Современные методы Optical Flow с GPU реализацией

Максим Харенко

Video Group
CS MSU Graphics & Media Lab

Содержание

- **Введение**
- Локальный метод
- Вариационный метод
- Фазовый метод
- Заключение

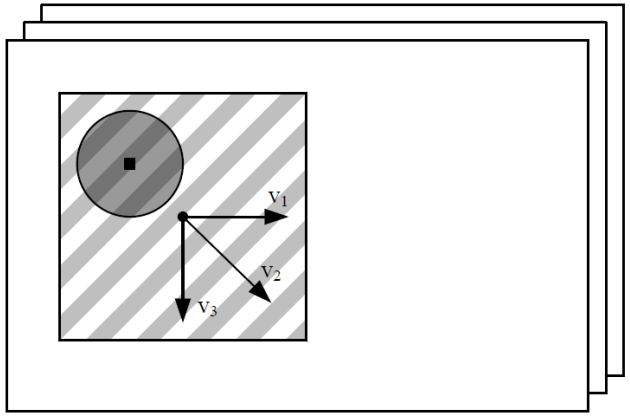
Введение

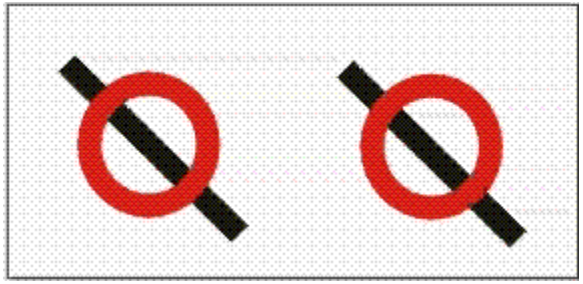
Optical Flow — векторное поле, определяющее скорость движения каждой точки изображения

- Первые публикации появились в 1980 г.
- Возможные применения:
  - Построение карт глубины
  - Сегментация
  - Задача распознавания
  - Точная компенсация движения

Введение

- Проекция движения объектов в 3D сцене на 2D плоскость — real motion
- Real motion сложно оценить из-за отличий от apparent motion:
  - Occlusion problem
  - Wrong motion problem
  - Aperture problem
  - Texture problem





aperture problem

Постановка задачи

Даны два последовательных изображения:

$I_1(\mathbf{x})$ — первый кадр $\quad \mathbf{x} = (x, y)^\top \in \Omega$

$I_2(\mathbf{x})$ — второй кадр $\quad \Omega \subset \mathbb{R}^2$ — область изображения

Необходимо найти векторное поле $\mathbf{u}(\mathbf{x})$, задающее соответствие точек изображений, например:

$$I_1(\mathbf{x} + \mathbf{u}(\mathbf{x})) = I_2(\mathbf{x}), \forall \mathbf{x} \in \Omega$$

$$\mathbf{u}(\mathbf{x}) = (u_1(\mathbf{x}), u_2(\mathbf{x}))^\top : \mathbb{R}^2 \to \mathbb{R}^2$$

Локальные методы

Условие Optical Flow (Gradient constraint equation):

$$\nabla I(\mathbf{x})^\top \mathbf{u} + I_t(\mathbf{x}) = 0 \qquad \nabla I = (I_x, I_y)^\top$$

Локальный метод Lucas & Kanade:

$$E(\mathbf{u}) = \sum_{\mathbf{x}\in\omega} g(\mathbf{x}) \left(\nabla I(\mathbf{x})^\top \mathbf{u} + I_t(\mathbf{x})\right)^2$$

$g(\mathbf{x}) \geq 0, \mathbf{x} \in \omega$ — весовая функция

Вектор смещения ищется в каждой точке независимо

Глобальные методы

Глобальный метод Horn & Schunck:

$$E(\mathbf{v}) = \int_{\Omega} \left[\underbrace{\left(\nabla I \cdot \mathbf{v} + I_t \right)^2}_{\text{Условие optical flow}} + \alpha^2 \underbrace{\left(\|\nabla u\|_2^2 + \|\nabla v\|_2^2 \right)}_{\text{Условие гладкости}} \right] d\mathbf{x}$$

$$\mathbf{v}(\mathbf{x}) = (u(\mathbf{x}), v(\mathbf{x}))$$

α — параметр модели

$$\nabla u = \left(\frac{\partial u}{\partial x}, \frac{\partial u}{\partial y} \right)$$

$$\nabla v = \left(\frac{\partial v}{\partial x}, \frac{\partial v}{\partial y} \right)$$

Сглаживает поле независимо от структуры движения

Вариационные методы

Цель — минимизация энергии:

$$E(\mathbf{u}) = E_{D_1}(\mathbf{u}) + \alpha E_{D_2}(\mathbf{u}) + \beta E_S(\mathbf{u})$$

$$\alpha, \beta \geq 0$$

$$E_{D_1}(\mathbf{u}) = \int_\Omega \rho\left(\left|I(\mathbf{x}+\mathbf{u}) - I(\mathbf{x})\right|^2\right) d\mathbf{x}$$

$$\rho(s^2) = \sqrt{s^2 + \sigma^2}$$

$$E_{D_2}(\mathbf{u}) = \int_\Omega \rho\left(\left|\nabla I(\mathbf{x}+\mathbf{u}) - \nabla I(\mathbf{x})\right|^2\right) d\mathbf{x}$$

$$E_S(\mathbf{u}) = \int_\Omega \rho\left(\left|\nabla u\right|^2 + \left|\nabla v\right|^2\right) d\mathbf{x}$$

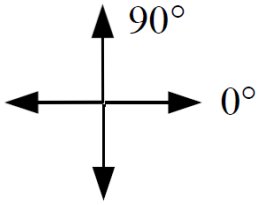
σ – регуляризационный параметр

Фазовые методы

Основная идея — применить предварительную фильтрацию изображений

$$(I \cdot F)(\mathbf{x}, t) = (I \cdot F)(\mathbf{x} + \mathbf{v}\delta t, t + \delta t)$$

Фильтрация проводится в различных направлениях:



$R(\mathbf{x})$ — отфильтрованное изображение

$$R(\mathbf{x}) = (I \cdot \mathrm{F})(\mathbf{x}) = \rho(\mathbf{x}) e^{i\phi(\mathbf{x})}$$

$\rho(\mathbf{x}) = \sqrt{\mathrm{Re}(R(\mathbf{x}))^2 + \mathrm{Im}(R(\mathbf{x}))^2}$ — амплитуда $\quad \phi(\mathbf{x}) = \arctan\left(\frac{\mathrm{Im}(R(\mathbf{x}))}{\mathrm{Re}(R(\mathbf{x}))}\right)$ — фаза

Phase gradient constraint:

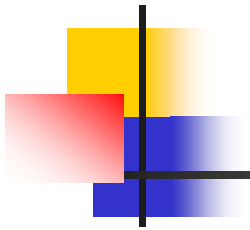
$$\nabla\phi \cdot \mathbf{v} + \psi = 0 \qquad \mathbf{v} = (u, v)$$

$$\nabla\phi = \left(\frac{\partial\phi}{\partial x}, \frac{\partial\phi}{\partial y}\right)^{\mathrm{T}} \quad \psi = \frac{\partial\phi}{\partial t}$$

$\mathbf{v}_\perp$ — проекция $\mathbf{v}$ на $\nabla\phi$

$$\mathbf{v}_\perp = \frac{-\psi}{|\nabla\phi|}\frac{\nabla\phi}{|\nabla\phi|}$$

Realtime phase-based optical flow on the GPU. K. Pauwels, M. M. Van Hulle. IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops, 2008



Содержание

- Введение
- **Локальный метод**
- Вариационный метод
- Фазовый метод
- Заключение

Локальный метод
Алгоритм

Используется модель Lucas & Kanade:

$$I(\mathbf{x}+\omega,t+1)-I(\mathbf{x},t)=0\,,\ \ (\nabla I)^T\omega+I_t=0 \qquad (\nabla I)=(I_xI_y)^T$$

- Видимое смещение $\omega=(u,v)^T$ ищется в виде:

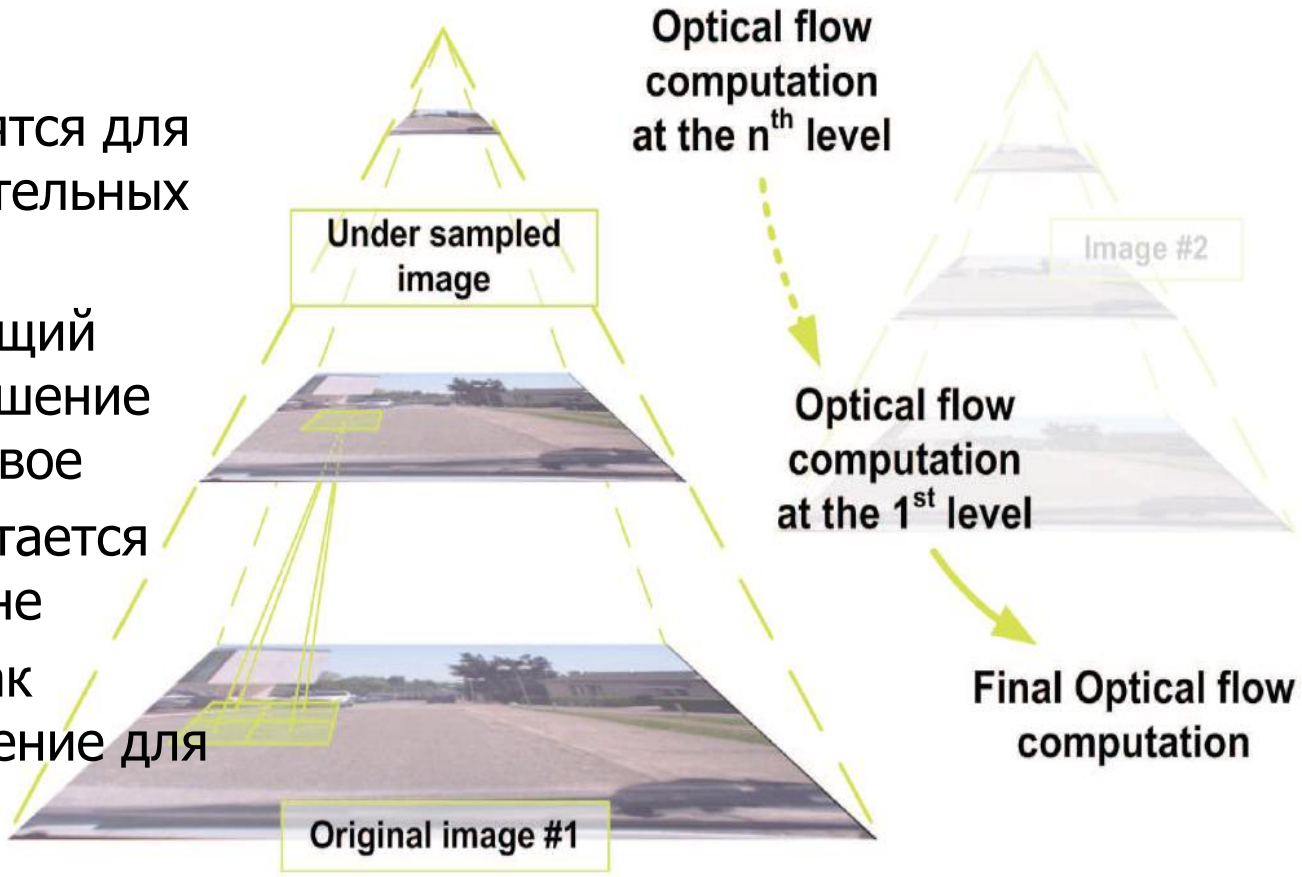
$$\begin{bmatrix} u \\ v \end{bmatrix}=(A^TA)^{-1}A^Tb\,,\quad A=\begin{bmatrix} I_{x1} & I_{y1} \\ I_{x2} & I_{y2} \\ \vdots & \vdots \\ I_{xn} & I_{yn} \end{bmatrix},\ b=\begin{bmatrix} I_{t1} \\ I_{t2} \\ \vdots \\ I_{tn} \end{bmatrix}$$

- Полученная система решается с помощью метода наименьших квадратов с L2-нормой:

$$\begin{bmatrix} u \\ v \end{bmatrix}=(A^TA+\alpha I)^{-1}A^Tb \qquad 0<\alpha<10^{-3}$$

Локальный метод
Пирамидальный алгоритм

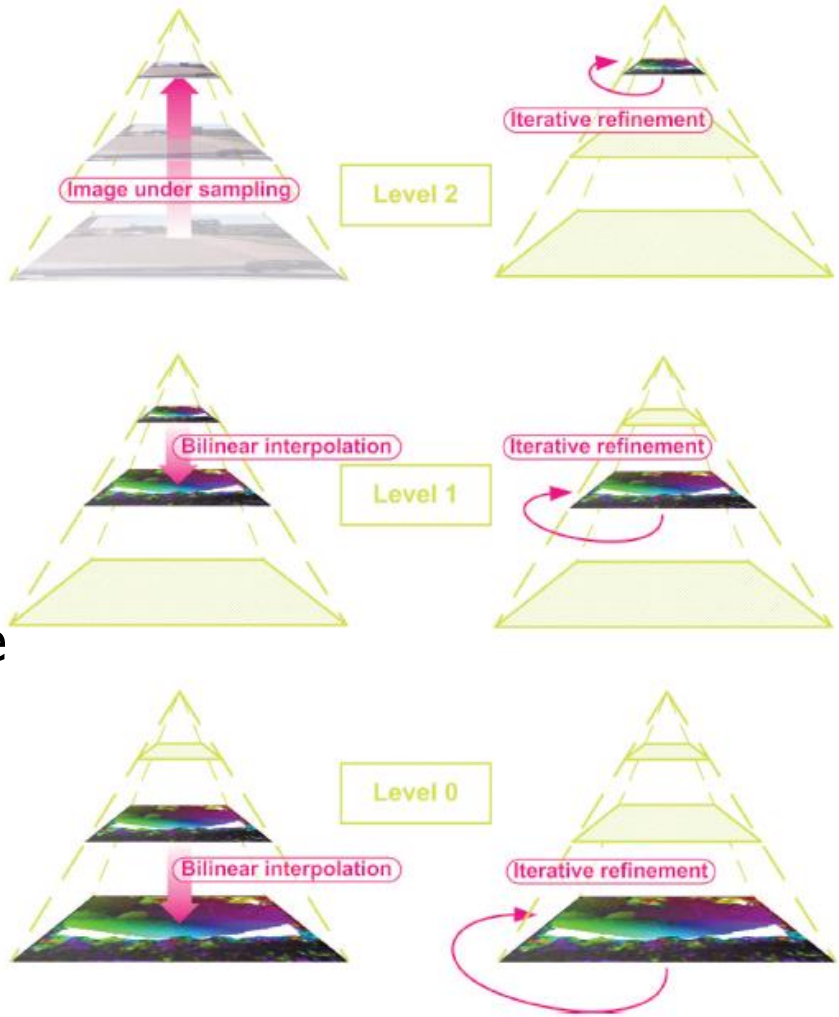
- Пирамиды строятся для двух последовательных кадров
- Каждый следующий уровень - разрешение уменьшается вдвое
- Optical Flow считается на вернем уровне
- Используется как начальное значение для новой оценки
- Билинейная интерполяция



Julien Marzat, Yann Dumortier, Andre Ducrot, "Real-Time Dense and Accurate Parallel Optical Flow using CUDA", WSCG, 2009

Локальный метод
Итеративное и временное улучшение

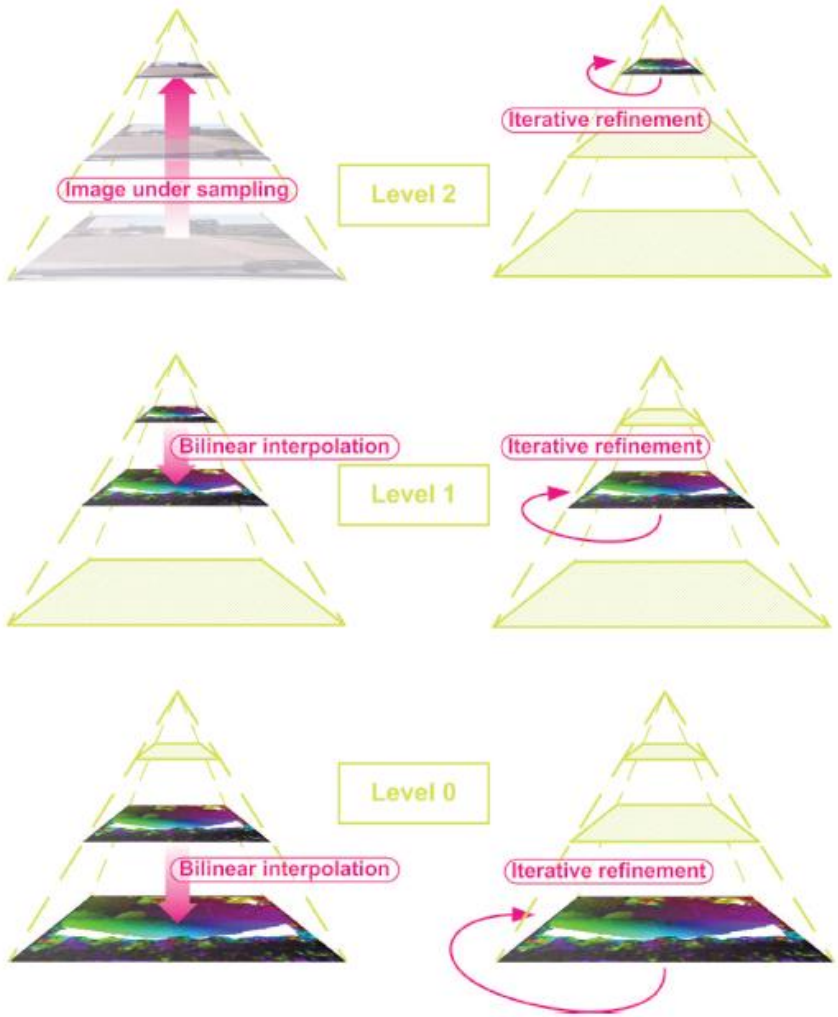
- На каждом уровне пирамиды — итеративное улучшение
- Текущий кадр трансформируется с помощью последнего посчитанного потока: каждая точка изображения сдвигается на соответствующее смещение посчитанное ранее



Julien Marzat, Yann Dumortier, Andre Ducrot, "Real-Time Dense and Accurate Parallel Optical Flow using CUDA", WSCG, 2009

Локальный метод
Итеративное и временное улучшение

- Если смещение не целое, используется билинейная интерполяция
- Временное улучшение заключается в использовании посчитанных между N-1 и N кадрами полей скоростей, как начальное значение для расчета оптического потока между N и N+1 кадрами



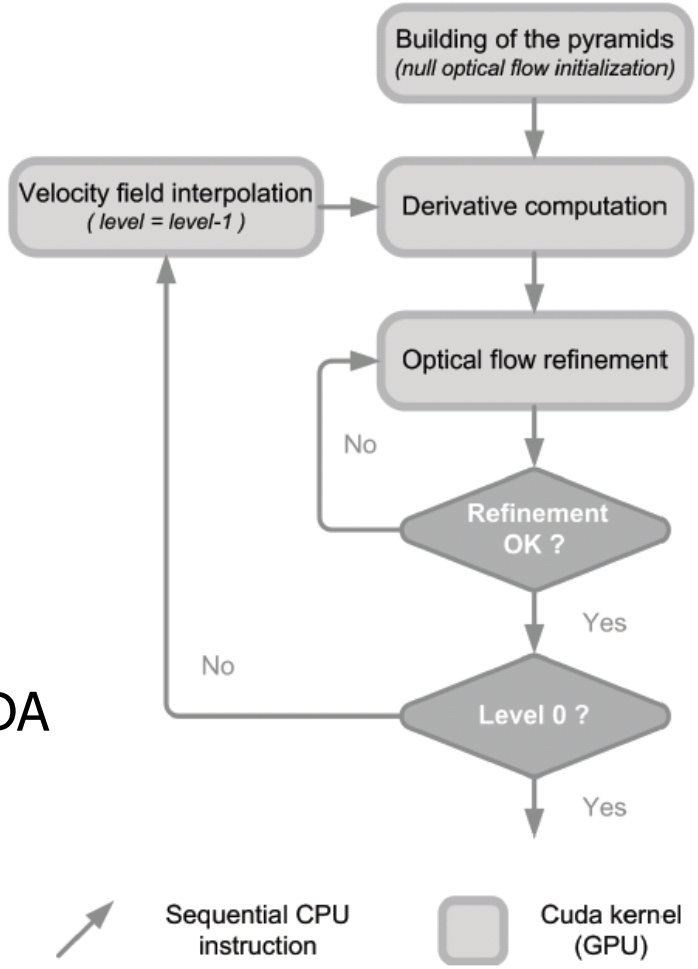
Julien Marzat, Yann Dumortier, Andre Ducrot, "Real-Time Dense and Accurate Parallel Optical Flow using CUDA", WSCG, 2009

Локальный метод
Algorithm parallelization

Расчеты optical flow для одного пикселя происходят независимо друг от друга. Вычисляются одновременно

- Построение пирамид, интерполяция и расчет производных – не зависят от соседних пикселей
- Вычисление смещений точки выполняется, используя только информацию из производной

На пиксель используется один CUDA thread

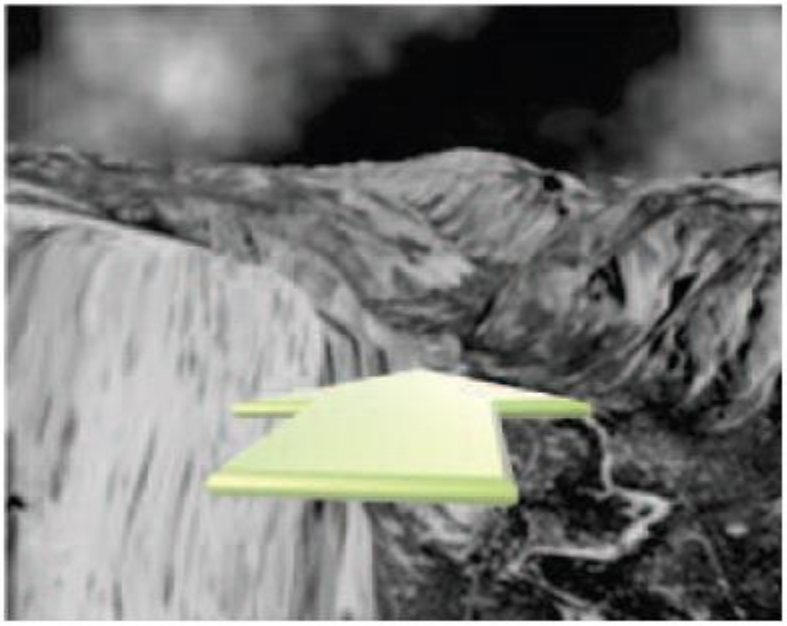


Julien Marzat, Yann Dumortier, Andre Ducrot, "Real-Time Dense and Accurate Parallel Optical Flow using CUDA", WSCG, 2009

Локальный метод
Результаты работы

- Время исполнения CUDA реализации на последовательности 316×252 Yosemite 21 ms per frame (47 fps)
- На реальной последовательности 640×480 время исполнения на оптимизированной последовательной реализации на C составило около 7 секунд на кадр
- На реальной последовательности 640×480 время исполнения CUDA реализации — 67 ms per frame (15 fps)

Локальный метод
Результаты(1)

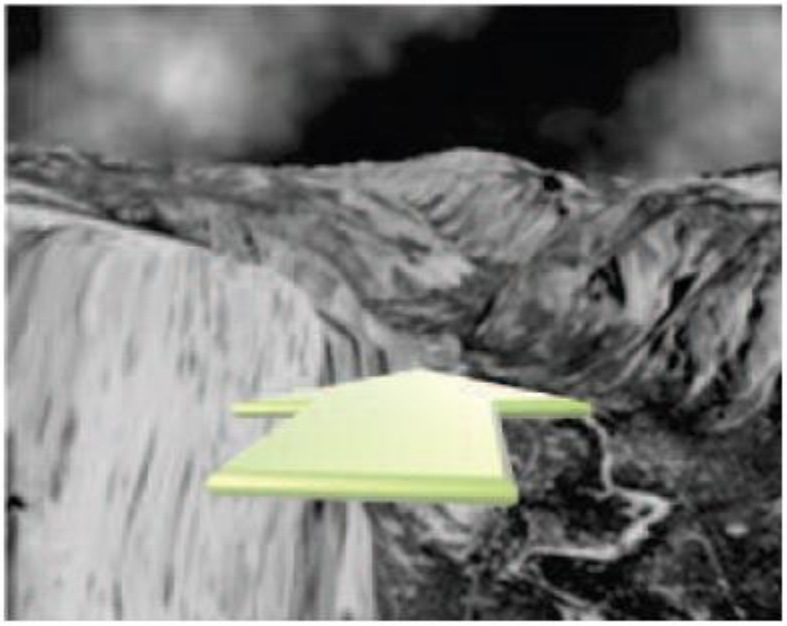


(a) Synthetic sequence

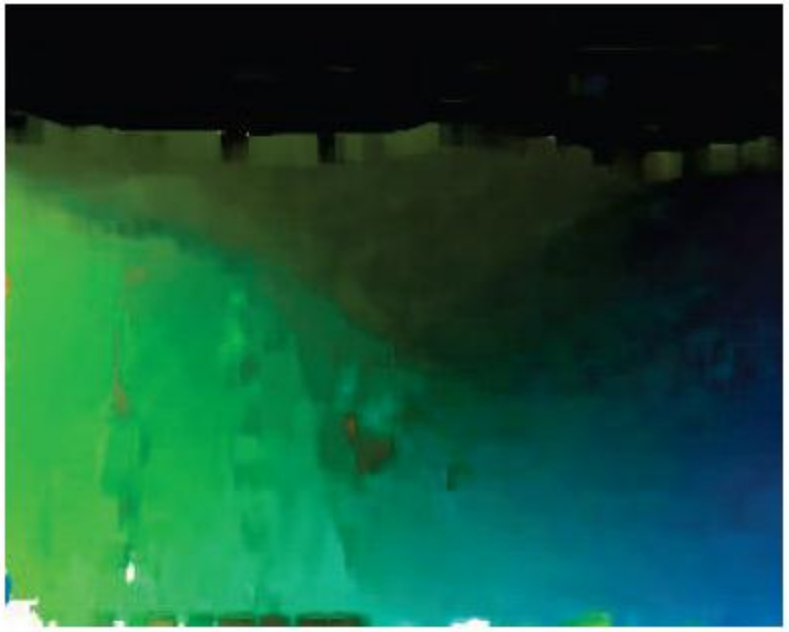
(b) Ground truth

Julien Marzat, Yann Dumortier, Andre Ducrot, "Real-Time Dense and Accurate Parallel Optical Flow using CUDA", WSCG, 2009

Локальный метод
Результаты(2)



(a) Synthetic sequence

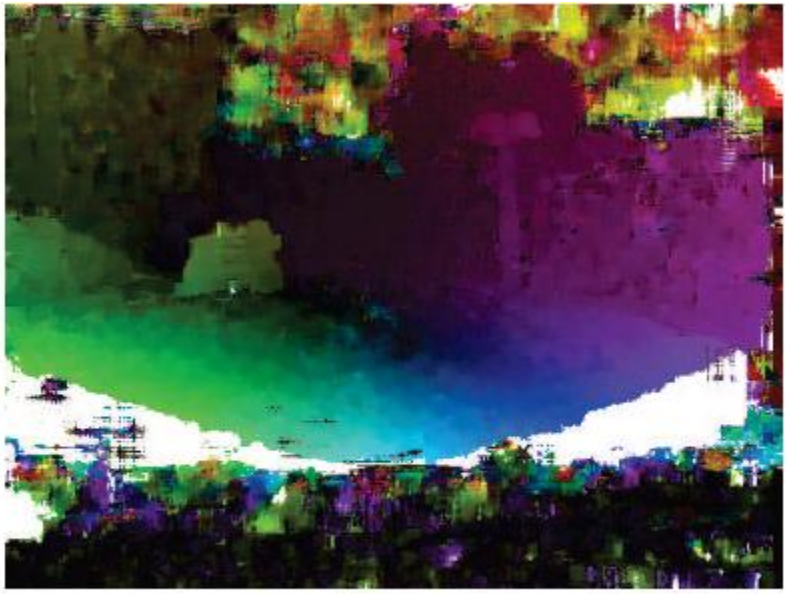


(c) Our result

Julien Marzat, Yann Dumortier, Andre Ducrot, "Real-Time Dense and Accurate Parallel Optical Flow using CUDA", WSCG, 2009

Локальный метод
Результаты(3)

(a) Real sequence

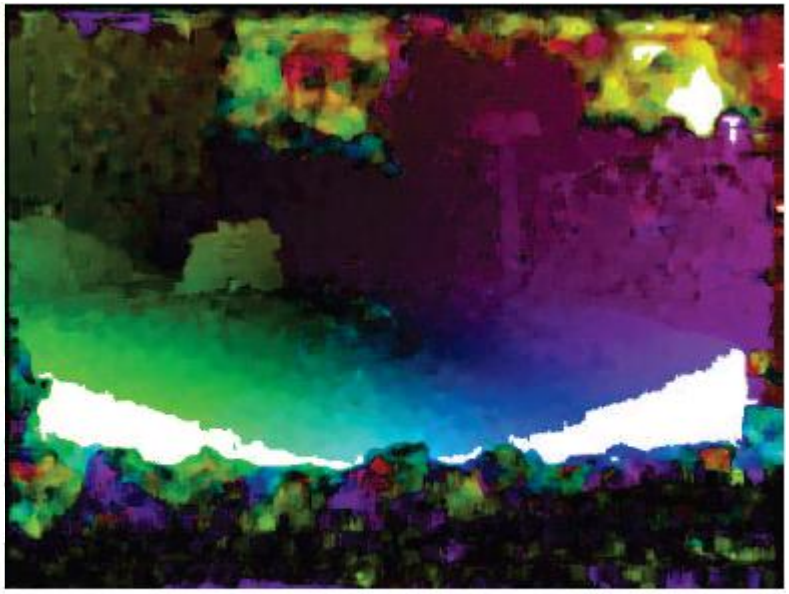


(b) OpenCV (CPU) version

Julien Marzat, Yann Dumortier, Andre Ducrot, "Real-Time Dense and Accurate Parallel Optical Flow using CUDA", WSCG, 2009

Локальный метод
Результаты(4)

(a) Real sequence



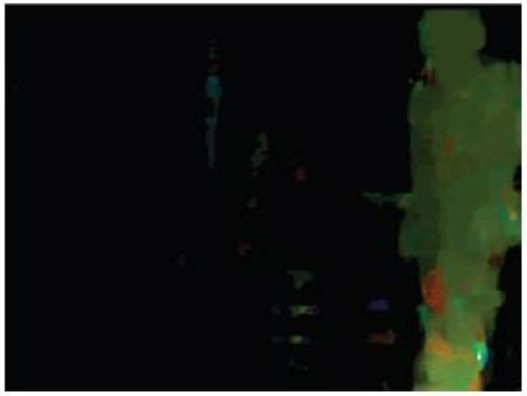
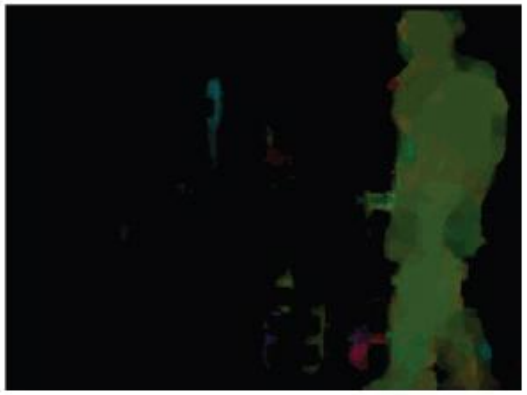
(c) CUDA (GPU) version

Julien Marzat, Yann Dumortier, Andre Ducrot, "Real-Time Dense and Accurate Parallel Optical Flow using CUDA", WSCG, 2009

Локальный метод
Результаты(5)

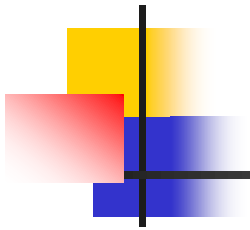




кадр1 кадр2 кадр3

Julien Marzat, Yann Dumortier, Andre Ducrot, "Real-Time Dense and Accurate Parallel Optical Flow using CUDA", WSCG, 2009



Содержание

- Введение
- Локальный метод
- **Вариационный метод**
- Фазовый метод
- Заключение

Вариационный метод
Алгоритм

- Вариационная формулировка optical flow

$$E(v) = \lambda \int_{\mathscr{T}} \|I_1(\boldsymbol{x} + v(\boldsymbol{x})) - I_0(\boldsymbol{x})\| \, \mathrm{d}\boldsymbol{x} + \int_{\mathscr{T}} \|\nabla v(\boldsymbol{x})\| \, \mathrm{d}\boldsymbol{x}$$

- Условие optical flow в области $\boldsymbol{x} + \boldsymbol{v}_0^{\boldsymbol{x}}$ для каждого $\boldsymbol{x}$

$$I_1(\boldsymbol{x} + v(\boldsymbol{x})) - I_0(\boldsymbol{x}) \approx \underbrace{I_1(\boldsymbol{x} + \boldsymbol{v}_0^{\boldsymbol{x}}) + J_{I_1}(\boldsymbol{x} + \boldsymbol{v}_0^{\boldsymbol{x}})(v(\boldsymbol{x}) - \boldsymbol{v}_0^{\boldsymbol{x}}) - I_0(\boldsymbol{x})}_{=\rho(v)(\boldsymbol{x})}$$

$$E_1(u) = \int_{\mathscr{T}} \|\nabla u(\boldsymbol{x})\| \, \mathrm{d}x + \frac{1}{2\theta} \int_{\mathscr{T}} \|v(\boldsymbol{x}) - u(\boldsymbol{x})\|^2 \, \mathrm{d}\boldsymbol{x}$$

$$E_2(v) = \lambda \int_{\mathscr{T}} \|\rho(v)(\boldsymbol{x})\| \, \mathrm{d}x + \frac{1}{2\theta} \int_{\mathscr{T}} \|v(\boldsymbol{x}) - u(\boldsymbol{x})\|^2 \, \mathrm{d}\boldsymbol{x}$$

J_{I_1} — якобиан I_1 ; λ , θ — параметры

Lars Lau Rak, Lars Roholm, Mads Nielsen, Francois Lauze, "TV-L1 Optical Flow for Vector Valued Images", EMMCVPR, 2011

Вариационный метод

Т.к. в $E_2(v)$ не присутствуют производные по v, ее минимизация сводится к точечной минимизации для фиксированных $\boldsymbol{x} + \boldsymbol{v}_0^{\boldsymbol{x}}$ и $\boldsymbol{u}(\boldsymbol{x})$ строго выпуклой функции

$$F(\boldsymbol{v}) = \frac{1}{2}\|\boldsymbol{v} - \boldsymbol{u}_0\|^2 + \lambda\|A\boldsymbol{v} + \boldsymbol{b}\|$$

A – дифференциал *I1* в точке $\boldsymbol{x} + \boldsymbol{v}_0^{\boldsymbol{x}}$

b – вещественное число

Минимизация сводится к подсчету остаточной проекции

Lars Lau Rak, Lars Roholm, Mads Nielsen, Francois Lauze, "TV-L1 Optical Flow for Vector Valued Images", EMMCVPR, 2011

Вариационный метод
Предположение 1

$$F(\boldsymbol{v}) = \frac{1}{2}\|\boldsymbol{v} - \boldsymbol{u}_0\|^2 + \lambda\|A\boldsymbol{v} + \boldsymbol{b}\|$$

В случае, если $\boldsymbol{b} \notin \operatorname{Im} A$, $F(\boldsymbol{v})$ гладкая и может быть минимизирована обычными методами

Иначе $\boldsymbol{b} \in \operatorname{Im} A$, $F(\boldsymbol{v})$, не являясь гладкой, достигает минимума в $\boldsymbol{v} = \boldsymbol{u} - \pi_{\lambda\mathfrak{C}}\left(\boldsymbol{u} + A^{\dagger}\boldsymbol{b}\right)$

Где $\pi_{\lambda\mathfrak{C}}$ – проекция на выпуклое множество $\lambda\mathfrak{C} = \{\lambda\boldsymbol{x},\ \boldsymbol{x} \in \mathfrak{C}\}$

Вариационный метод
Реализация

Data: Two vector valued images I_0 and I_1
Result: Optical flow field u from I_0 to I_1
for $L = L_{\max}$ *to* 0 **do**
  `// Pyramid levels`
  Downsample the images I_0 and I_1 to current pyramid level
  for $W = 0$ *to* $W_{\max}$ **do**
    `// Warping`
    if $I_1(\boldsymbol{x} + u(\boldsymbol{x})) - I_0(\boldsymbol{x}) \in \mathrm{Im} J_{I_1}(\boldsymbol{x} + u(\boldsymbol{x}))$ **then**
      Compute v as the minimizer of E_1, using Proposition 1 (ii) on current pyramid level
    else
      Compute the minimizer v by gradient descent
    end
    for $I = 0$ *to* $I_{\max}$ **do**
      `// Inner iterations`
      Solve (3) for u on current pyramid level
    end
  end
  Upscale flows to next pyramid level
end

Algorithm 1: General TV-L^1 algorithm for vector valued images.

Lars Lau Rak, Lars Roholm, Mads Nielsen, Francois Lauze, "TV-L1 Optical Flow for Vector Valued Images", EMMCVPR, 2011

Вариационный метод
Проекция на эллиптический шар

Data: The point $\boldsymbol{w}$ and the matrix C specifying the bounding ellipsoid.
Result: The orthogonal projection of $\boldsymbol{w}$ onto the ellipsoid.
Set $\boldsymbol{v}^0 = \dfrac{\boldsymbol{w}}{\sqrt{\langle \boldsymbol{w}, C\boldsymbol{w} \rangle}}$, and let the stepsize $\tau > 0$ be small enough.
while $\boldsymbol{v}^n$ *has not converged* **do**

$$\boldsymbol{v}^{n+1} = \boldsymbol{v}^n + \tau(\boldsymbol{w} + \langle \boldsymbol{v}^n, \boldsymbol{v}^n - \boldsymbol{w} \rangle C\boldsymbol{v}^n) \quad \texttt{// Gradient step}$$

$$\boldsymbol{v}^{n+1} = \frac{\boldsymbol{v}^{n+1}}{\sqrt{\langle \boldsymbol{v}^{n+1}, C\boldsymbol{v}^{n+1} \rangle}} \quad \texttt{// Reprojection}$$

end

Algorithm 2: Projection of a point $\boldsymbol{w}$ onto an ellipsoid.

Lars Lau Rak, Lars Roholm, Mads Nielsen, Francois Lauze, "TV-L1 Optical Flow for Vector Valued Images", EMMCVPR, 2011

Вариационный метод
Реализация

- 5 уровней пирамиды с уменьшением разрешения вдвое
- На шаге проекции градиент оценивается бикубическим подходом (bicubic interpolation)
- Процедура минимизации (для предположения 1) хорошо распараллеливается
- Реализовано на CUDA C

Вариационный метод
Входные данные

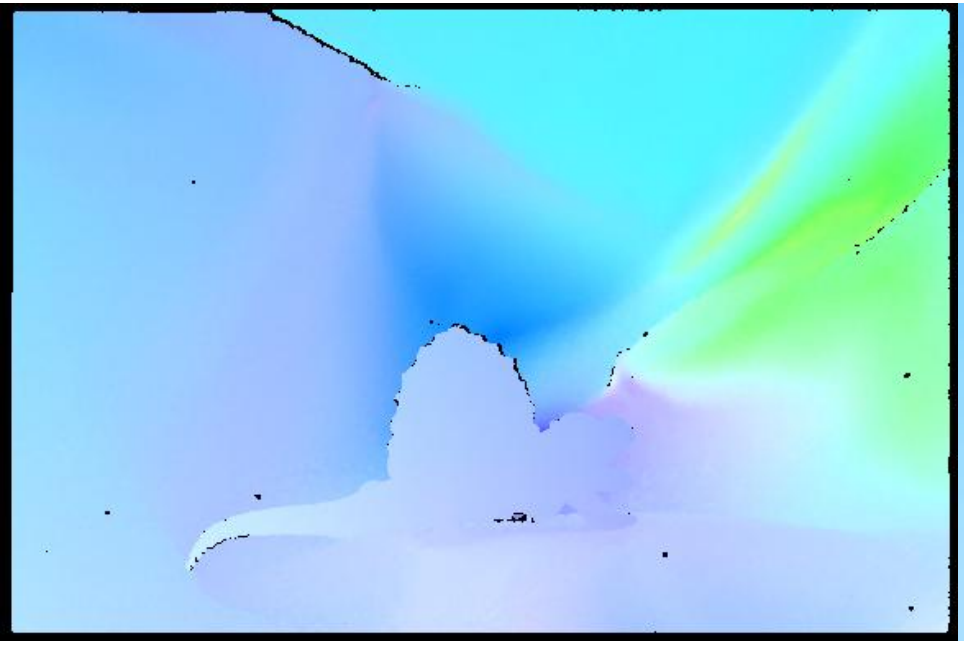
Color

Gradient

Laplacian

Lars Lau Rak, Lars Roholm, Mads Nielsen, Francois Lauze, "TV-L1 Optical Flow for Vector Valued Images", EMMCVPR, 2011

Вариационный метод
Результаты на dimetradon

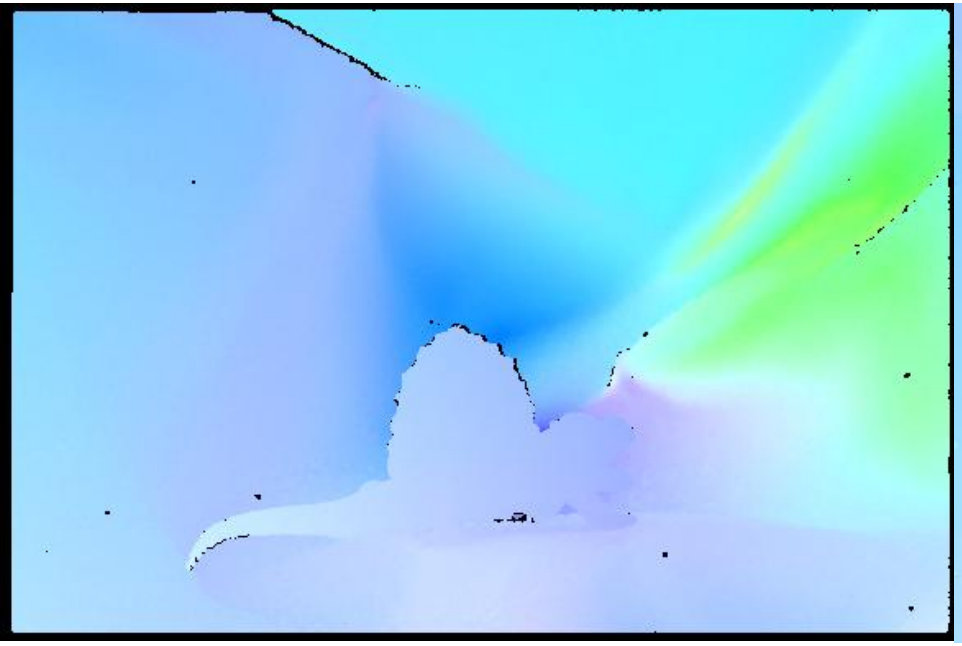


Ground truth

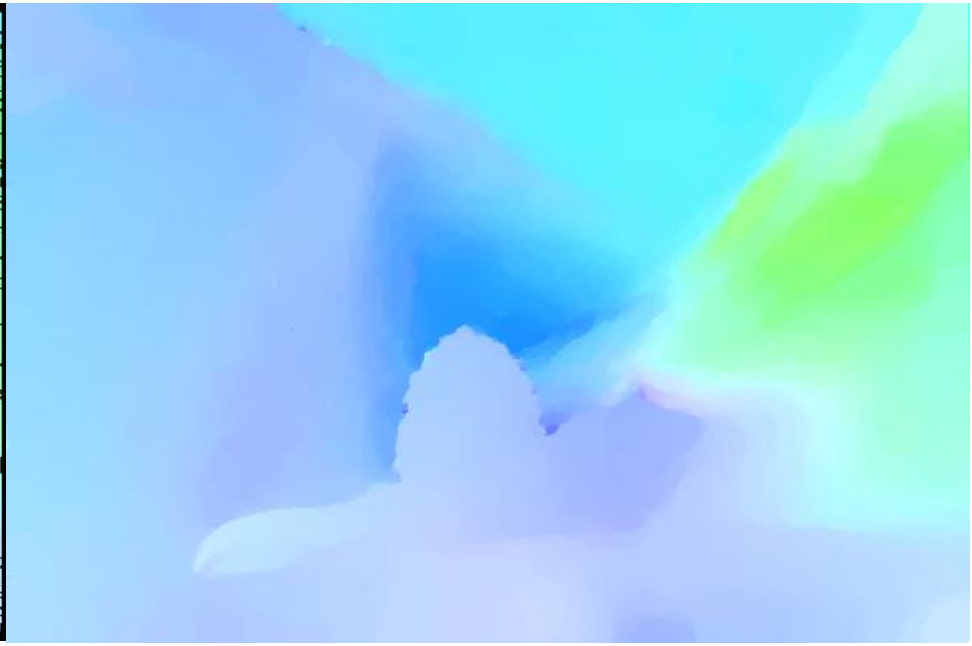
RGB-алгоритм

Lars Lau Rak, Lars Roholm, Mads Nielsen, Francois Lauze, "TV-L1 Optical Flow for Vector Valued Images", EMMCVPR, 2011

Вариационный метод
Результаты на dimetradon



Ground truth



GCA-алгоритм

Lars Lau Rak, Lars Roholm, Mads Nielsen, Francois Lauze, "TV-L1 Optical Flow for Vector Valued Images", EMMCVPR, 2011

Вариационный метод
Результаты на groove3

Ground truth

RGB-алгоритм

Lars Lau Rak, Lars Roholm, Mads Nielsen, Francois Lauze, "TV-L1 Optical Flow for Vector Valued Images", EMMCVPR, 2011

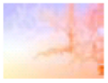
Вариационный метод
Результаты groove3

BCA-алгоритм

RGB+MF-алгоритм

Lars Lau Rak, Lars Roholm, Mads Nielsen, Francois Lauze, "TV-L1 Optical Flow for Vector Valued Images", EMMCVPR, 2011

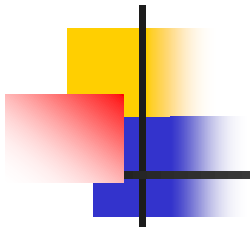
Вариационный метод
Average endpoint error

	BCA	GCA	RGB	Δ-RGB	RGB+MF	"improved" [9]
 Dimetrodon	0.14	**0.10**	0.16	0.22	**0.16**	0.19
 Grove2	0.18	0.23	**0.17**	0.24	**0.15**	**0.15**
 Grove3	0.85	0.76	**0.62**	0.84	**0.57**	0.67
 Hydrangea	**0.20**	0.22	0.24	0.23	0.25	**0.15**
 RubberWhale	0.20	0.20	**0.17**	0.18	0.17	**0.09**
 Urban2	0.59	0.42	**0.38**	1.52	0.36	**0.32**
Urban3	0.82	0.99	**0.62**	1.25	**0.50**	0.63
 Venus	0.54	0.58	**0.53**	0.67	0.49	**0.26**

Lars Lau Rak, Lars Roholm, Mads Nielsen, Francois Lauze, "TV-L1 Optical Flow for Vector Valued Images", EMMCVPR, 2011

Вариационный метод
Результаты

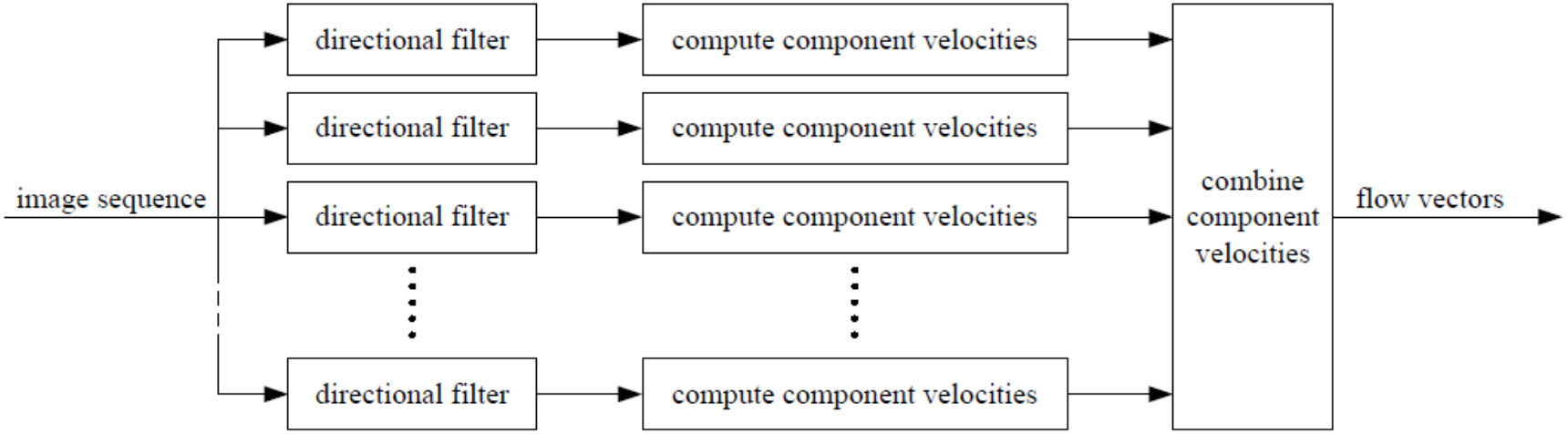
- Время расчета optical flow для пары 640x480 RGB кадров около 0.5 секунды
- За счет небольшого снижения точности (снижение количества итераций warps) RGB+MF алгоритм производит расчет optical flow для 640x480 RGB кадров в реальном времени



Содержание

- Введение
- Локальный метод
- Вариационный метод
- **Фазовый метод**
- Заключение

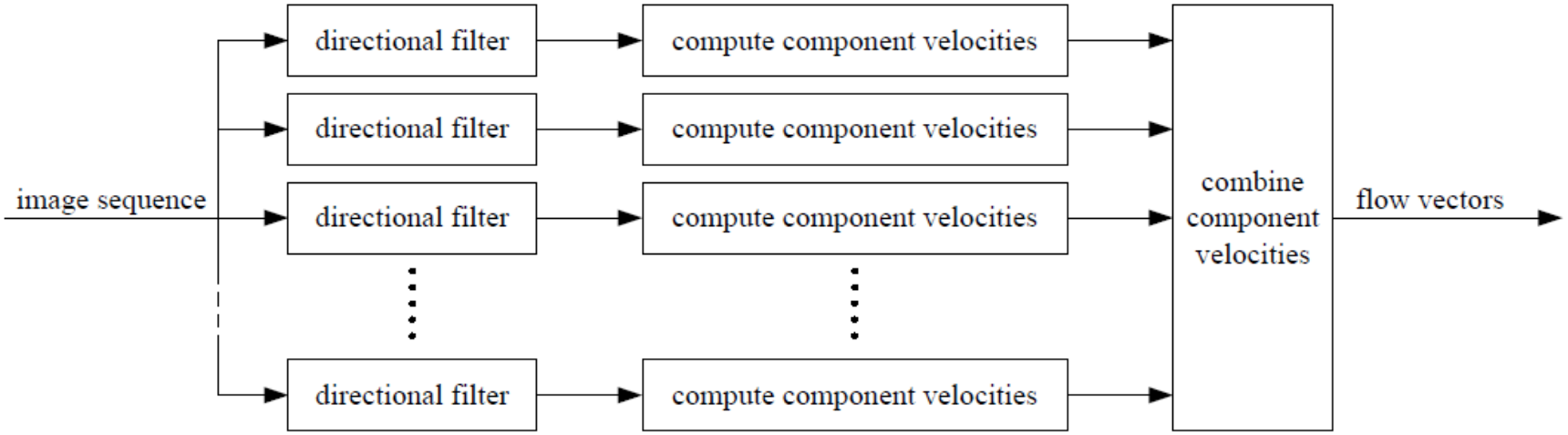
Фазовый метод
Базовый алгоритм



- Каждый кадр в последовательности раскладывается по направлениям набором фильтров направлений
- Для каждого направления считается величина и знак скорости (component velocities)

Robert Hegner, Ivar Austvoll, Tom Ryen, Guido M. Schuster," Efficient Implementation of Optical Flow Algorithm Based on Directional Filters on a GPU Using CUDA", EUSIPCO, 2011

Фазовый метод
Базовый алгоритм

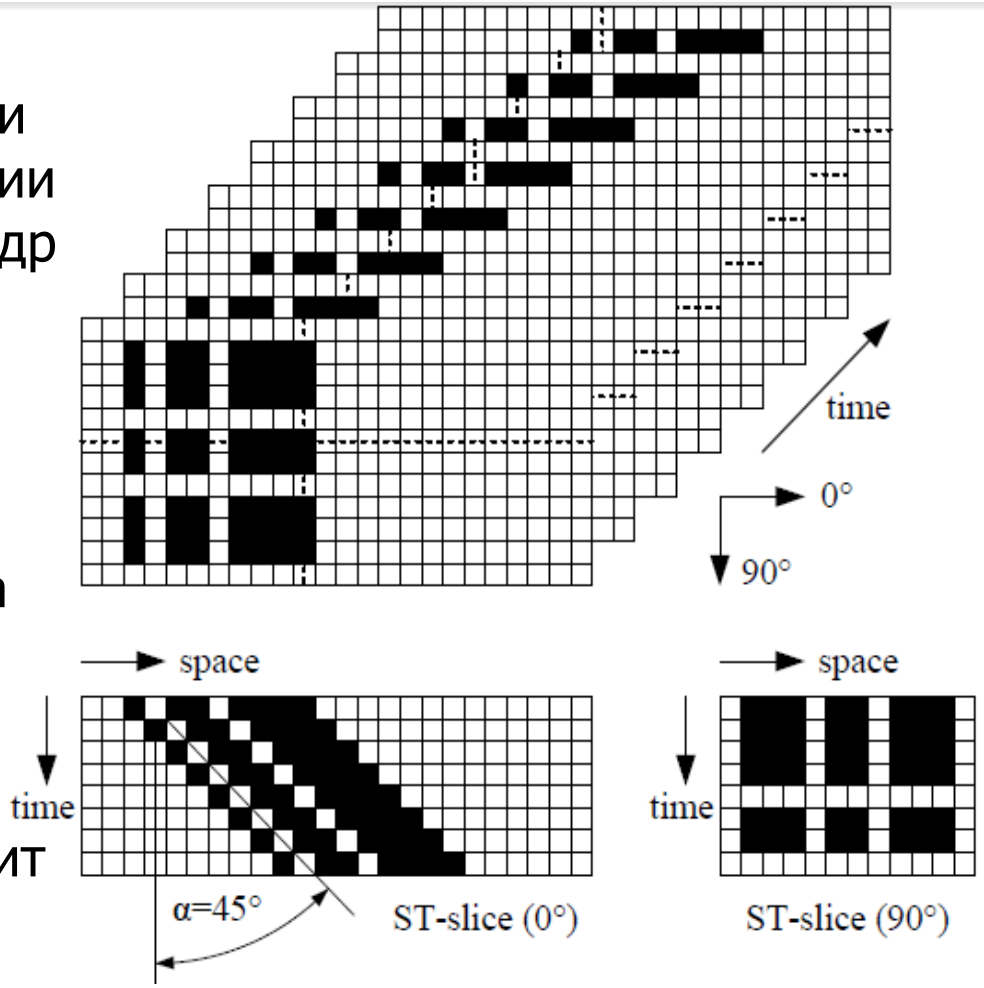


- Из системы линейных уравнений находятся начения x- и y-компоненты векторов optical flow

Robert Hegner, Ivar Austvoll, Tom Ryen, Guido M. Schuster,” Efficient Implementation of Optical Flow Algorithm Based on Directional Filters on a GPU Using CUDA”, EUSIPCO, 2011

Фазовый метод
Пример ST-slice

- Объект на последовательности кадров движется в направлении 0˚(вправо) на 1 пиксель за кадр
- Вдоль пунктирной линии сделано 2 среза (по направлениям 0˚ и 90˚) последовательности кадров
- Пространственная координата среза всегда соответствует направлению
- Component velocity для заданного направления зависит от tg α



Robert Hegner, Ivar Austvoll, Tom Ryen, Guido M. Schuster," Efficient Implementation of Optical Flow Algorithm Based on Directional Filters on a GPU Using CUDA", EUSIPCO, 2011

Фазовый метод
Фильтрация

Фильтр направлений состоит из двух частей: продольная и поперечная

- Продольная часть – фильтр низких частот с узкой полосой пропускания, пропускающий частотный спектр, находящийся в направлении фильтра
- Поперечная часть – bandwidth filter с широкой полосой, пропускающей всю часть спектра находящегося в заданной области для данного направления. Такой тип сигналов называется аналитическим (для негативных частот спектр сигнала равен нулю)

Robert Hegner, Ivar Austvoll, Tom Ryen, Guido M. Schuster," Efficient Implementation of Optical Flow Algorithm Based on Directional Filters on a GPU Using CUDA", EUSIPCO, 2011

Фазовый метод
Фильтрация

Т.к. фильтр направлений производит analytic signal, то опишем ST-slice как

$$z(\boldsymbol{s}) = a(\boldsymbol{s}) \cdot e^{j\phi(\boldsymbol{s})}$$

- S – вектор состоящий из координаты s направленной вдоль направления фильтра и временной координаты t
- $a(\boldsymbol{s})$ – функция амплитуды
- $\phi(\boldsymbol{s})$ – фазовая функция

Robert Hegner, Ivar Austvoll, Tom Ryen, Guido M. Schuster,” Efficient Implementation of Optical Flow Algorithm Based on Directional Filters on a GPU Using CUDA”, EUSIPCO, 2011

Фазовый метод
Мгновенное значение частота

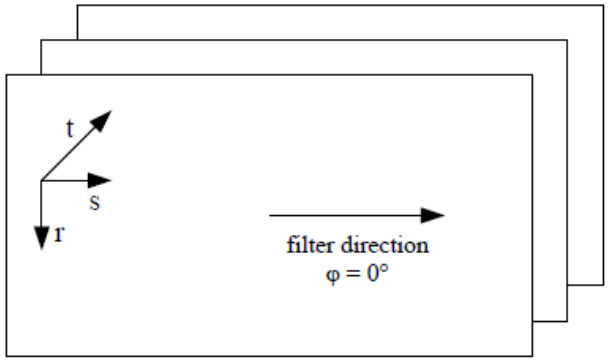
- Component velocity зависит от смещения фазы на ST-slice
- Следовательно, важная информация содержится в фазовом градиент $\nabla\phi(s)$, называемом мгновенным значением частоты (FM функция)

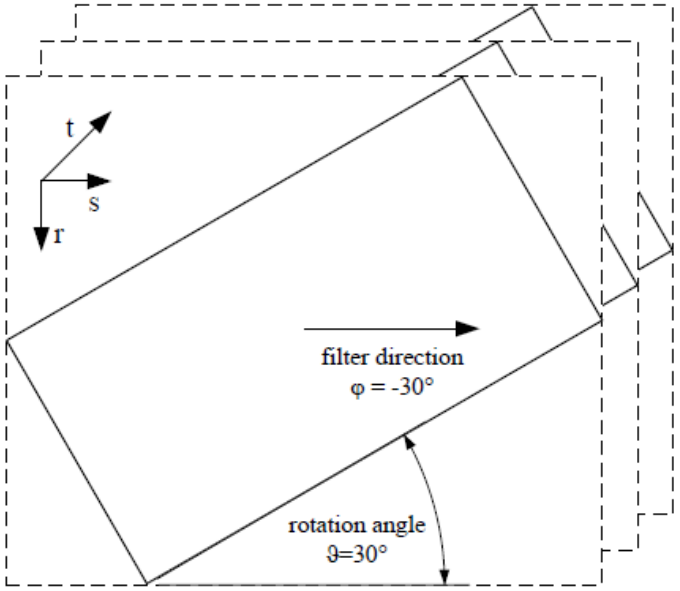
Robert Hegner, Ivar Austvoll, Tom Ryen, Guido M. Schuster," Efficient Implementation of Optical Flow Algorithm Based on Directional Filters on a GPU Using CUDA", EUSIPCO, 2011

Фазовый метод
Вращение

$$min_x = \min\{x'_1, x'_2, x'_3, x'_4\}$$
$$max_x = \max\{x'_1, x'_2, x'_3, x'_4\}$$
$$min_y = \min\{y'_1, y'_2, y'_3, y'_4\}$$
$$max_y = \max\{y'_1, y'_2, y'_3, y'_4\}$$
$$w' = \lceil max_x - min_x + 1 \rceil$$
$$h' = \lceil max_y - min_y + 1 \rceil$$

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} \cos\vartheta & -\sin\vartheta \\ \sin\vartheta & \cos\vartheta \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}$$

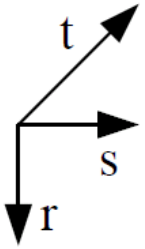




Robert Hegner, Ivar Austvoll, Tom Ryen, Guido M. Schuster," Efficient Implementation of Optical Flow Algorithm Based on Directional Filters on a GPU Using CUDA", EUSIPCO, 2011

Фазовый метод
Оценка мгновенных частот

- $z(r,s,t)$ – комплексная функция, получаемая из фильтра направлений
- component velocity получают, изучая смещение фазы на ST-slice



$$\left|\nabla_s \hat{\phi}(r,s,t)\right| = \arccos \mathrm{Re}\left\{\frac{z(r,s+1,t)+z(r,s-1,t)}{2\cdot z(r,s,t)}\right\}$$

$$\left|\nabla_t \hat{\phi}(r,s,t)\right| = \arccos \mathrm{Re}\left\{\frac{z(r,s,t+1)+z(r,s,t-1)}{2\cdot z(r,s,t)}\right\}$$

$$\operatorname{sgn}\nabla_s \hat{\phi}(r,s,t) = \operatorname{sgn}\arcsin \mathrm{Re}\left\{\frac{z(r,s+1,t)-z(r,s-1,t)}{2j\cdot z(r,s,t)}\right\}$$

$$\operatorname{sgn}\nabla_t \hat{\phi}(r,s,t) = \operatorname{sgn}\arcsin \mathrm{Re}\left\{\frac{z(r,s,t+1)-z(r,s,t-1)}{2j\cdot z(r,s,t)}\right\}$$

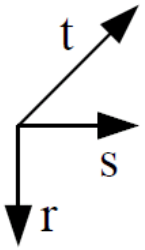
$$\nabla_s \hat{\phi}(r,s,t) = \operatorname{sgn}\nabla_s \hat{\phi}(r,s,t)\cdot\left|\nabla_s \hat{\phi}(r,s,t)\right|$$

$$\nabla_t \hat{\phi}(r,s,t) = \operatorname{sgn}\nabla_t \hat{\phi}(r,s,t)\cdot\left|\nabla_t \hat{\phi}(r,s,t)\right|$$

Robert Hegner, Ivar Austvoll, Tom Ryen, Guido M. Schuster," Efficient Implementation of Optical Flow Algorithm Based on Directional Filters on a GPU Using CUDA", EUSIPCO, 2011

Фазовый метод
Тензор

- Чтобы посчитать направление движения (угол α) структуры на ST-slice из мгновенного значения частоты, вводят тензор
- Тензор – матрица 2x2, описывающая структуру без ST-slice
- $\nabla_t \hat{\phi}(r,s,t)$, $\nabla_s \hat{\phi}(r,s,t)$ – мгновенные значения частоты



$$\boldsymbol{T}(r,s,t) = \begin{bmatrix} T_{ss}(r,s,t) & T_{st}(r,s,t) \\ T_{st}(r,s,t) & T_{tt}(r,s,t) \end{bmatrix}$$

$$T_{ss}(r,s,t) = \left\langle \nabla_s \hat{\phi}_i \cdot \nabla_s \hat{\phi}_i \right\rangle_{i \in \Omega}$$

$$T_{st}(r,s,t) = \left\langle \nabla_s \hat{\phi}_i \cdot \nabla_t \hat{\phi}_i \right\rangle_{i \in \Omega}$$

$$T_{tt}(r,s,t) = \left\langle \nabla_t \hat{\phi}_i \cdot \nabla_t \hat{\phi}_i \right\rangle_{i \in \Omega}$$

Robert Hegner, Ivar Austvoll, Tom Ryen, Guido M. Schuster," Efficient Implementation of Optical Flow Algorithm Based on Directional Filters on a GPU Using CUDA", EUSIPCO, 2011

Фазовый метод
Тензор

- e_1 – собственный вектор, определяется наименьшим собственным значением $T(r,s,t)$

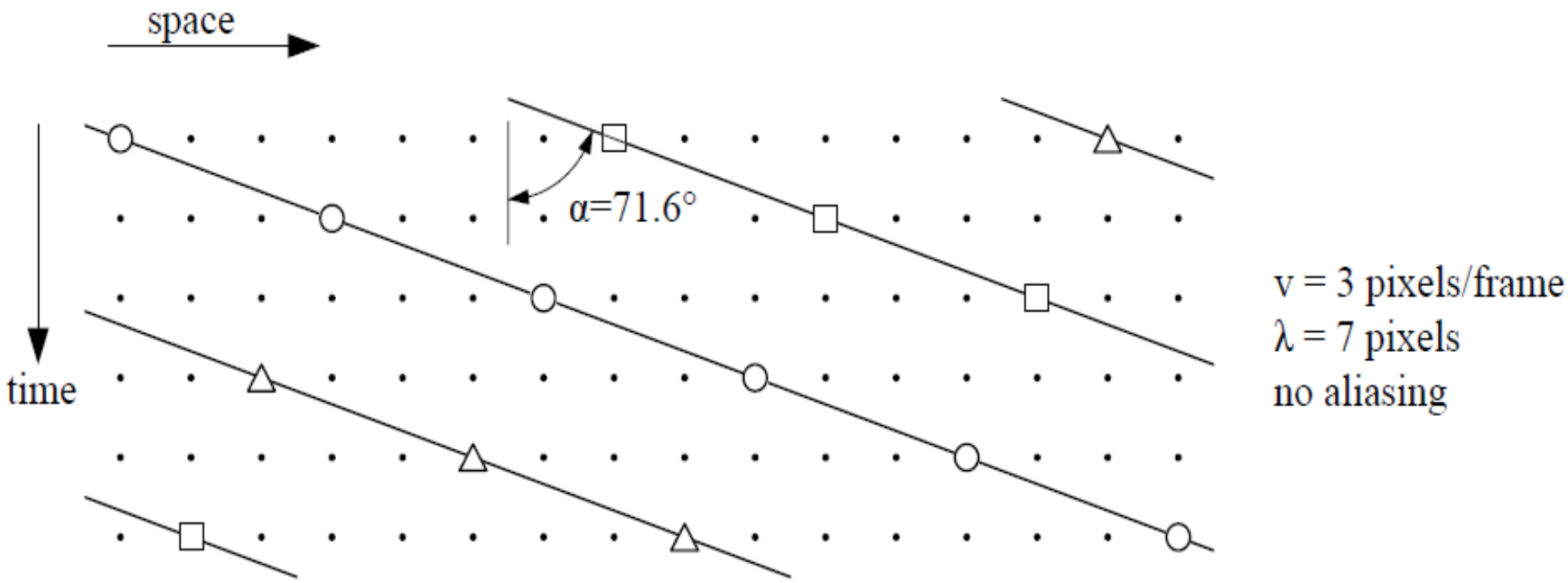
- Тогда component velocities ищется как

$$v_c = \tan(\alpha) = \frac{e_1(1)}{e_1(2)}$$

Robert Hegner, Ivar Austvoll, Tom Ryen, Guido M. Schuster,” Efficient Implementation of Optical Flow Algorithm Based on Directional Filters on a GPU Using CUDA”, EUSIPCO, 2011

Фазовый метод
Алиасинг

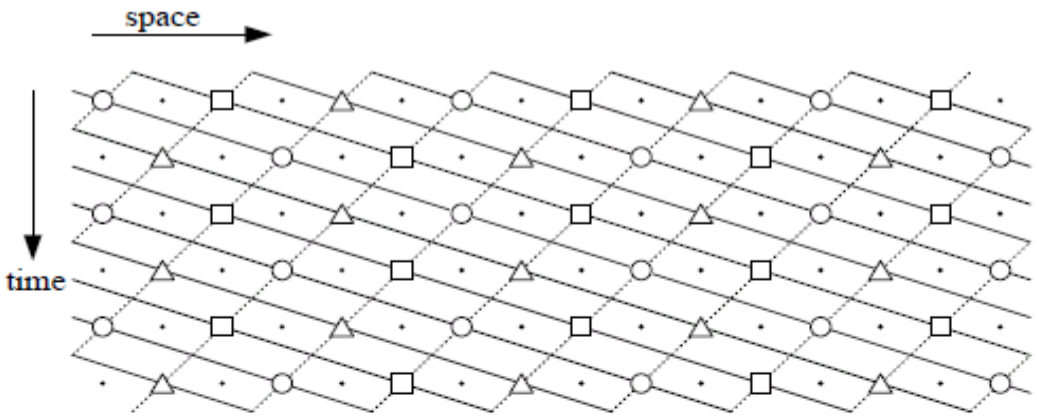
В базовом алгоритме при высоких скоростях появляется алиасингом



пространственно временной срез

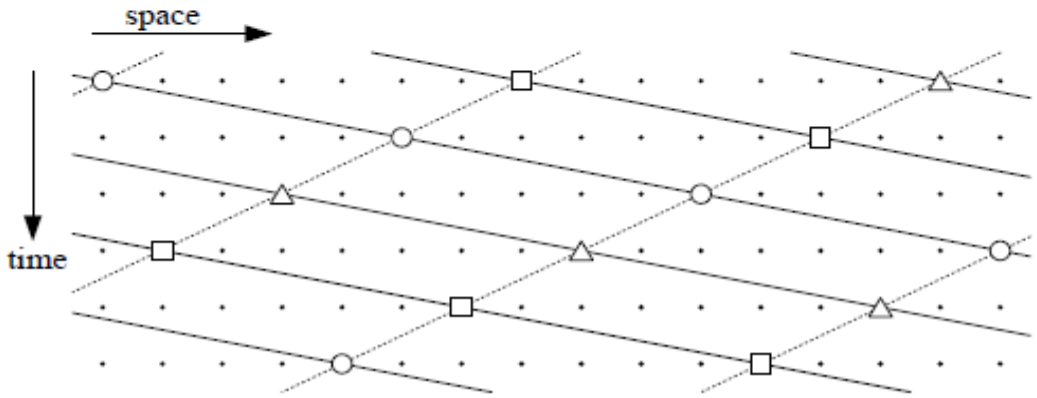
Robert Hegner, Ivar Austvoll, Tom Ryen, Guido M. Schuster," Efficient Implementation of Optical Flow Algorithm Based on Directional Filters on a GPU Using CUDA", EUSIPCO, 2011

Фазовый метод
Алиасинг



v = 3 pixels/frame
λ = 2 pixels
aliasing!

пространственно временной срез

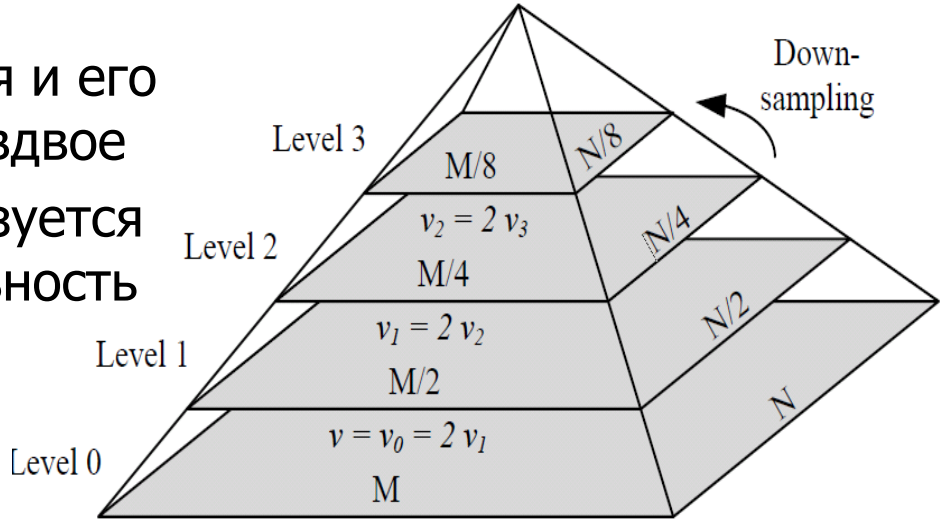


v = 5 pixels/frame
λ = 7 pixels
aliasing!

Robert Hegner, Ivar Austvoll, Tom Ryen, Guido M. Schuster," Efficient Implementation of Optical Flow Algorithm Based on Directional Filters on a GPU Using CUDA", EUSIPCO, 2011

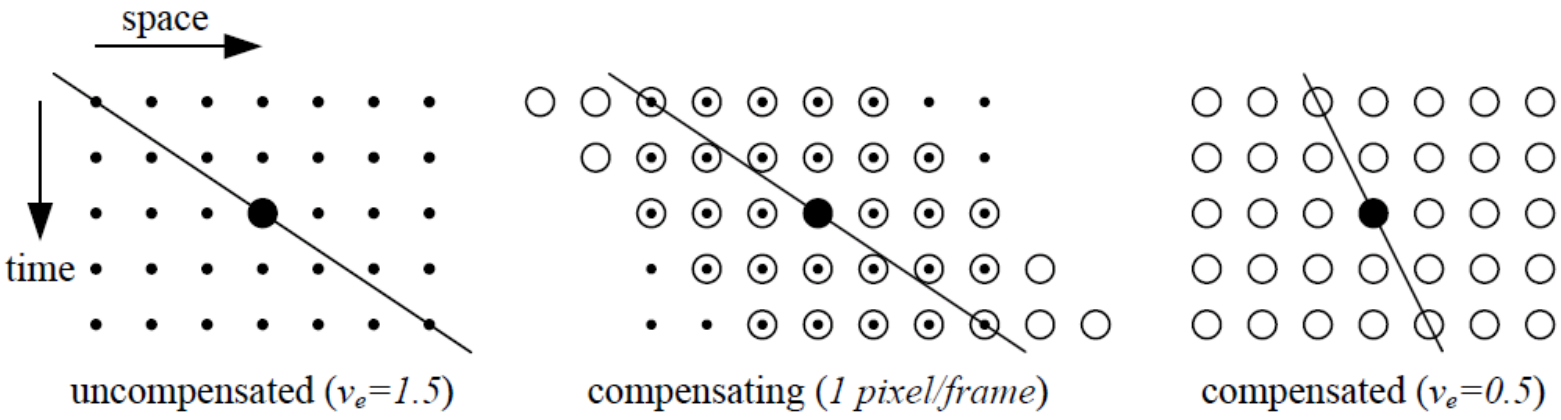
Фазовый метод
Pyramid algorithm

- Для лучшей оценки высоких скоростей используется пирамидальная схема с множителем 2
- Изображение сглаживается и его разрешение уменьшается вдвое
- Для грубой оценки используется сглаженная последовательность с низким разрешением



Robert Hegner, Ivar Austvoll, Tom Ryen, Guido M. Schuster," Efficient Implementation of Optical Flow Algorithm Based on Directional Filters on a GPU Using CUDA", EUSIPCO, 2011

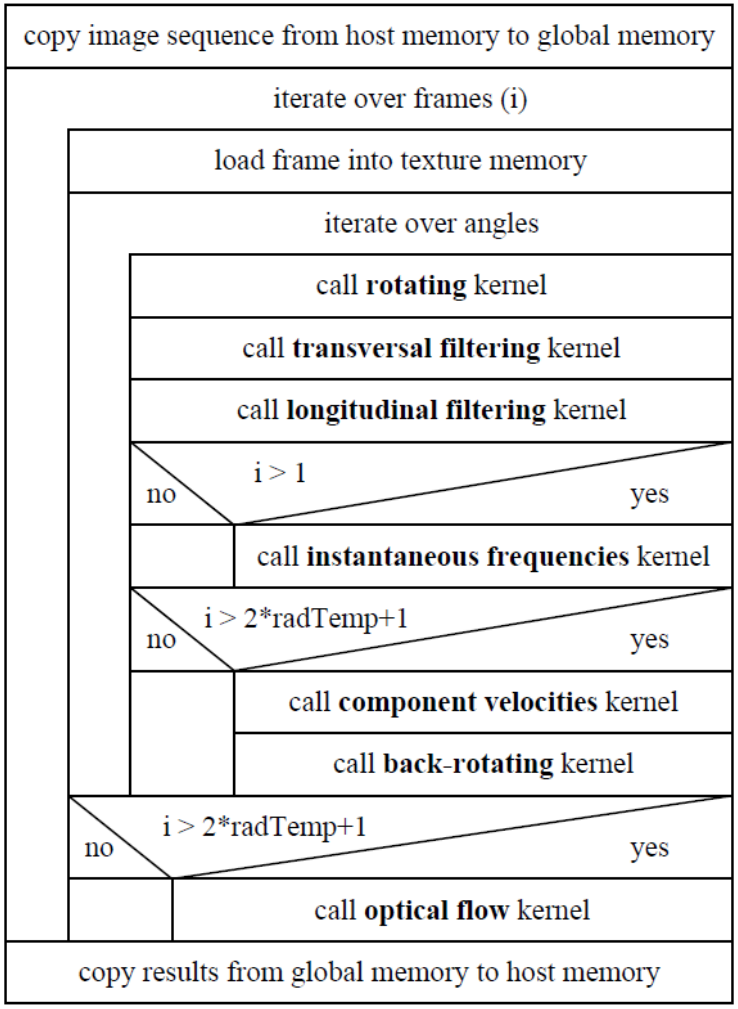
Фазовый метод
Компенсация



Robert Hegner, Ivar Austvoll, Tom Ryen, Guido M. Schuster," Efficient Implementation of Optical Flow Algorithm Based on Directional Filters on a GPU Using CUDA", EUSIPCO, 2011

Фазовый метод
Реализация базового алгоритма(1)

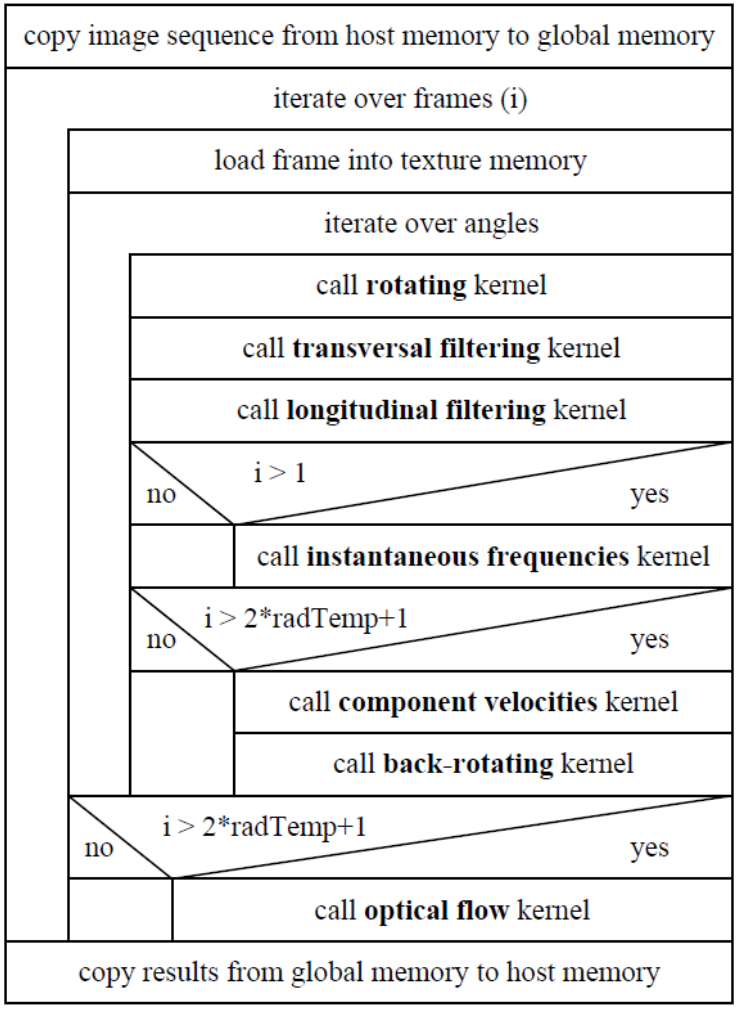
- Передача данных между RAM и global memory дорога, поэтому вся последовательность копируется в global memory
- Все промежуточные результаты хранятся в GPU
- Текущий кадр загружается в texture memory
- При подсчете мгновенных частот кадра требуются результаты работы фильтра направлений для предыдущего, текущего и следующего кадра



Robert Hegner, Ivar Austvoll, Tom Ryen, Guido M. Schuster," Efficient Implementation of Optical Flow Algorithm Based on Directional Filters on a GPU Using CUDA", EUSIPCO, 2011

Фазовый метод
Реализация базового алгоритма(2)

- При подсчете component velocities для одного кадра необходим набор мгновенных частот (в зависимости от временного радиуса ST-slice)
- Продольная и поперечная части фильтра направлений выполняются последовательно
- Вместо использования набора фильтров направлений кадр вращается. После подсчета component velocities происходит обратный поворот



Robert Hegner, Ivar Austvoll, Tom Ryen, Guido M. Schuster," Efficient Implementation of Optical Flow Algorithm Based on Directional Filters on a GPU Using CUDA", EUSIPCO, 2011

Фазовый метод
Оптимизация

- Transversal filtering kernel эффективен с shared memory
- Longitudinal filtering kernel эффективен с texture memory
- Использование векторного типа (float2) снижает количество обращений к памяти, считывая и записывая несколько переменных за одну транзакцию
- Например, используя векторный тип вместе с texture memory, число обращений к back-rotating kernel снижается на 50%

Robert Hegner, Ivar Austvoll, Tom Ryen, Guido M. Schuster,” Efficient Implementation of Optical Flow Algorithm Based on Directional Filters on a GPU Using CUDA”, EUSIPCO, 2011

Фазовый метод
Результаты(1)

Технические характеристики: Intel Core 2 Quad 2.4GHz, 2GB DDR2-SDRAM, NVIDIA GeForce GTX 260, CUDA compute capability 1.3, 27 multiprocessors, 216 cores, 896MB global memory

Image sequence size	Matlab [*min*]	CUDA [*sec*]	Speedup
$190 \times 256 \times 21$	10.3	0.124	4977
$240 \times 256 \times 21$	12.8	0.141	5434
$512 \times 512 \times 50$	343.2	1.372	15009

Robert Hegner, Ivar Austvoll, Tom Ryen, Guido M. Schuster," Efficient Implementation of Optical Flow Algorithm Based on Directional Filters on a GPU Using CUDA", EUSIPCO, 2011

Фазовый метод
Результаты(2)

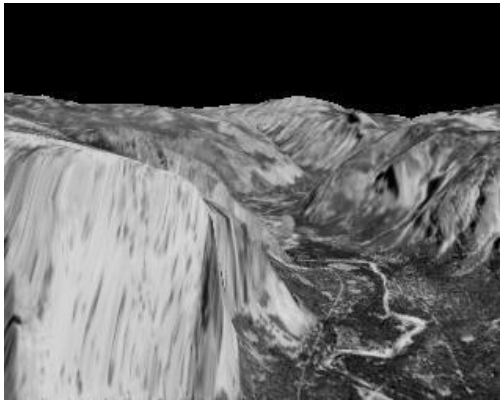
GRAPHICS & MEDIA LAB
VIDEO GROUP

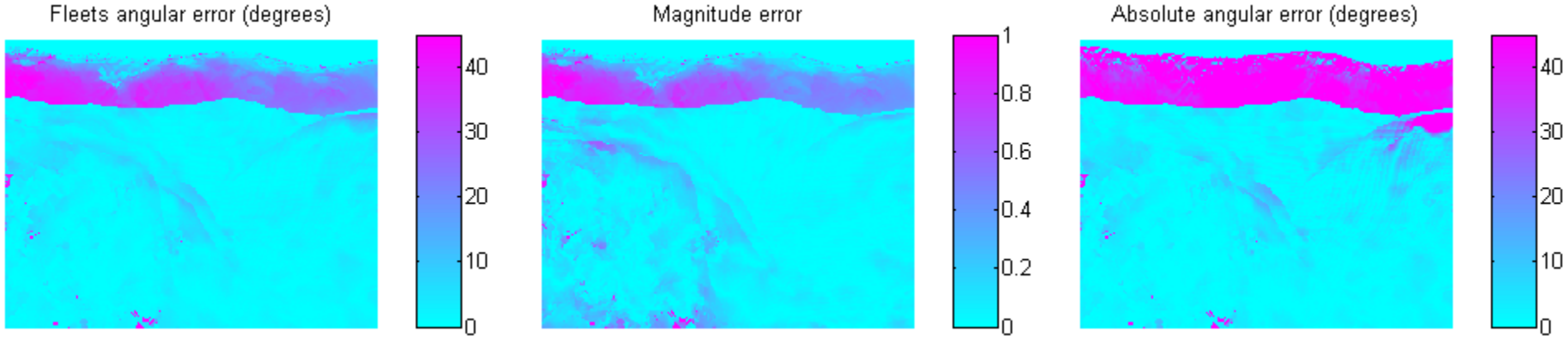
Sequence	Pyramid Algorithm CUDA	Basic Algorithm CUDA	Lucas-Kanade Extended CUDA	Lucas-Kanade OpenCV	Lucas-Kanade Pyramid OpenCV	Horn-Schunck OpenCV
Diverging Tree	**1.322**	1.750	2.147	4.631	3.634	7.011
Translating Tree	0.581	0.874	0.823	11.310	**0.307**	14.508
Rubber Whale	**8.453**	9.173	13.022	17.339	13.249	18.051
Yosemite Cloudless	6.163	9.625	4.136	7.852	**3.996**	8.225
Grove 2	**5.224**	31.437	10.907	17.862	5.689	20.151
Hydrangea	**7.562**	84.220	23.635	24.473	8.336	25.317
Urban 3	16.105	73.862	35.990	44.164	**11.369**	44.969
Grove 3	12.556	46.217	26.547	29.798	**12.483**	30.994
Urban 2	28.559	52.872	54.494	49.804	**18.692**	48.815

average angle error

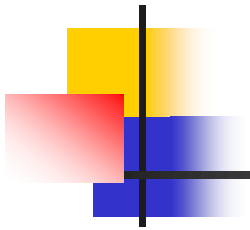
Robert Hegner, Ivar Austvoll, Tom Ryen, Guido M. Schuster,” Efficient Implementation of Optical Flow Algorithm Based on Directional Filters on a GPU Using CUDA”, EUSIPCO, 2011

Фазовый метод
Результаты Yosemite Cloudless





Robert Hegner, Ivar Austvoll, Tom Ryen, Guido M. Schuster," Efficient Implementation of Optical Flow Algorithm Based on Directional Filters on a GPU Using CUDA", EUSIPCO, 2011



Содержание

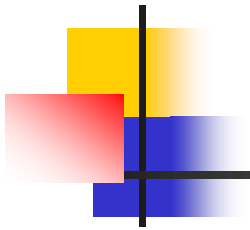
- Введение
- Локальный метод
- Вариационный метод
- Фазовый метод
- **Заключение**

Заключение

- Алгоритмы с фильтрами направлений хорошо распараллеливаются
- TV-L1 демонстрирует высокое качество в реальном времени
- Для оценки сильного движения современные методы часто используют пирамидальные алгоритмы

Литература

1. S. Baker, D. Scharstein, JP Lewis, S. Roth, M. Black, R. Szeliski. A Database and Evaluation Methodology for Optical Flow, 2011
2. Julien Marzat, Yann Dumortier, Andre Ducrot, "Real-Time Dense and Accurate Parallel Optical Flow using CUDA", WSCG, 2009
3. Robert Hegner, Ivar Austvoll, Tom Ryen, Guido M. Schuster," Efficient Implementation of Optical Flow Algorithm Based on Directional Filters on a GPU Using CUDA", EUSIPCO, 2011
4. Lars Lau Rak, Lars Roholm, Mads Nielsen, Francois Lauze, "TV-L1 Optical Flow for Vector Valued Images", EMMCVPR, 2011
5. Middlebury benchmark. http://vision.middlebury.edu/flow/



GRAPHICS & MEDIA LAB
VIDEO GROUP

?

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