Видео как временной сигнал

Модель I(x, y, t)

Трёхмерная природа

Видео — не набор кадров, а связанная временная структура с внутренней динамикой

Физический процесс

Интенсивность пикселя — проявление процесса в мировых координатах

Временная структура

Гладкость и коррелированность требуют методов анализа сигналов

Трёхмерная природа видеосигнала

Видео — наблюдение непрерывного оптического процесса, дискретизированного камерой в функцию I(x, y, t), где t — параметр времени.

> I(x,y,t) — не коллекция изображений, а связанная во времени структура с гладкостью и внутренней динамикой

Интенсивность пикселя — проявление физического процесса в мировой системе координат. Пиксельные координаты — лишь проекция, зависящая от параметров камеры, движения объектов и освещения.

Временная дискретизация

Частота кадров

Камера снимает с частотой fs кадров в секунду

$$\Delta t = \frac{1}{f_s}$$

Временной шаг определяет минимальное различимое изменение состояния сцены

Временная производная

$$I_t(x, y, t) \approx I(x, y, t + \Delta t) - I(x, y, t)$$

Если Δt слишком велико, движение между кадрами разрушает возможность вычислять производную

Эффект aliasing-движения возникает при больших смещениях объектов за кадр — аналог частотных артефактов в аудио

Временная когерентность

Свойство, при котором изменения яркости принадлежат дифференцируемой траектории объекта, а не случайному процессу.

$$I(x, y, t) \approx I(x + u, y + v, t + \Delta t)$$

Принцип: «яркость движется вместе с объектом»

Нарушения когерентности

- Резкое изменение освещения
- Отражения и блики
- Прозрачные объекты
- Тени с иной траекторией
- Мерцание источников света

Факторы нарушения гладкости

Каждый кадр — результат интегрирования фотонного потока в течение времени экспозиции τ:

$$I_{obs}(x, y, t) = \int_{t-\tau}^{t} I(x, y, s)\, ds$$

1

Motion blur

Быстрое движение вызывает размытие, исчезают высокочастотные компоненты

2

Компрессия

H.264 создаёт блоковые структуры макроблоков 16×16 с нелинейными артефактами

3

Rolling shutter

Анизотропия по строкам: I(x,y,t) → I(x,y,t+αy), где α — задержка между строками

Пространственно-временные производные

Вводятся три производные:

$$I_x, \quad I_y, \quad I_t$$

Дифференциальная модель

Предполагается гладкость функции I(x,y,t). В цифровом виде она дискретна, шумна и имеет разрывы.

Применяется гауссово сглаживание:

$$I_x = G_\sigma * \frac{\partial I}{\partial x}$$

$$I_y = G_\sigma * \frac{\partial I}{\partial y}$$

Корректность вычислений

Нарушение гладкости приводит к нестабильным значениям производных.

Уравнение оптического потока становится плохо обусловленным.

Gσ — гауссово ядро для стабилизации

Практическое исследование

Простейшее взаимодействие с временными разностями:

```
import cv2
import numpy as np

cap = cv2.VideoCapture("input.mp4")
ret, prev = cap.read()
prev_gray = cv2.cvtColor(prev, cv2.COLOR_BGR2GRAY)

while True:
 ret, frame = cap.read()
 if not ret: break

 gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
 It = cv2.absdiff(gray, prev_gray)

 cv2.imshow("Frame", gray)
 cv2.imshow("Temporal difference |It|", It)

 prev_gray = gray.copy()
 if cv2.waitKey(1) & 0xFF == ord('q'): break
```

Инженерные замечания

Ограничения разностей

Абсолютная разность не различает движение и вспышку света — фиксирует любое изменение

Быстрые движения

Дают насыщенные области, но могут быть размазаны из-за motion blur

Статичная сцена

Разность показывает шум сенсора и компрессии — инструмент диагностики качества

Необходимость физической модели

Что не может разность

- Различить реальное движение
- Отделить тень от облака
- Учесть изменение экспозиции
- Отфильтровать шумовые флуктуации

Что даёт оптический поток

Векторное поле смещений, объясняющее изменение интенсивности

Зависимость между пространственными и временными производными

> Разность отвечает на вопрос «что изменилось», поток — на вопросы «как» и «куда»

Сохранение яркости вдоль траектории

Точка мира проецируется на пиксель (x,y) в момент t. При движении со скоростью (u,v) в момент t+Δt она появляется в (x+uΔt, y+vΔt).

$$I(x, y, t) = I(x + u\Delta t, y + v\Delta t, t + \Delta t)$$

01

Условия справедливости

Малые Δt, медленное изменение освещения

02

Отсутствие размытия

Нет сильного motion blur

03

Дифференцируемость

Движение должно быть гладким

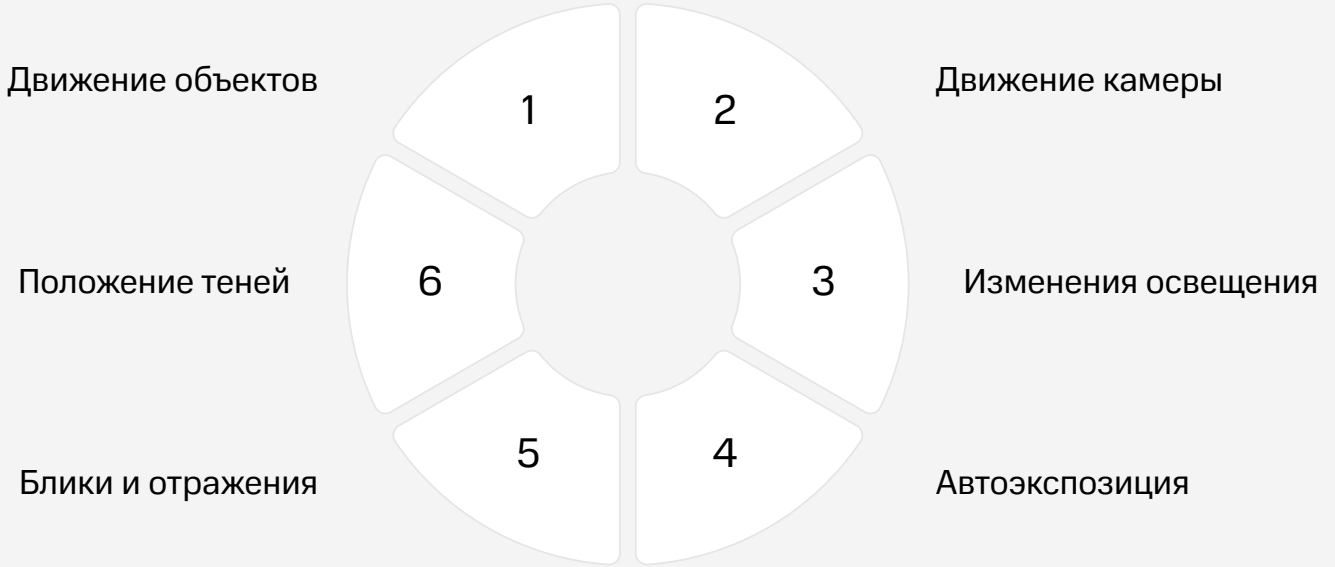
Линеаризация по Тейлору приводит к уравнению яркостной консистентности

Видеосигнал как физическая система

Интенсивность I(x,y,t) — не отражательная способность, а результат визуального образа:

$$I(x, y, t) = \mathcal{P}(L(\mathbf{X}, t))$$

где L(**X**, t) — освещённость точки мира **X**, а — перспектива камеры



Итоги раздела

Сформировано базовое понимание видеоряда как трёхмерного сигнала, где каждое значение зависит от структуры сцены, освещения, параметров камеры и времени.

1

Простые разности

Недостаточно информации о направлении движения

2

Физическое предположение

Перенос яркости создаёт основу для математической модели

3

Следующий шаг

Уравнение яркостной консистентности и алгоритмы

Уравнение яркостной консистентности

Дифференциальные методы оптического потока

От переноса яркости к аналитической модели

Ключевое физическое предположение: яркость точки сцены сохраняется вдоль её траектории.

$$I(x, y, t) = I(x + u\Delta t, y + v\Delta t, t + \Delta t)$$

Инвариантность яркости

Равенство отражает не статичность сцены, а инвариантность яркости вдоль траектории движения

Разложение Тейлора

Применяется по малым Δt, предполагая малость смещений по сравнению с размерами объектов

> Математическая модель движения — производная от физических свойств оптической сцены

Рождение уравнения яркостной консистентности

Разложение по Тейлору:

$$I(x + u\Delta t, y + v\Delta t, t + \Delta t) \approx I(x, y, t) + I_x u\Delta t + I_y v\Delta t + I_t \Delta t$$

Сокращая I(x,y,t), получаем фундаментальное уравнение:

$$I_x u + I_y v + I_t = 0$$

Ix, Iy

Частные производные яркости в пространстве

It

Временная производная

u, v

Компоненты оптического потока

Проблема: два неизвестных (u, v) и одно уравнение — задача недоопределена

Геометрическая интерпретация

Уравнение Ixu + Iyv + It = 0 означает, что вектор (u, v) должен лежать в подпространстве, ортогональном градиенту (Ix, Iy).

> ### Проблема апертуры
>
> Через маленькое «отверстие» наблюдателю доступна только компонента скорости, перпендикулярная градиенту

Пример

Движущийся вертикальный кант может сдвигаться в любом направлении вдоль своей протяжённости

Градиент фиксирует только смещение поперёк контура

Решение

Требуются дополнительные ограничения, обеспечивающие анизотропию локальной текстуры

Градиентное поле без разнообразия направлений → вырожденная матрица

Локальная аппроксимация постоянной скорости

Предположение: в небольшой окрестности окна W все пиксели движутся одинаково.

$$I_x^{(i)} u + I_y^{(i)} v + I_t^{(i)} = 0, \quad i = 1, \dots, n$$

Векторная форма: **Av** = **b**

$$A = \begin{pmatrix} I_x^{(1)} & I_y^{(1)} \\ \vdots & \vdots \\ I_x^{(n)} & I_y^{(n)} \end{pmatrix} \qquad \mathbf{v} = \begin{pmatrix} u \\ v \end{pmatrix} \qquad \mathbf{b} = \begin{pmatrix} I_t^{(1)} \\ \vdots \\ I_t^{(n)} \end{pmatrix}$$

Решение МНК:

$$\mathbf{v} = (A^\top A)^{-1} A^\top \mathbf{b}$$

Метод Лукаса–Канаде использует именно эту идею

Матрица градиентов и детектор Харриса

Матрица структурного тензора:

$$G = A^\top A = \begin{pmatrix} \Sigma I_x^2 & \Sigma I_x I_y \\ \Sigma I_x I_y & \Sigma I_y^2 \end{pmatrix}$$

Оба λ малы

Нет структуры → поток неопределён

Один λ мал

Кант → поток определён только перпендикулярно контуру

Оба λ велики

Угол → локальное движение хорошо фиксируется

Эта матрица лежит в основе детектора Харриса, выделяющего угловые точки

Численная устойчивость решения

Решение **v** = (A⊠A)-1A⊠**b** существует, когда A⊠A невырождена.

Проблема

Если det(G) мал, решение чувствительно к шуму

Методы стабилизации

- Сглаживание перед вычислением производных
- Гауссовы веса в окне
- Многоуровневые пирамиды
- Ограничения на собственные числа

Практическая реализация Лукаса–Канаде

```
import cv2
import numpy as np

prev = cv2.imread("frame1.png", cv2.IMREAD_GRAYSCALE)
next = cv2.imread("frame2.png", cv2.IMREAD_GRAYSCALE)

points = cv2.goodFeaturesToTrack(prev, maxCorners=200,
 qualityLevel=0.01,
 minDistance=5)

flow, status, err = cv2.calcOpticalFlowPyrLK(prev, next,
 points, None)

for (p0, p1, ok) in zip(points, flow, status):
 if ok:
 x0, y0 = p0.ravel()
 x1, y1 = p1.ravel()
 cv2.arrowedLine(prev, (x0,y0), (x1,y1), 255, 1)
```

Инженерные замечания к LK

calcOpticalFlowPyrLK

Использует пирамидальный LK, устойчивый к большим смещениям

goodFeaturesToTrack

Выбирает «углы», где матрица G хорошо обусловлена

Источники ошибок

Текстурно-плохие области, блики, тесселяция, тени

HDR и компрессия

LK часто требует нормализации яркости для закомпрессированных видео

Ограничения дифференциальных методов

1 Малость смещений

При движении более 3–5 пикселей линейная аппроксимация нарушается

2 Неизменность яркости

Нарушается при тенях, отражениях, экспозиционных скачках

3 Однородные текстуры

Поток восстанавливается плохо вдоль кантов

4 Шум в производных

Порождает сильные выбросы в решениях

Главная трудность — апертура: движение вдоль контура невозможно определить дифференциальными методами

Развитие идей: пирамидальный LK

Грубый уровень

Малое разрешение → малые смещения, вычисление потока

Интерполяция

Поток интерполируется на более высокий уровень

Уточнение

Нелинейная оптимизация на многоуровневой поверхности ошибок

Пирамиды делают методы менее чувствительными к шуму — грубый уровень подавляет высокочастотные компоненты

Переход к плотным методам

Ограничения LK

Хорошо работает для точек, но не предоставляет плотное поле движения

В условиях компрессии, шума, низкой текстуры часто недостаточен

Плотные методы

Оценивают движение для каждого пикселя

Используют более богатые модели: локальные полиномиальные аппроксимации

> Алгоритм Фарнебэка — один из наиболее успешных и используемых плотных методов

Плотный оптический поток

Метод Фарнебэка и инженерные нюансы

От разреженного к плотному потоку

Лукас–Канаде даёт высокую точность только для «хороших» точек. Многие задачи требуют непрерывное поле смещений для каждого пикселя.

Сегментация движения

Выделение движущихся областей

Стабилизация видео

Компенсация движения между кадрами

Анализ деформаций

Расчёт моментных карт и накопление света

Разреженные методы не способны реконструировать поток в слаботекстурированных областях

Квадратичная аппроксимация интенсивности

В окрестности каждого пикселя интенсивность представляется как квадратичная форма:

$$I(\mathbf{x}) \approx \mathbf{x}^\top A\mathbf{x} + \mathbf{b}^\top \mathbf{x} + c$$

где **x** = (x, y), A — симметричная матрица 2×2, **b** — вектор, c — скаляр

Локальная поверхность

Модель описывает яркость как эллиптическую параболу

Трансформация при сдвиге

Если изображение смещается на вектор **d** = (u, v), можно аналитически выразить изменение коэффициентов

Вывод смещения из параболической модели

Для первого кадра:

$$I_1(\mathbf{x}) \approx \mathbf{x}^\top A \mathbf{x} + \mathbf{b}^\top \mathbf{x} + c$$

Для второго кадра, сдвинутого на **d**:

$$I_2(\mathbf{x}) \approx I_1(\mathbf{x} + \mathbf{d})$$

Изменение линейного члена:

$$\mathbf{b}_2 = \mathbf{b}_1 + 2A\mathbf{d}$$

Вектор смещения:

$$\mathbf{d} = \frac{1}{2} A^{-1}(\mathbf{b}_2 - \mathbf{b}_1)$$

> Метод Фарнебэка формирует параболическую модель вокруг каждого пикселя для обоих кадров и извлекает смещение из аналитического преобразования параметров

Эффективная аппроксимация

Шум, компрессия и градиентная неоднородность делают прямое вычисление A и **b** нестабильным.

01

Гауссово сглаживание

Локальная структура становится ближе к параболической

02

Корреляционные ядра

Вычисление квадратичных коэффициентов через фильтрацию

03

Симметризация матриц

Устранение ошибок дискретизации

04

Итеративное уточнение

Компенсация больших смещений

Метод опирается на согласованность локальных «парабол» между кадрами

Многоуровневые пирамиды

Уменьшение изображения

Размытие + downsample

Оценка на низком уровне

Мелкое разрешение, грубый поток

Интерполяция вверх

Поток переносится на следующий уровень

Уточнение

Итеративное улучшение до исходного разрешения

Пирамидальная структура делает метод устойчивым при смещениях в десятки пикселей

Вычислительная сложность

Плотный поток требует оценивания модели для каждого пикселя. Для изображения W×H:

O(1)	$\log_2$	3–5
Оценка модели	Уровни пирамид	Итерации
Операций на пиксель	$\log_2 \min(W,H)$	Уточнений на уровень

Общая сложность:

$$O(WH \log WH)$$

Фарнебэк тяжелее LK, но достаточно быстр для реального времени при оптимизированной реализации

Реализация Фарнебэка в OpenCV

```
import cv2
import numpy as np

prev = cv2.imread("frame1.png", cv2.IMREAD_GRAYSCALE)
next = cv2.imread("frame2.png", cv2.IMREAD_GRAYSCALE)

flow = cv2.calcOpticalFlowFarneback(
 prev, next, None,
 pyr_scale=0.5,
 levels=5,
 winsize=15,
 iterations=5,
 poly_n=5,
 poly_sigma=1.2,
 flags=0
)

u = flow[..., 0]
v = flow[..., 1]
```

Параметры метода Фарнебэка

poly_n и poly_sigma

Степень сглаживания и размер полигона аппроксимации

winsize

Влияет на устойчивость: большое окно — устойчивее, но менее чувствительно

pyr_scale

Коэффициент уменьшения на каждом уровне пирамиды

Метод хорошо работает на текстурных и среднетекстурных областях. Сильные однородные области дают шумовой поток.

Визуализация поля потока

Плотный поток — двумерное поле векторов, интерпретируемое как смещение каждого пикселя между кадрами.

HSV-кодирование

Hue — направление, Value — скорость

Стрелочная визуализация

На редкой сетке координат

Полутона компонента

Выделение движения нормали к контуру

Визуализация выявляет шумовые области — однородные поверхности или тени, где поток формируется из-за интерполяции. Такие области маскируются порогами по скорости.

Устойчивость метода Фарнебэка

Преимущества

- Сглаживание как часть модели
- Квадратичная аппроксимация вместо точечных производных
- Итеративное уточнение при больших смещениях

Чувствительность к

- Сильному motion blur
- Локальным изменениям яркости
- Блочным артефактам H.264/H.265
- Очень малым текстурам

Плотный поток часто сочетают с фильтрами: медианным, билинейным сглаживанием, bilateral-фильтром или пост-обработкой регуляризацией

Итоги раздела

Построена плотная модель движения, основанная на квадратичной аппроксимации локальных областей и анализе изменения коэффициентов при сдвиге изображения.

1

Полноценное поле

Движение для каждого пикселя

2

Устойчивость

К шумам и реальным видеопотокам

3

Применения

Стабилизация, трекинг, сегментация

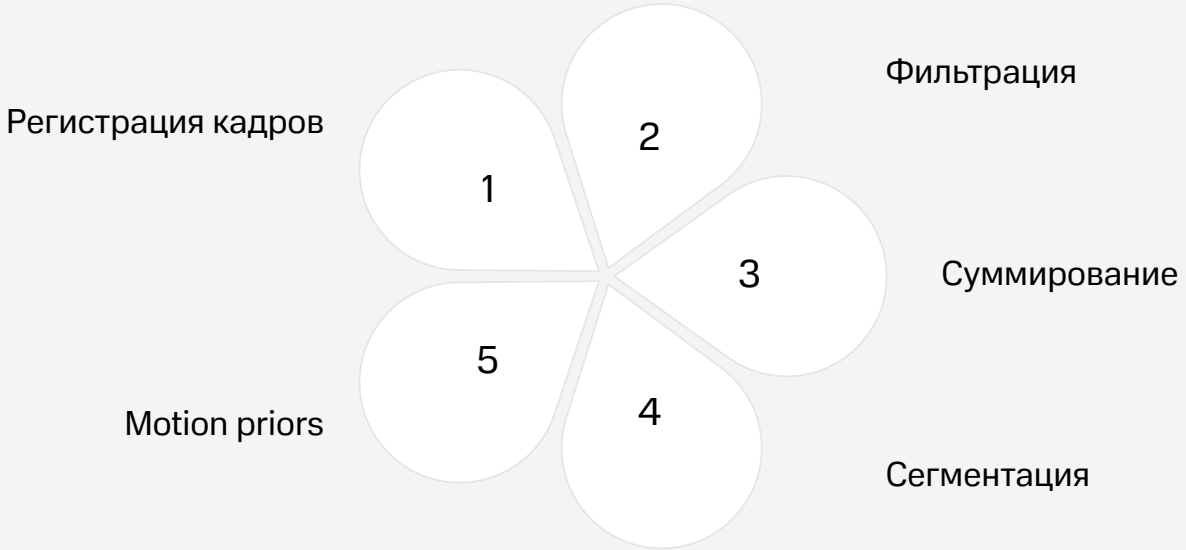
Применения оптического потока

Стабилизация, слежение, сегментация движения

Геометрический мост между кадрами

Оптический поток — оператор соответствия между кадрами, выражающий переход как деформацию координат:

$$I_{t+1}(x, y) \approx I_t(x - u(x, y), y - v(x, y))$$



Поток используется как регуляризатор или грубая геометрическая подсказка в deep learning системах

Стабилизация видео

Ключевая задача — отделение движения камеры от движения сцены. Глобальная компонента аппроксимируется трансформацией:

$$\mathbf{x}' = A\mathbf{x} + \mathbf{t}$$

где A — матрица масштабирования и вращения, $\mathbf{t}$ — трансляция

01

Вычисление потока

Обычно Farnebäck

02

Оценка глобальной модели

Через RANSAC

03

Фильтрация параметров

Низкочастотным фильтром

04

Компенсация движения

Обратным преобразованием

05

Обработка границ

Кроп или inpainting

Трекинг объектов

Если объект выделен на одном кадре, поток даёт возможность предсказать его положение на следующем:

$$\mathbf{x}_{t+1} = \mathbf{x}_t + \mathbf{d}(\mathbf{x}_t)$$

Инициализация

Предсказание позиции объекта до применения детектора

Регуляризация

Корректировка траектории при выпадении детектора

Оценка деформаций

Отслеживание нежёстких объектов

Motion priors

Для SORT, DeepSORT, ByteTrack, BoT-SORT

Сегментация движущихся объектов

Норма вектора потока:

$$M(x, y) = \sqrt{u(x, y)^2 + v(x, y)^2}$$

Выделение областей движения порогированием:

$$\Omega = \{(x, y) : M(x, y) > \tau\}$$

Применения

- Отделение объектов от фона
- Сегментация людей и транспорта
- Выделение областей интереса
- Детекция начала движения

Улучшения

- Морфологическая фильтрация
- Медианное сглаживание
- Подавление малых градиентов
- Комбинирование с фоновыми моделями

Регуляризация потока

Плотные методы создают поле движения, которое может быть шумным. Используется регуляризация как задача оптимизации:

$$E = \iint (I_x u + I_y v + I_t)^2 \, dxdy + \lambda \iint (|\nabla u|^2 + |\nabla v|^2) \, dxdy$$

> Вторая часть функционала заставляет поток быть гладким

Принцип лежит в основе метода Хорна–Шунка и используется в современных вариациях как регуляризатор поверх плотного потока.

Оптический поток в инженерных системах редко используется в «сыром» виде — почти всегда применяется фильтрация или оптимизация

Интеграция в инженерные пайплайны

В робототехнике поток используется совместно с IMU. В видеокомпрессии аналогичен motion vectors. В медицине — анализ деформируемых тканей.

Компенсация движения и warping

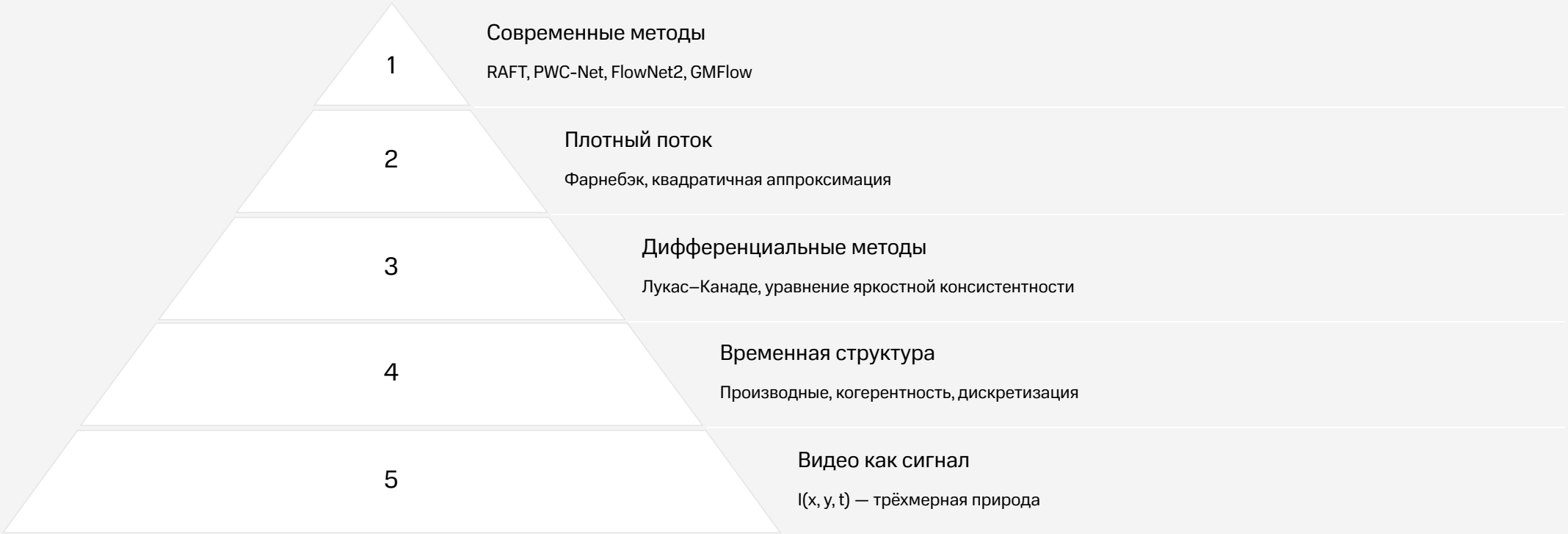
Оптический поток позволяет «переносить» один кадр в координатную систему другого через деформацию изображения:

```
import cv2
import numpy as np

h, w = flow.shape[:2]
grid_x, grid_y = np.meshgrid(np.arange(w), np.arange(h))

map_x = (grid_x + flow[..., 0]).astype(np.float32)
map_y = (grid_y + flow[..., 1]).astype(np.float32)

warped = cv2.remap(prev_frame, map_x, map_y,
                   cv2.INTER_LINEAR)
```

Поток должен быть обратным: для реконструкции кадра t используя кадр t+1, необходим поток dt←t+1

Заключение

Мы построили целостную картину движения как фундаментального аспекта визуального восприятия.



От фиксированных фильтров к обучаемым свёрточным представлениям — следующий этап курса