{r,k} = \frac{1}{h w} \sum_{i=1}^h \sum_{j=1}^w Lum_{r,i,j}$$

$$SND_k = \frac{1}{h w} \sum_{i=1}^h \sum_{j=1}^w (|Lum_{l,i,j} - Lum_{r,i,j} - m_{l,k} + m_{r,k}|)$$

if $SND_k < w_2$ then add CT_k to $CFLT$



CC – кросс-корреляция

$Lum_{l(r),i,j}$ – яркость пикселя (i,j)
на левом (правом) ракурсе

SND_k – сумма нормализованных
разностей для k -ого
Candidate Template

$CFLT$ – Candidate Feature
Look-up Table

w_1, w_2 – веса

w, h – размеры Candidate Template

Hybrid color correction

Веса для локальных окрестностей

$$D_k = \frac{1}{hw} \sum_{i=1}^w \sum_{j=1}^h C_{k,r,i,j} - C_{k,l,i,j}$$

$$w_{i,k,s} = e^{-\frac{sd_{i,k}^4}{2\sigma_s^2}}$$

$$w_{i,j,c} = \begin{cases} -\frac{cd_{i,k}}{th_c} + 1, & \text{if } cd_{i,k} \leq th_c \\ 0, & \text{otherwise} \end{cases}$$

$$w_{i,k} = \frac{w_{i,k,s} w_{i,k,c}}{\sum_{p=1}^U w_{i,p,s} w_{i,p,c}}$$

$$w_{i,\max} = \max_k \arg (w_{i,k,s} w_{i,k,c})$$

$sd_{i,k}$ – расстояние от i -го пикселя до k -ой feature point

$cd_{i,k}$ – разность между интенсивностью i -ой точки и интенсивностью k -го элемента *CFLT*

$C_{k,l(r),i,j}$ – значение цветовой компоненты для (i,j) -ой точки k -го элемента *CFLT* левого (правого) ракурса

σ_s, th_c – веса

$w_{i,k}, w_{i,\max}$ – полученные веса

Изменение input image

$$C_{Local,i} = C_i + \sum_{k=1}^U w_{i,k} D_k$$

$$C_{Hybrid,i} = a w_{i,max} C_{Local,i} + (1 - a w_{i,max}) C_{Global,i}$$

C_i – цвет i -го пикселя

$C_{Local,i}$ – цвет, изменённый с помощью локальной коррекции

$C_{Global,i}$ – цвет, изменённый с помощью histogram matching

$C_{Hybrid,i}$ – результирующий цвет

a – весовой коэффициент

Результат

Reference image



Результат

Input image



Результат

Reference image



Результат

Histogram matching



Результат

Reference image



Результат

Hybrid color correction





Выводы

Достоинства:

- Учёт освещённости разных объектов
- Стабильность к зашумлённым feature points

Недостаток:

- Большое время выполнения за счёт использования Optical Flow



Содержание

- Введение
- Цветокоррекция
- Исправление геометрии
 - **Основные понятия**
 - Постановка задачи
 - Ректификация
 - Perception optimization
- Временная синхронизация
- Заключение

Основные понятия

- Однородные координаты — координаты, обладающие свойством, что определяемый ими объект не меняется при умножении всех координат на одно и то же ненулевое число
- Перевод однородных координат в евклидовы:

$$(x, y, z) \rightarrow \left(\frac{x}{z}, \frac{y}{z} \right)$$

Основные понятия

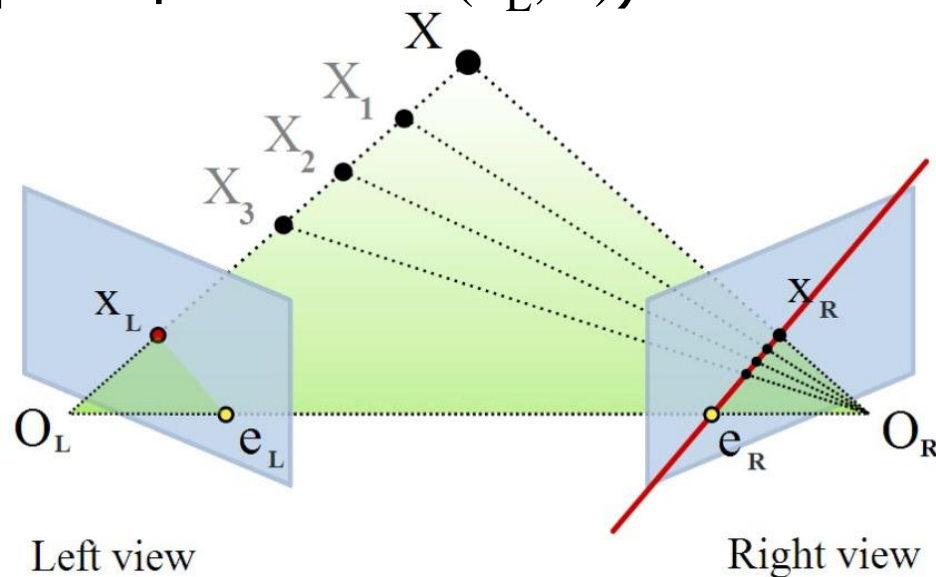
Эпиполярная геометрия

O_L, O_R – фокусы камер

x_L, x_R – проекции точки X

e_L, e_R – эпиполярные точки (epipoles)

(x_R, e_R) – эпиполярная линия
(проекция линии (x_L, X))



Основные понятия

Фундаментальная матрица – матрица 3×3 , связывающая соответствующие точки на двух ракурсах

$$x_r^T F x_l = 0$$

x_l, x_r – соответствующие точки в однородных координатах, выбранные на левом и правом ракурсе соответственно

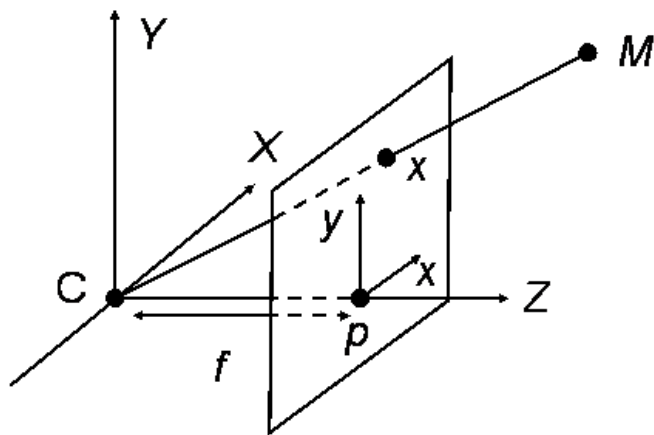
$F x_l$ – определяет эпиполярную линию на правом ракурсе, на которой должна лежать x_r

Эпиполярные точки удовлетворяют:

$$e_R^T F = F e_l = 0$$

Основные понятия

Camera matrix



Camera matrix:

$$\mathbf{K} = \begin{bmatrix} f m_x & s & p_x \\ & f m_y & p_y \\ & & 1 \end{bmatrix}$$

C – фокусная точка камеры

p – главная точка

M – точка, которую будем проецировать

f – фокусное расстояние

x – проекция точки M

p_x, p_y – координаты проекции главной точки на плоскость CXY

m_x, m_y – масштабирующие коэффициенты между CXY и pxy системами координат

s – косинус угла между осями p_x и p_y

Основные понятия

Camera matrix (2)

Общий вид (идеальная матрица):

$$C = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & \frac{1}{f} & 0 \end{pmatrix} \sim \begin{pmatrix} f & 0 & 0 & 0 \\ 0 & f & 0 & 0 \\ 0 & 0 & 1 & 0 \end{pmatrix}$$

f – фокусное расстояние камеры

Проекция 3D изображения на плоскость
камеры с использованием матрицы камеры:

$$\begin{pmatrix} x' \\ y' \\ 1 \end{pmatrix} \sim C \begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix}$$



Содержание

- Введение
- Цветокоррекция
- Исправление геометрии
 - Основные понятия
 - **Постановка задачи**
 - Ректификация
 - Perception optimization
- Временная синхронизация
- Заключение



Ректификация

Постановка задачи:

Найти такие проективные преобразования для двух ракурсов, что после их применения будут отсутствовать:

- вертикальный параллакс
- поворот изображений друг относительно друга
- искажения вертикальной перспективы

Ректификация

Пример

Входные данные



левый ракурс

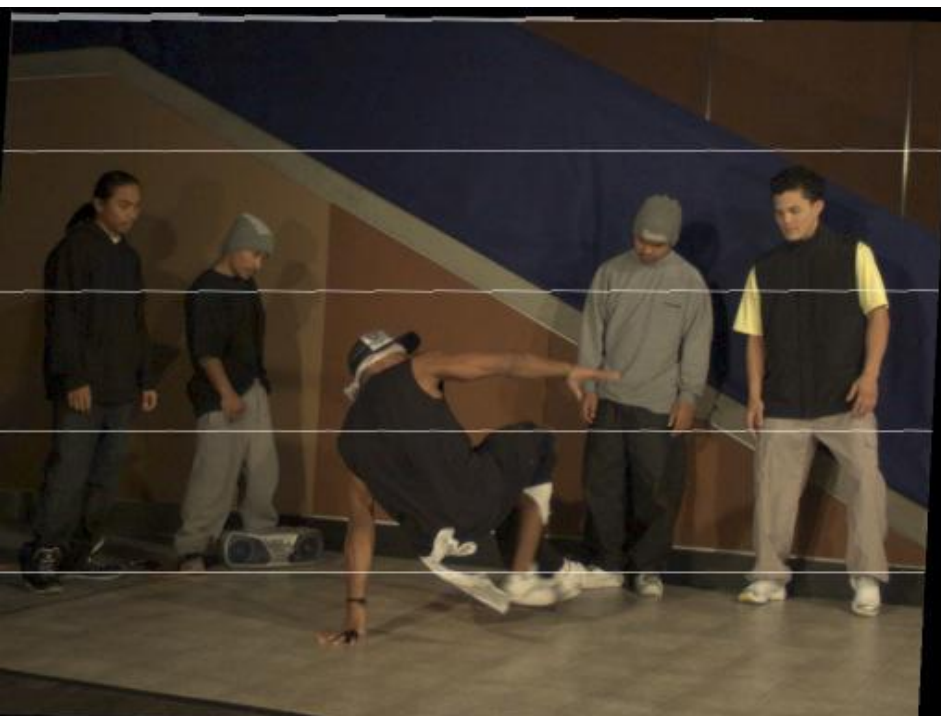


правый ракурс

Ректификация

Пример (2)

Выходные данные



левый ракурс



правый ракурс

Ректификация

Пример (3)

Сравнение



ВХОДНЫЕ ДАННЫЕ



ВЫХОДНЫЕ ДАННЫЕ



Содержание

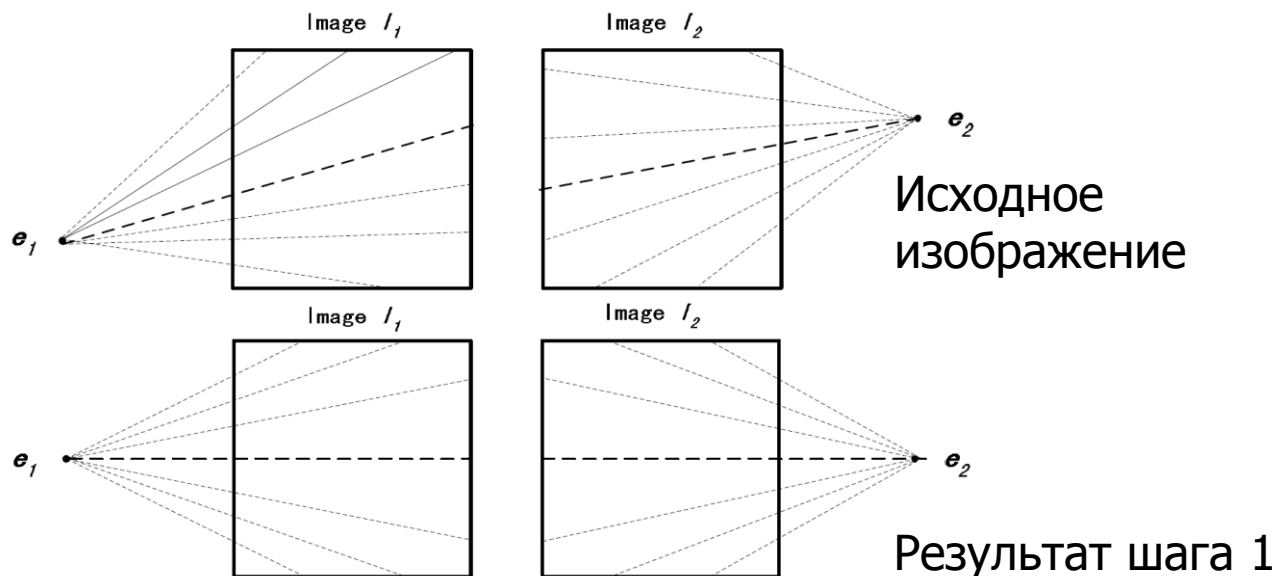
- Введение
- Цветокоррекция
- Исправление геометрии
 - Основные понятия
 - Постановка задачи
 - **Ректификация**
 - Perception optimization
- Временная синхронизация
- Заключение

Ректификация

Шаг 1

Цель: отобразить эпиполярные точки на бесконечность

1. Обнуление y -координаты у эпиполярных точек

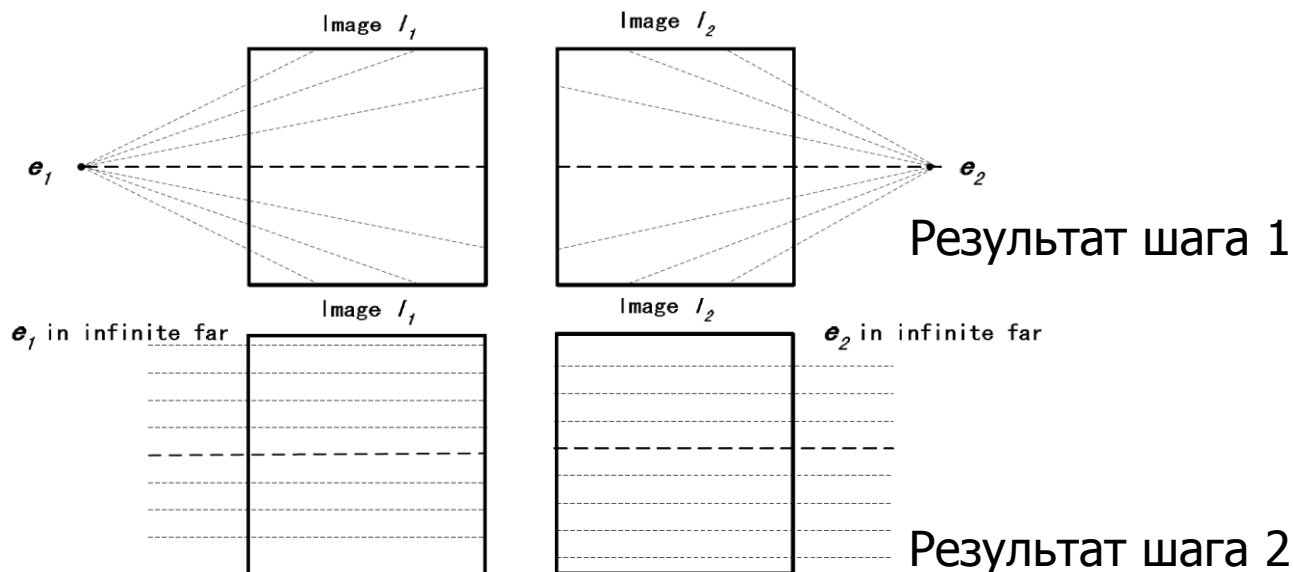


Ректификация

Шаг 2

Цель: отобразить эпилярные точки на бесконечность

2. Отображение x -координаты у эпилярных точек на левом и правом ракурсе $-\infty$ и $+\infty$ соответственно

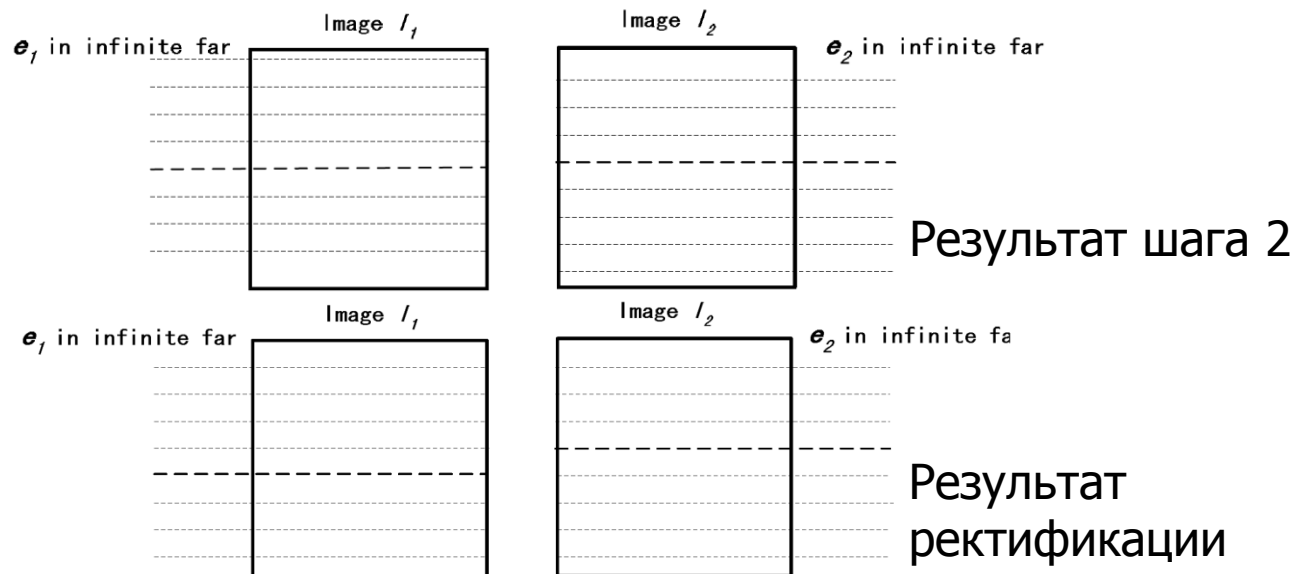


Ректификация

Шаг 3

Цель: отобразить эпиполярные точки на бесконечность

- Вертикальное сопоставление полученных параллельных эпиполярных линий (устранение вертикального параллакса)





Содержание

- Введение
- Цветокоррекция
- Исправление геометрии
 - Основные понятия
 - Постановка задачи
 - Ректификация
 - **Perception optimization**
- Временная синхронизация
- Заключение

Расположение камер

Идеальный случай

$$P_1 = K[I | 0]$$

$$P_2 = K[I | t] \quad t = (k, 0, 0)^T$$

P_1, P_2 – матрицы камер C1, C2
соответственно

K – внутренняя матрица камер

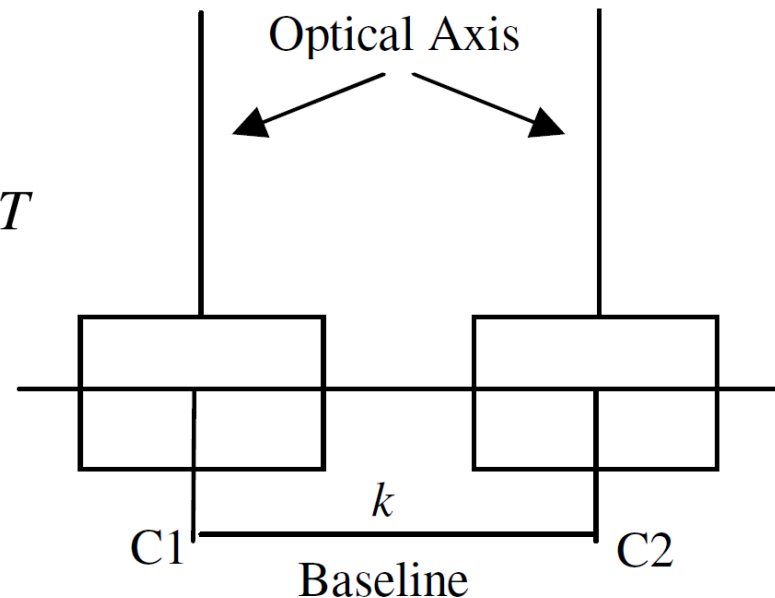


Схема расположения камер
в идеальном случае

Расположение камер

Общий случай

$$\begin{cases} P_1 = K_1 R_1 [I \mid -C_1] = M_1 [I \mid -C_1] \\ P_2 = K_2 R_2 [I \mid -C_2] = M_2 [I \mid -C_2] \end{cases}$$

R_1, R_2 – матрицы поворота камер

K_1, K_2 – внутренние матрицы камер

C_1, C_2 – координаты центров камер

$$R_1 = \begin{bmatrix} \cos \alpha & \sin \alpha & 0 \\ -\sin \alpha & \cos \alpha & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

$$R_2 = \begin{bmatrix} \cos \beta & \sin \beta & 0 \\ -\sin \beta & \cos \beta & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

Условия исправленного стерео:

$$\begin{cases} K_1 = K_2 = K \\ R_1 = R_2 = R \\ R(C_2 - C_1) = (k, 0, 0)^T \end{cases}$$

← условие отсчитывания углов α, β
от оси baseline, направленной
от камеры C1 к C2

Perception optimization

Предположение

Предположение о внутренней матрице камеры:

$$K = K_2 = \begin{bmatrix} f & 0 & p_x \\ 0 & -f & p_y \\ 0 & 0 & 1 \end{bmatrix} \quad \text{with} \quad \begin{cases} p_x = w/2 \\ p_y = h/2 \\ f = (w + h)/2 \end{cases}$$

f – фокусное расстояние камер
 w, h – размеры изображения

Perception optimization

Параметры преобразования

Предположение:

Точки на одинаковой глубине имеют одинаковый диспаратет

Алгоритм поиска наборов соответствующих точек с одинаковой глубиной:

1. Просмотреть все найденные соответствующие точки
2. Сгруппировать точки, имеющие одинаковый диспаратет

Perception optimization

Параметры преобразования (2)

Утверждение:

Для точек с одинаковой глубиной (X_1, X_2) , с проекциями (x_1, x_2) и (x_1', x_2') соответственно, выполняется:

$$x_1 - x_2 = x_1' - x_2'$$

Преобразование левого ракурса из минимизации функции:

$$\sum_p \sum_{i,j \in A_p} \left(H_2 M_0 (a(x_i - x_j) + b(y_i - y_j)) - H_2 (x'_i - x'_j) \right)^2$$

$$c = H_2 x'_0 - H_2 M_0 (ax_0 - bx_0)$$

M_0 находится из уравнения: $F = e_2 M_0$

H_1, H_2 – матрицы проективных преобразований для левого и правого ракурса соответственно

$(x_i, y_i), (x_j, y_j)$ – точки, имеющие одинаковую глубину на левом ракурсе

$(x'_i, y'_i), (x'_j, y'_j)$ – соответствующие им точки на правом ракурсе

Perception optimization

Параметры преобразования (3)

Проективные преобразования принимают вид:

$$H_1 = H_A H_2 M_0$$

$$H_2 = K R K_2^{-1}$$

$$R = (r, s, t)^T$$

$$r = \frac{(K_2^{-1} e_2)}{\|K_2^{-1} e_2\|}, \quad t = r \times (0, 1, 0), \quad s = r \times t$$

Параметры для преобразования левого ракурса:

$$H_A = \begin{bmatrix} a & b & c \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

a, b, c – параметры, требующие определения

e_2 – эпиполярная точка правого ракурса

R – матрица поворота

Perception optimization

Результат

Входные данные



левый ракурс



правый ракурс

Perception optimization

Результат

Выходные данные



левый ракурс



правый ракурс

Perception optimization

Результат



ВХОДНЫЕ ДАННЫЕ



ВЫХОДНЫЕ ДАННЫЕ



Выводы

Достоинство:

- Результат выглядит как отснятый с помощью стандартной установки камер

Недостаток:

- Высокая вычислительная сложность



Содержание

- Введение
- Цветокоррекция
- Исправление геометрии
- **Временная синхронизация**
 - Direct alignment algorithm
 - Feature-based algorithm
- Заключение



Временная синхронизация

Постановка задачи:

Определить величину временного сдвига в количестве кадров между видеопоследовательностями, снятыми с разных ракурсов



Содержание

- Введение
- Цветокоррекция
- Исправление геометрии
- Временная синхронизация
 - **Direct alignment algorithm**
 - Feature-based algorithm
- Заключение

Direct alignment algorithm

Обозначения

$$\vec{\mathbf{x}} = (x, y, t)$$

$$\vec{\mathbf{x}}' = (x', y', t') = (x + u, y + v, t + w)$$

$$\vec{\mathbf{u}} = (u, v, w)$$

$\bar{\mathbf{x}}$ – точка в эталонной последовательности

$\bar{\mathbf{x}}'$ – соответствующая ей точка во входной последовательности

$\bar{\mathbf{u}}'$ – пространственно-временной сдвиг

Direct alignment algorithm

Пространственное преобразование:

$$H = \begin{bmatrix} H_1 \\ H_2 \\ H_3 \end{bmatrix} = \begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{bmatrix}$$
$$x' = \frac{H_1 \vec{p}}{H_3 \vec{p}}, y' = \frac{H_2 \vec{p}}{H_3 \vec{p}}, \vec{p} = (x, y, 1)$$

H – матрица проективного преобразования

$(x, y), (x', y')$ – сопоставленные пространственные
координаты в reference и input

последовательностях соответственно

Direct alignment algorithm

Временное преобразование:

$$t' = s \cdot t + \Delta t$$

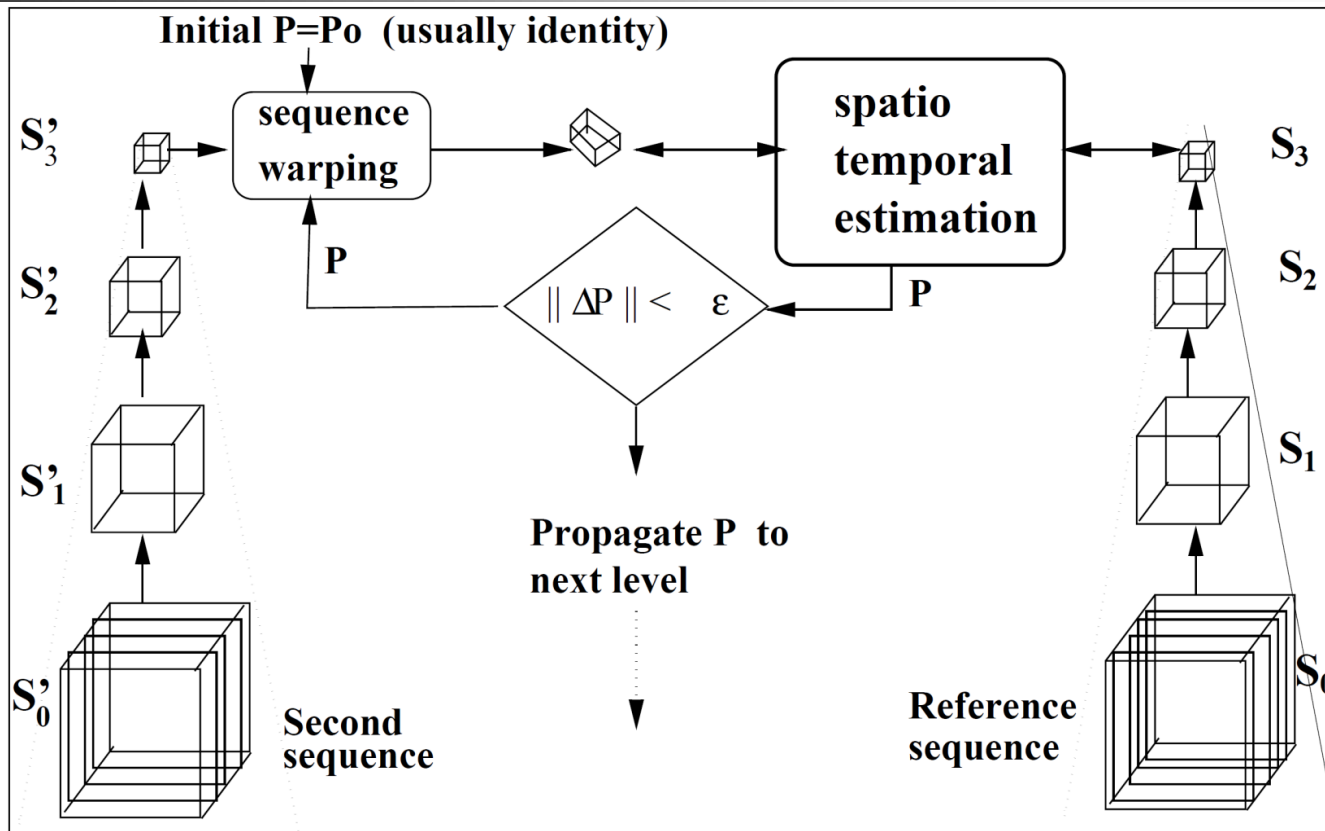
t, t' – сопоставленные моменты времени

s – отношение частот кадров двух камер, обычно известно

Δt – временной сдвиг между видеопоследовательностями

Direct alignment algorithm

Схема работы алгоритма



P – приближаемое преобразование

P_0 – начальное преобразование (единичная матрица)

Direct alignment algorithm

Определение преобразования

Параметры для определения:

$$\vec{P} = [\vec{P}_{spatial}, \vec{P}_{temporal}] = [h_{11} \ h_{12} \ h_{13} \ h_{21} \ h_{22} \ h_{23} \ h_{31} \ h_{32} \ h_{33} \ \Delta t]$$

SSD error function:

$$ERR(\vec{P}) = \sum_{\vec{x}=x,y,t} (S(\vec{x}) - S'(\vec{x} + \vec{u}(\vec{x}; \vec{P})))^2$$

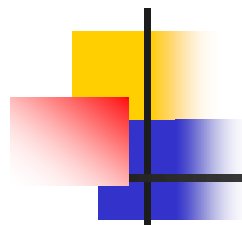
S, S' – яркость точки (x, y, t) в reference и input последовательностях соответственно

\vec{P} – искомое преобразование

Для минимизации функции ошибки $ERR(P)$ применяется алгоритм Гаусса-Ньютона

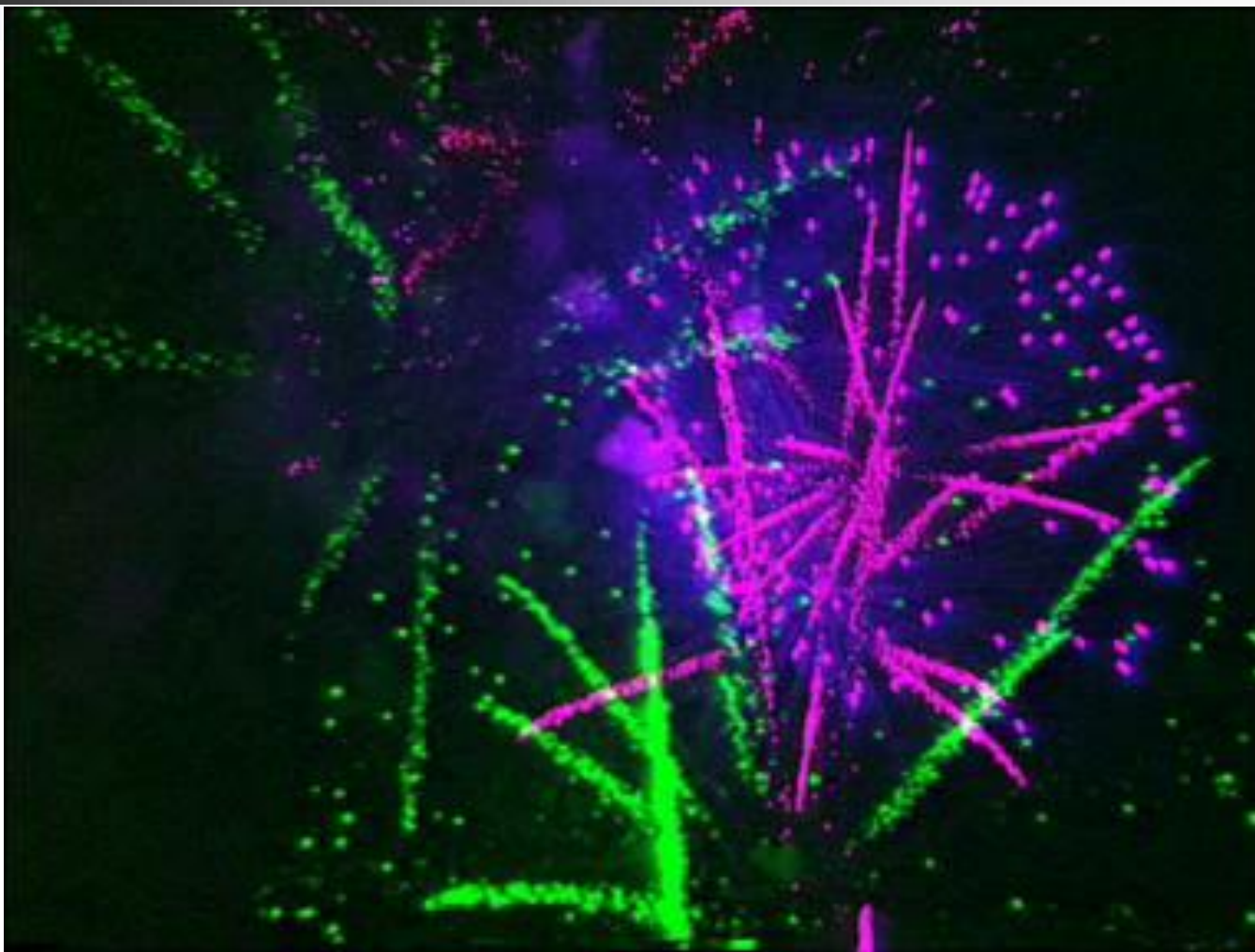
Fireworks sequence

Входные данные



Fireworks sequence

Наложение



Fireworks sequence

Результат





Light sequence

Входные данные



Light sequence

Наложение



Light sequence

Результат



Wind sequence

Входные данные

Wind sequence

Наложение



Wind sequence

Результат





Выводы

Достоинство:

- Высокая точность

Недостаток:

- Объекты должны быть удалены от камер на расстояние на порядок большее расстояния между камерами



Содержание

- Введение
- Цветокоррекция
- Исправление геометрии
- Временная синхронизация
 - Direct alignment algorithm
 - **Feature-based algorithm**
- Заключение



Feature-based algorithm

Алгоритм:

1. Вычисление фундаментальной матрицы, определение эпиполярных линий (с использованием SIFT)
2. Построение следа изображений на эпиполярных линиях
3. Определение временного сдвига по полученному следу

Feature-based algorithm

Пример

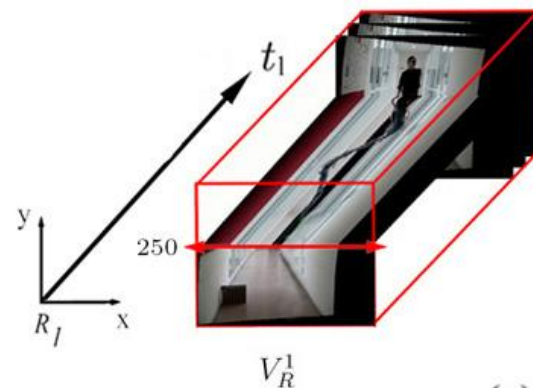
Входные последовательности:



Feature-based algorithm

Алгоритм

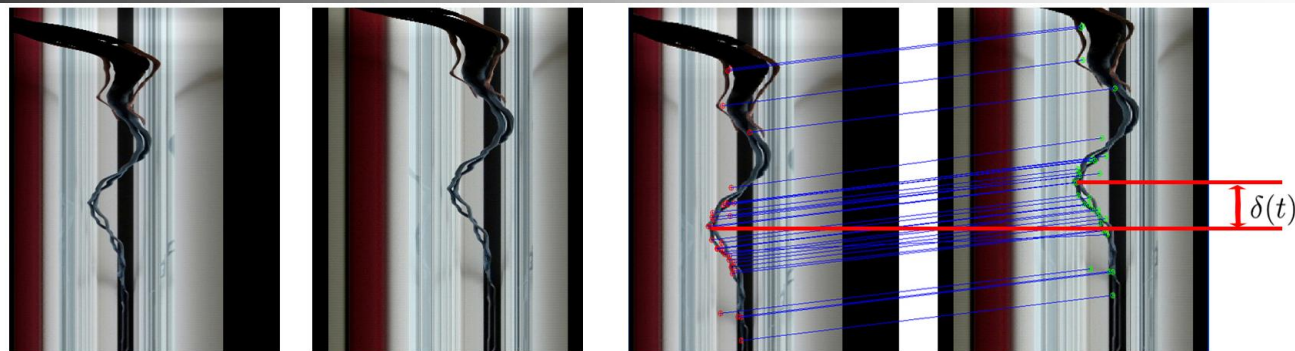
1. Поиск соответствующих точек (SIFT)
2. Восстановление эпиполярной геометрии
3. Выделение последовательности изменения для каждой эпиполярной линии



след эпиполярной линии

Feature-based algorithm

Алгоритм (2)



след эиполярной линии

результат работы SIFT

4. Сопоставление схожих участков следов эиполярных линий с использованием SIFT
5. Определение $\delta(t)$ с помощью медианы
 - как сдвига последовательности для данной эиполярной линии
 - по всем эиполярным линиям



Выводы

Достоинства:

- Интересный алгоритм
- Возможность адаптации к потерянным кадрам

Недостатки:

- Неустойчивость к активному движению по всем эпиполярным линиям
- Запуск камер должен производиться в статичном положении



Содержание

- Введение
- Цветокоррекция
- Исправление геометрии
- Временная синхронизация
- **Заключение**



Заключение

Рассмотрены алгоритмы:

- цветокоррекции, основанные на глобальном и локальном преобразовании
- исправления геометрии, основанные на восстановлении эпиполярной геометрии
- синхронизации стерео видео с использованием эпиполярной геометрии и без неё

Заключение

Дальнейшие планы

Реализовать алгоритмы:

- цветокоррекции
hybrid color correction
- синхронизации по времени
- коррекции геометрии с использованием эпиполярной геометрии с возможностью определения:
 - вертикального параллакса
 - поворота ракурсов
 - искажения вертикальной перспективы

Литература

1. Kenji Yamamoto, Ryutaro Oi, "Color Correction for Multi-view Video Using Energy Minimization of View Networks", International Journal of Automation and Computing, 2008
2. Jung-Jae Yu, Hae-Dong Kim, Ho-Wook Jang, Seung-Woo Nam, "A hybrid color matching between stereo image sequences", 3DTV Conference: The True Vision – Capture, Transmission and Display of 3D Video (3DTV-CON), 2011
3. Jin Zhou, Baoxin Li, "Image Rectification for Stereoscopic Visualization Without 3D Glasses", Lecture Notes in Computer Science, 2006
4. Yaron Caspi, Michal Irani, "Spatio-Temporal Alignment of Sequences", IEEE Transactions on Pattern Analysis and Machine Intelligence, 2002
5. Vincent Guitteny, Ryad Benosman, Christophe Charbuillet, "Synchronizing Video Sequences from Temporal Epipolar Lines Analysis", Lecture Notes in Computer Science, 2008

Литература (2)

6. Feng Shao, Zongju Peng, You Yang, "Color correction for multi-view video based on background segmentation and dominant color extraction", The World Scientific and Engineering Academy and Society (WSEAS) Transactions on Computers, 2008
7. Daniel Wedge, "Video Sequence Synchronization", The University of Western Australia, 2007
8. http://en.wikipedia.org/wiki/Histogram_matching
9. http://en.wikipedia.org/wiki/Homogeneous_coordinates
10. http://en.wikipedia.org/wiki/Epipolar_geometry
11. [http://en.wikipedia.org/wiki/Fundamental_matrix_\(computer_vision\)](http://en.wikipedia.org/wiki/Fundamental_matrix_(computer_vision))
12. Benjamin Stetter, "3D view of stereo laparoscope in the operating room", Munich Technical University, 2005
13. http://en.wikipedia.org/wiki/Camera_matrix
14. Guo-Yu Lin, Xu Chen, Wei-Gong Zhang, "A Robust Epipolar Rectification Method of Stereo Pairs", Proceedings of the 2010 International Conference on Measuring Technology and Mechatronics Automation, 2010

Лаборатория компьютерной графики и мультимедиа



Видеогруппа — это:

- Выпускники в аспирантурах Англии, Франции, Швейцарии (в России в МГУ и ИПМ им. Келдыша)
- Выпускниками защищено 5 диссертаций
- Наиболее популярные в мире сравнения видеокодеков
- Более 3 миллионов скачанных фильтров обработки видео