Дополненная реальность

- Augmented Reality (AR) - хороший пример систем реального времени
- Встраивание в видеопоток синтетических объектов с учетом ракурса съёмки в реальном времени
- Мы на AR можем изучить требования, предъявляемые к системам реального времени, и подходы к разработке алгоритмов

Требования к системе AR

- Время обработки одного кадра: 100-200мс, иначе у нас не будет работы «в реальном времени»
- Надежность:
  - Желательно, без сбоев в течение всего сеанса использования (от нескольких минут до целого дня)
  - Быстрое и автоматическое восстановление после сбоя

Надежность для видеопотока

- Предположим, мы воспользовались алгоритмом отслеживания объектов (контрольных точек)
- Оценим надежность системы
- Пусть вероятность ошибки 0.1% на кадр
- После n кадров, вероятность успеха 0.999^n
- При 30 кадрах/с у нас получается:
  - 3.0% шанс ошибки после 1 сек
  - 83.5% шанс ошибки после 1 минуты
  - 99.99% шанс ошибки после 5 минут

Вероятность ошибки

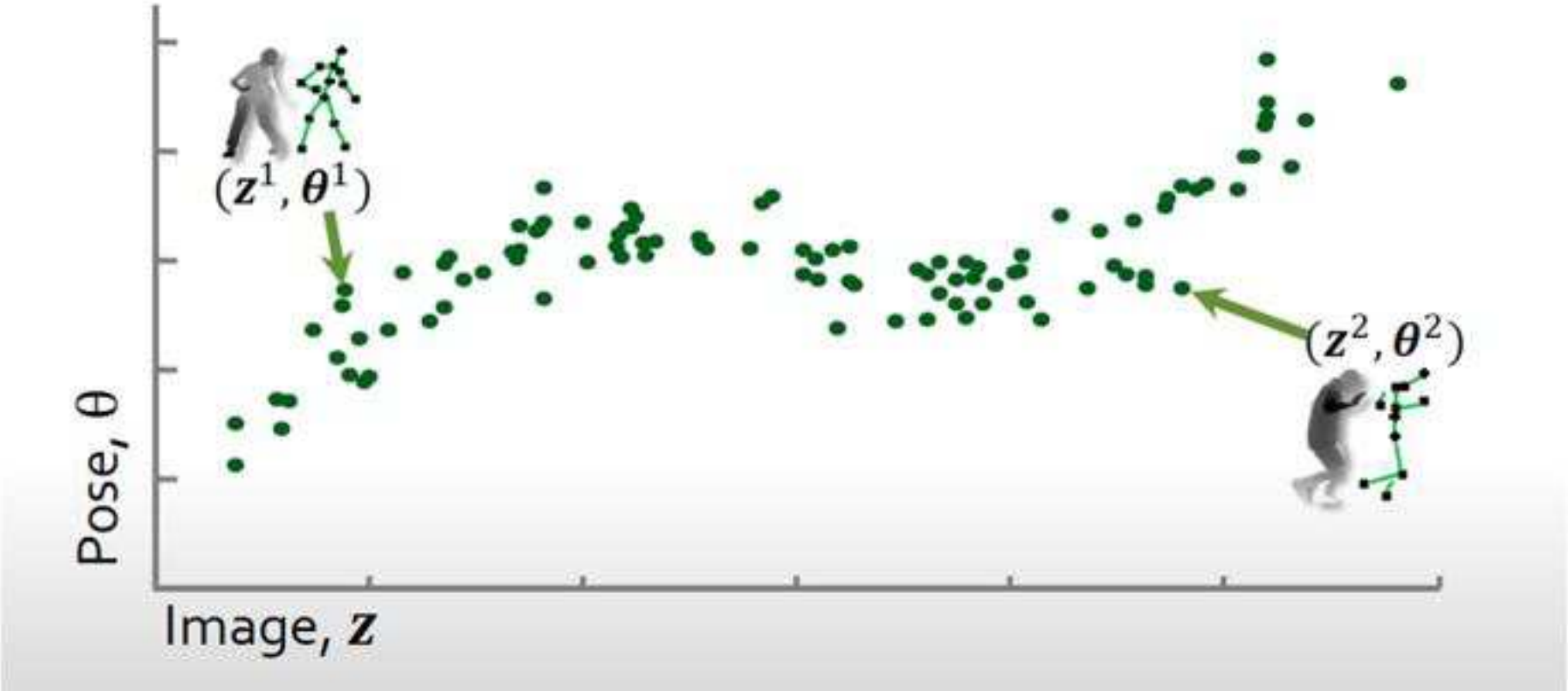
- Пусть вероятность ошибки 0.01% на кадр
- После n кадров, вероятность успеха $0.999n$
- При 30 кадрах/с у нас получается:
  - 0.3 % шанс ошибки после 1 сек
  - 16.5 % шанс ошибки после 1 минуты
  - 59.3 % шанс ошибки после 5 минут

Выводы из примера

- Нужен метод, решающий задачу AR по одному кадру
  - Или на маленьком наборе кадров
  - «Tracking by detection»
  - Даже если вероятность ошибки 10% - из 30 кадров на 27 кадрах система правильно сработает

- Как быть со временной информацией (видео)?
  - Нельзя уменьшать пространство поиска, можем пропустить правильные решения
  - Нужно её использовать для временной фильтрации – обработки результатов поиска по нескольким кадрам в совокупности
  - С помощью фильтрации можем отбрасывать ложные гипотезы и разрешать неоднозначные ситуации

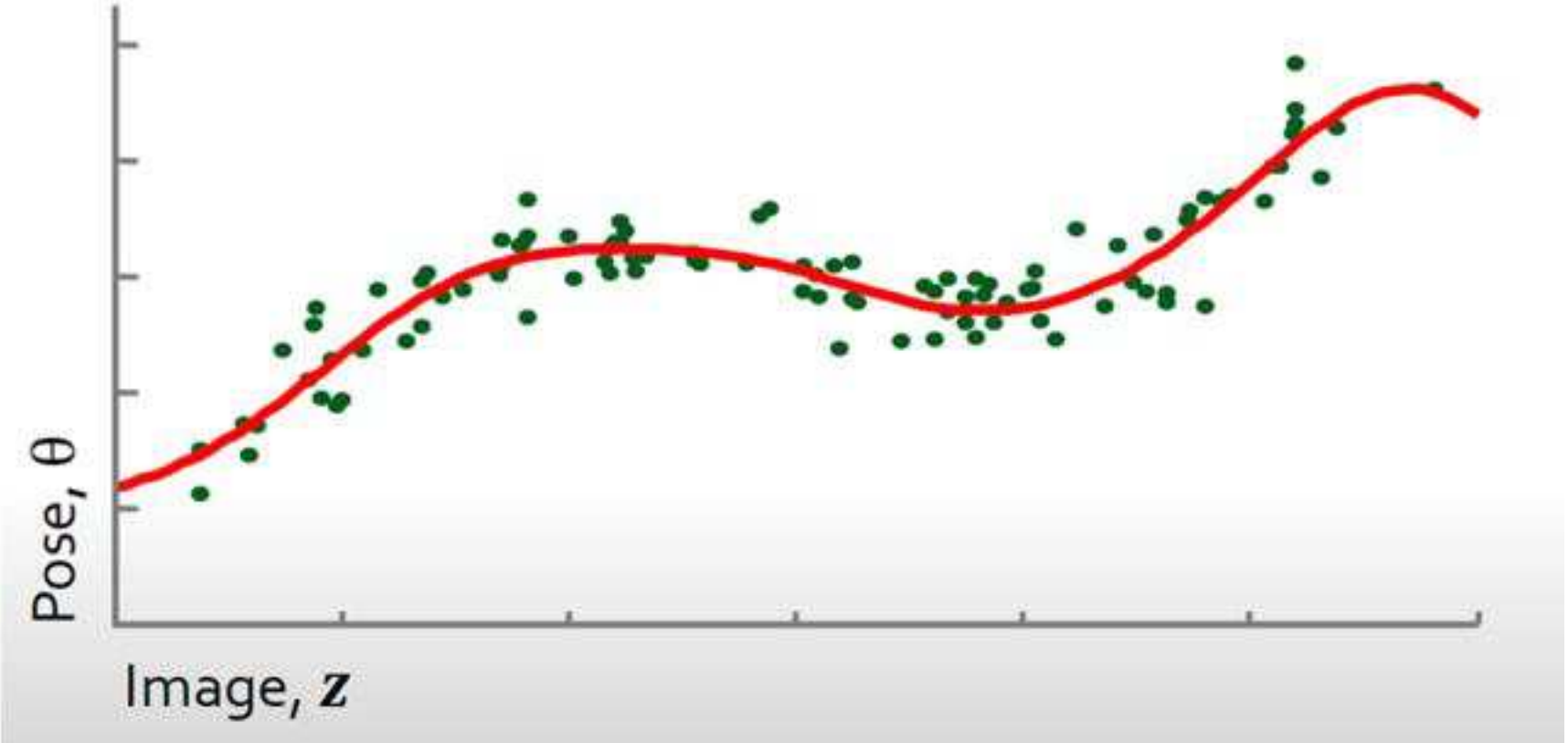
Пример



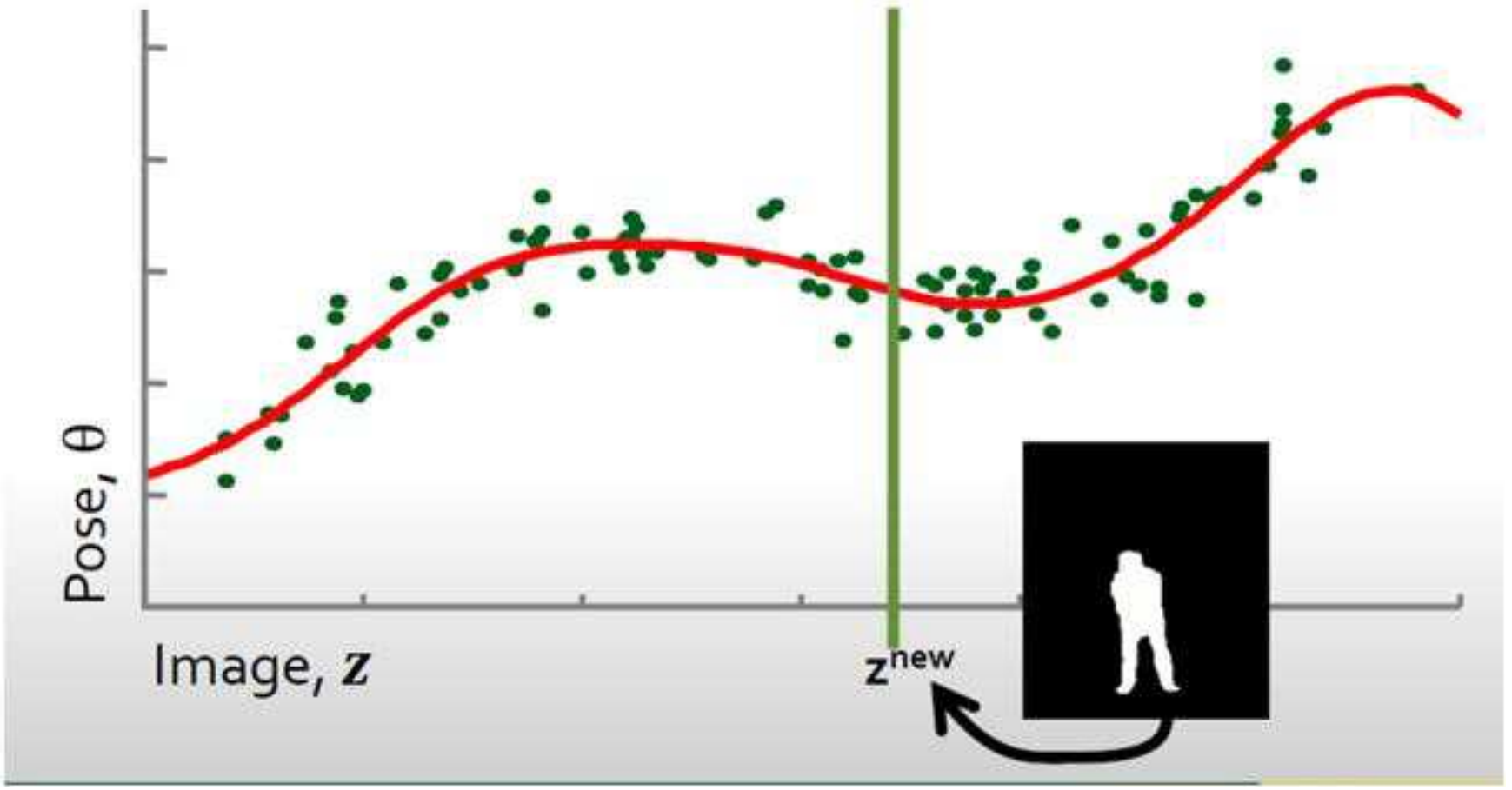
Определение позы человека $\boldsymbol{\theta}$ по изображению $\mathbf{z}$

Пример из лекции Andrew Fitzgibbon с лекции на Microsoft Computer Vision Summer School 2011

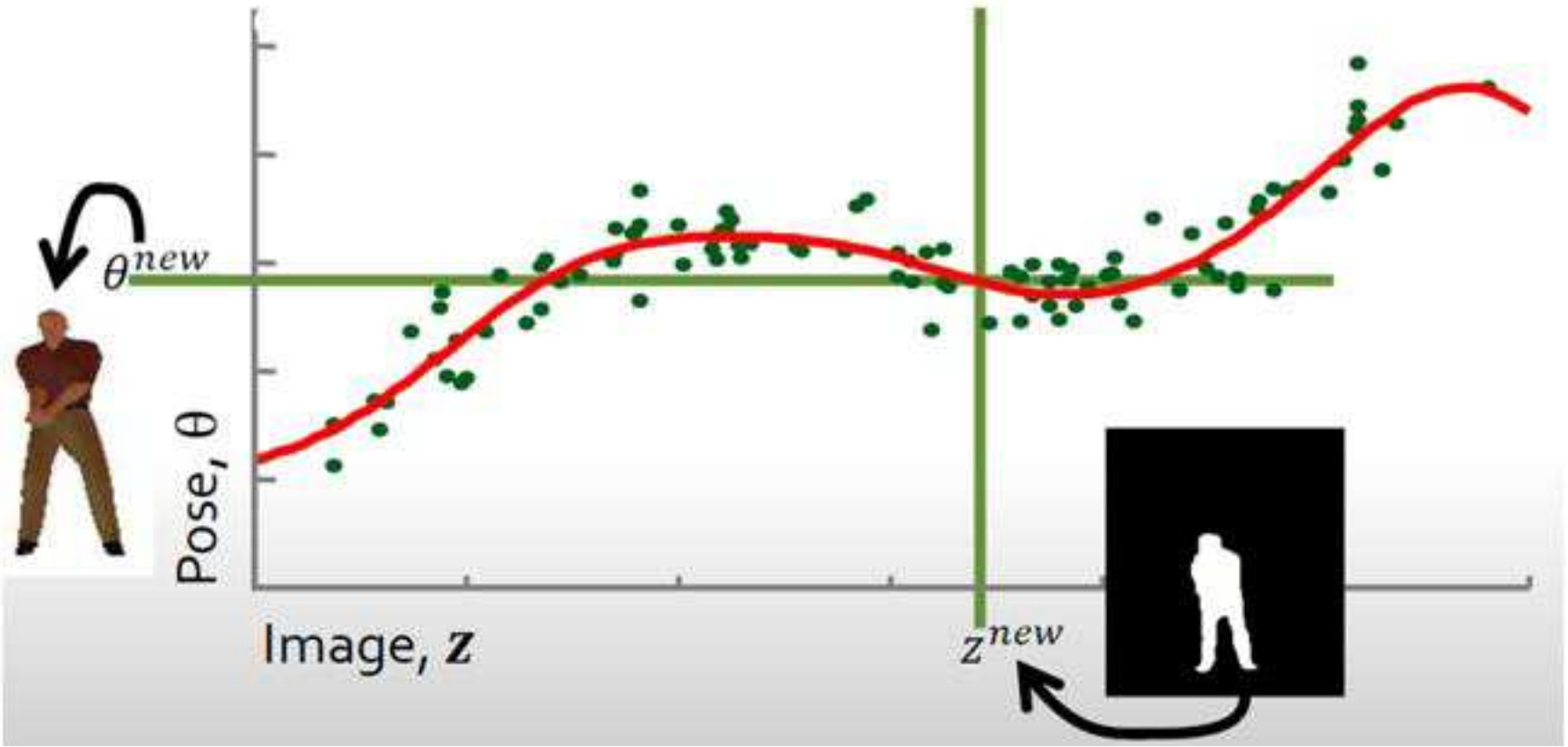
Пример



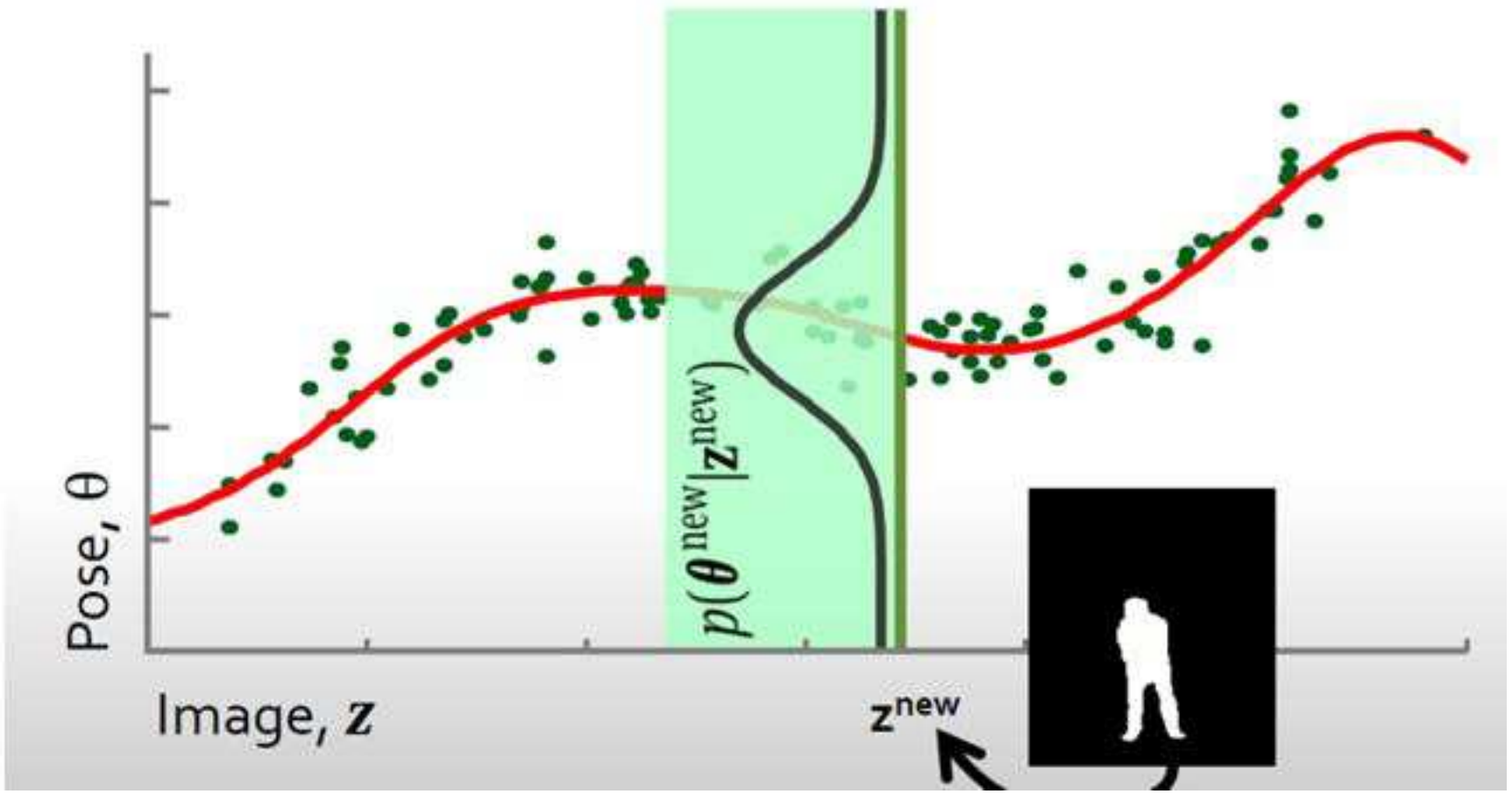
Пример



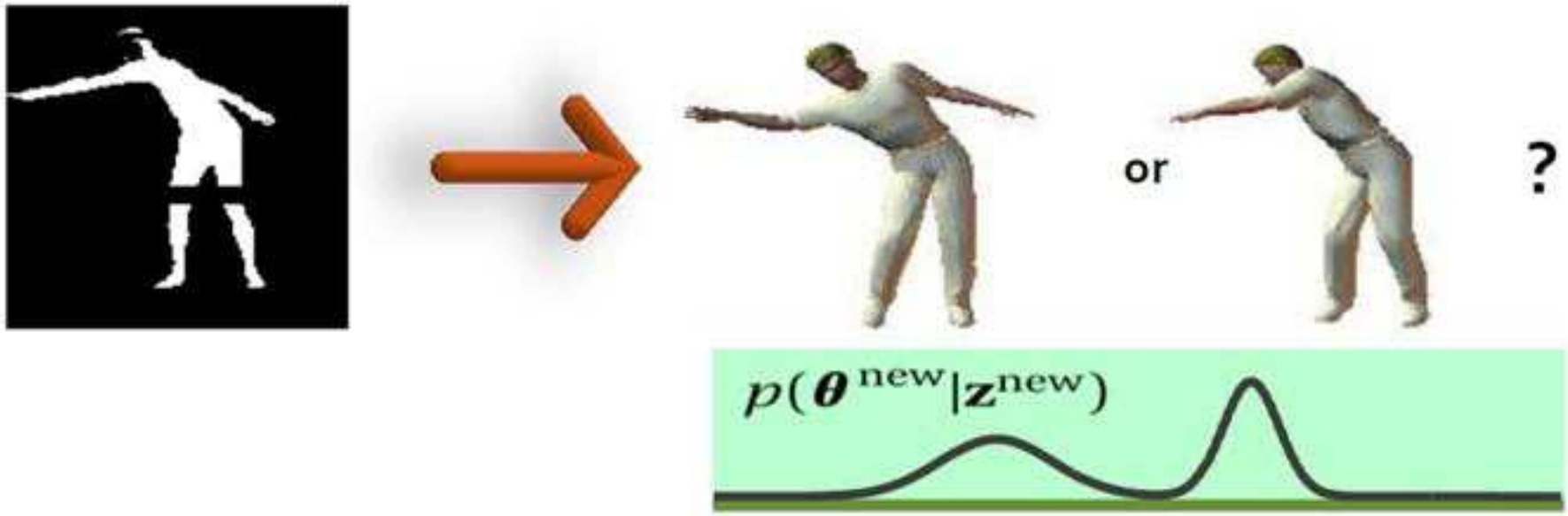
Пример



Пример

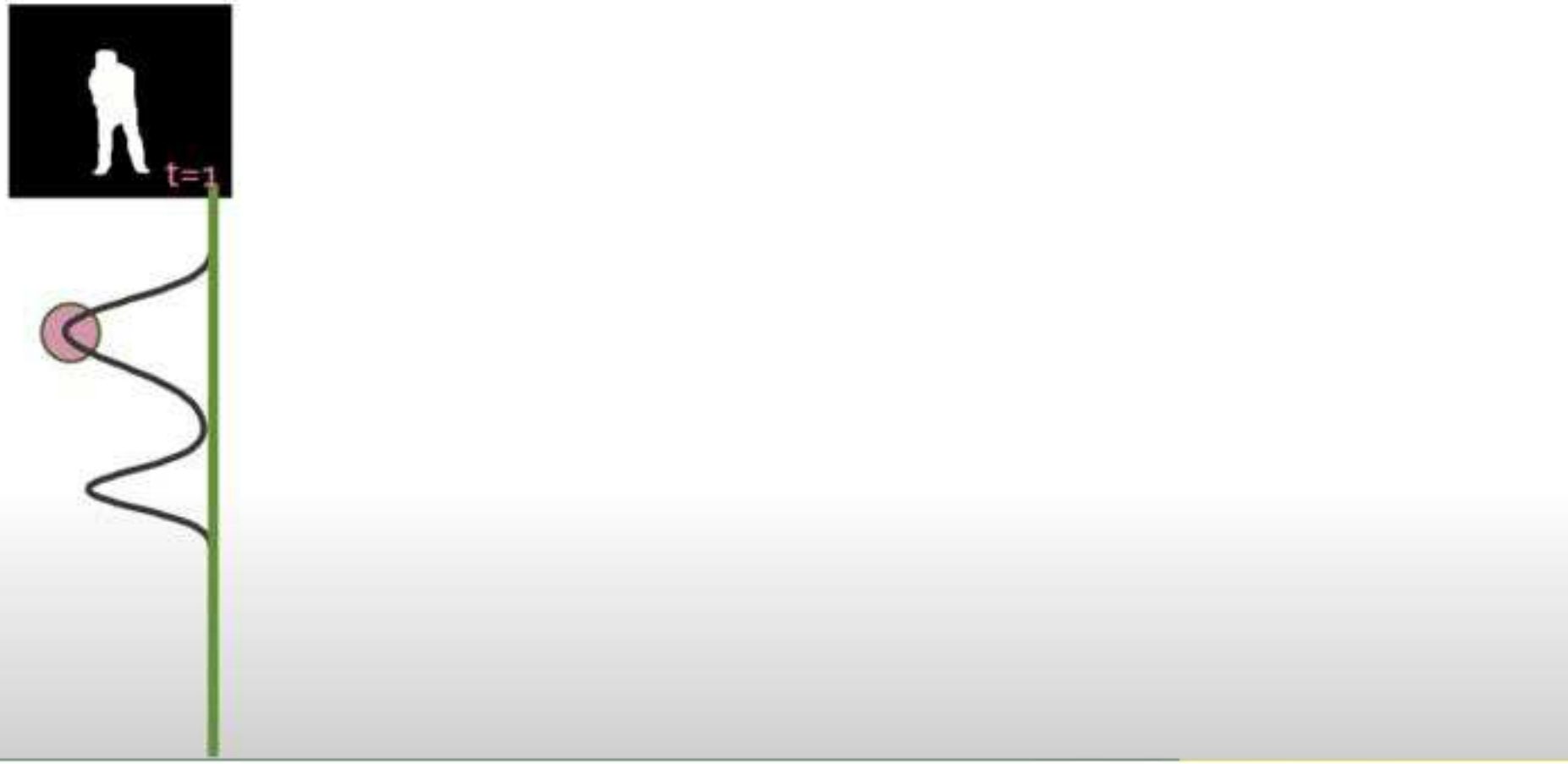

Pose, θ
Image, z
$p(\boldsymbol{\theta}^{\text{new}}|\mathbf{z}^{\text{new}})$
$\mathbf{z}^{\text{new}}$

Пример

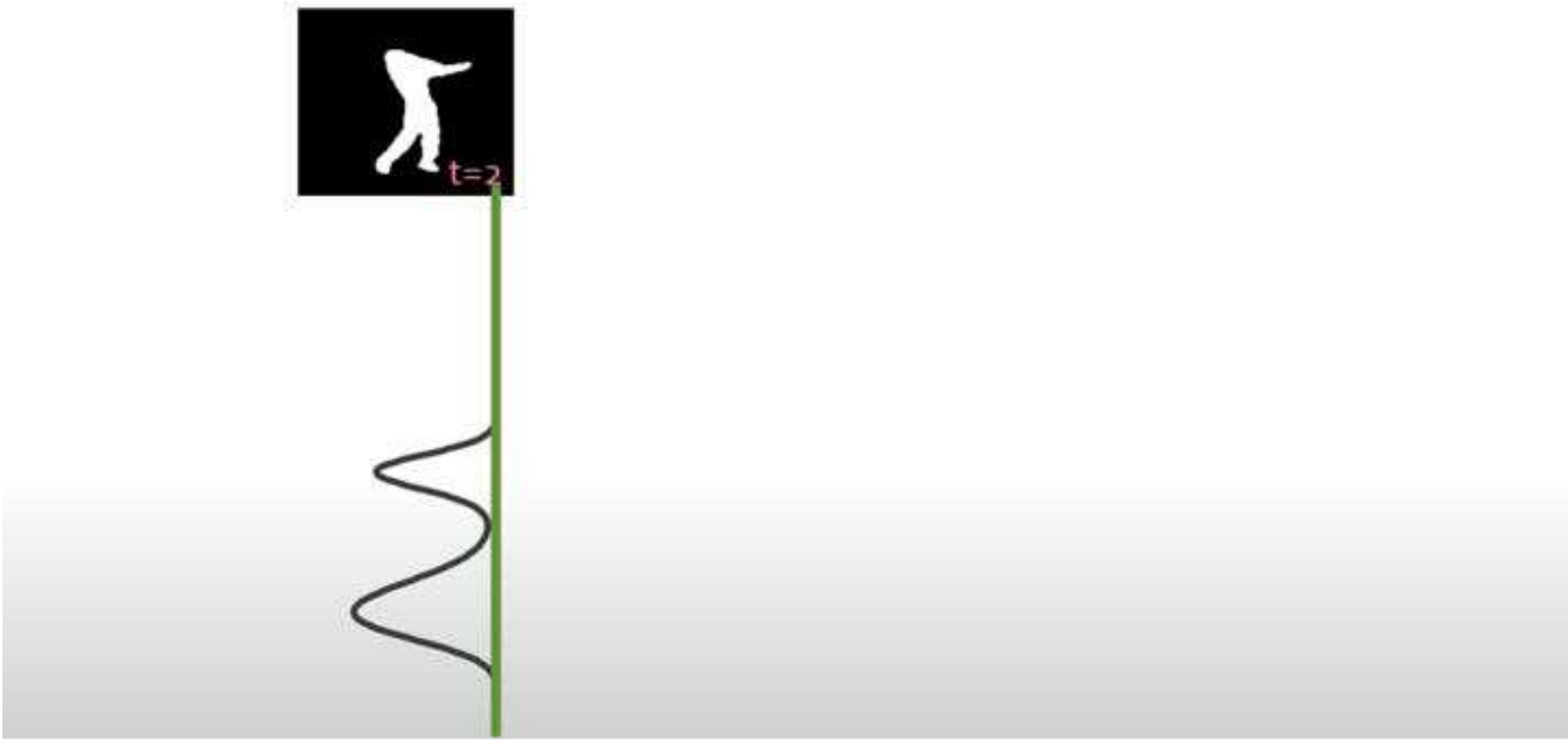


Пример

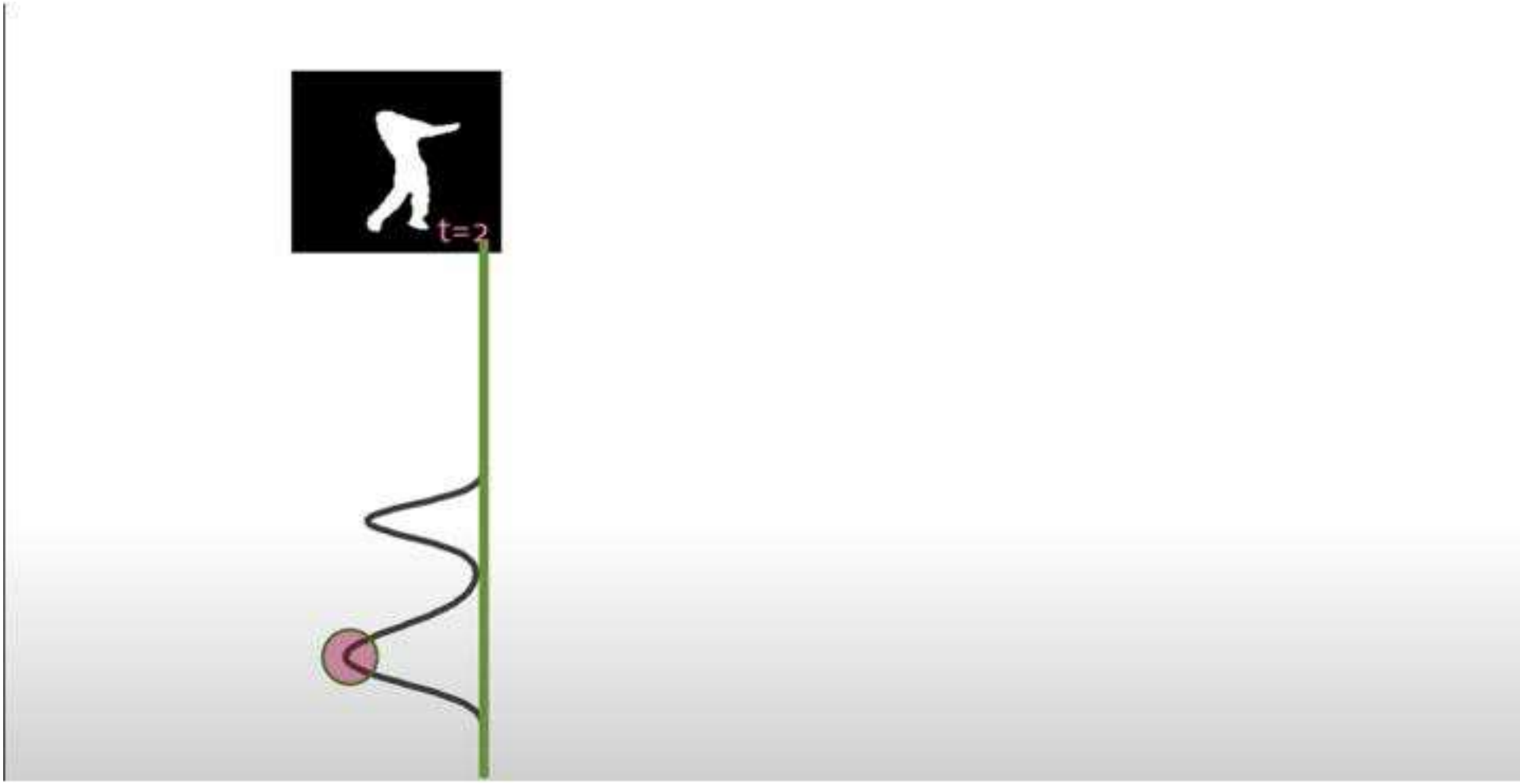
Пример



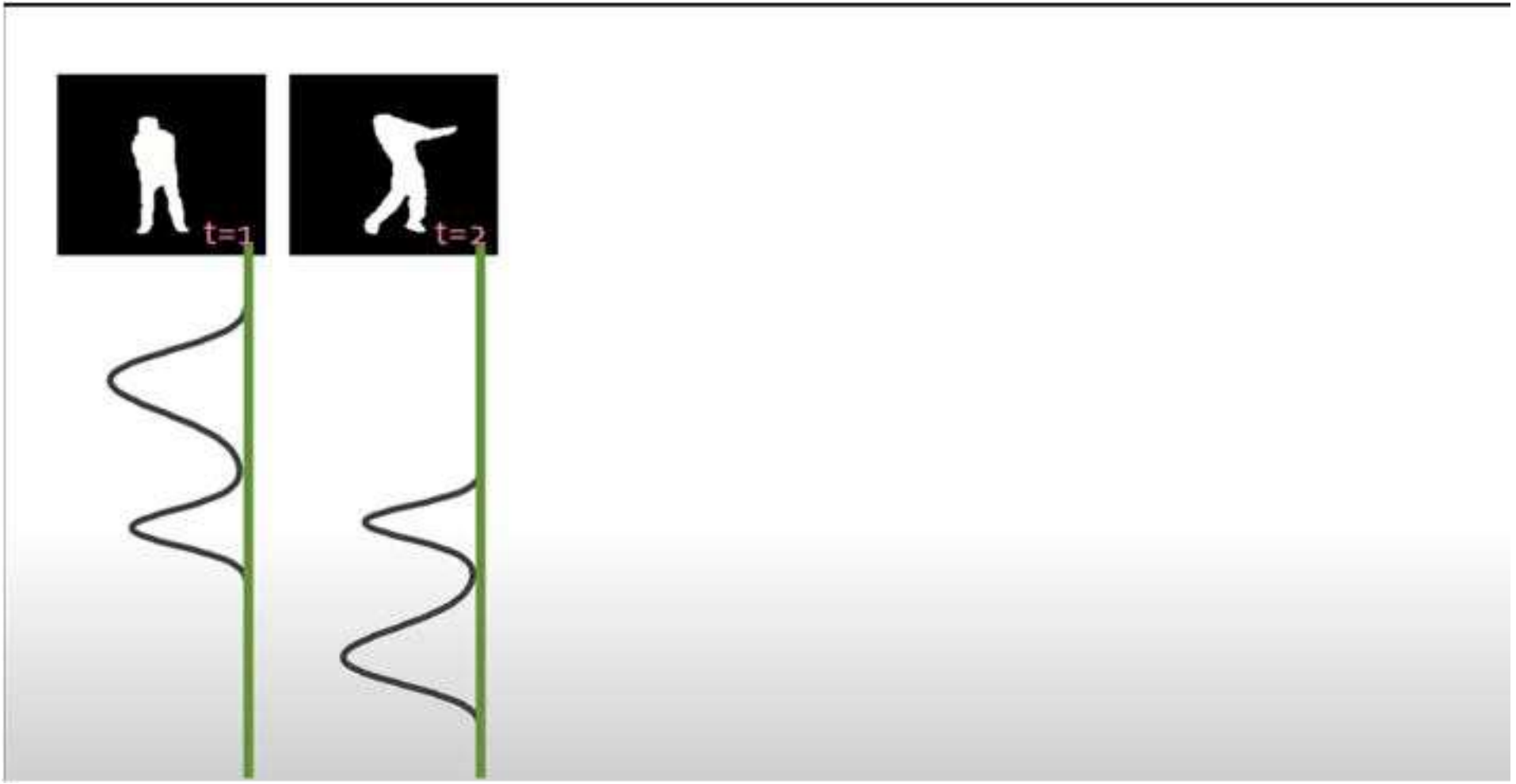
Пример



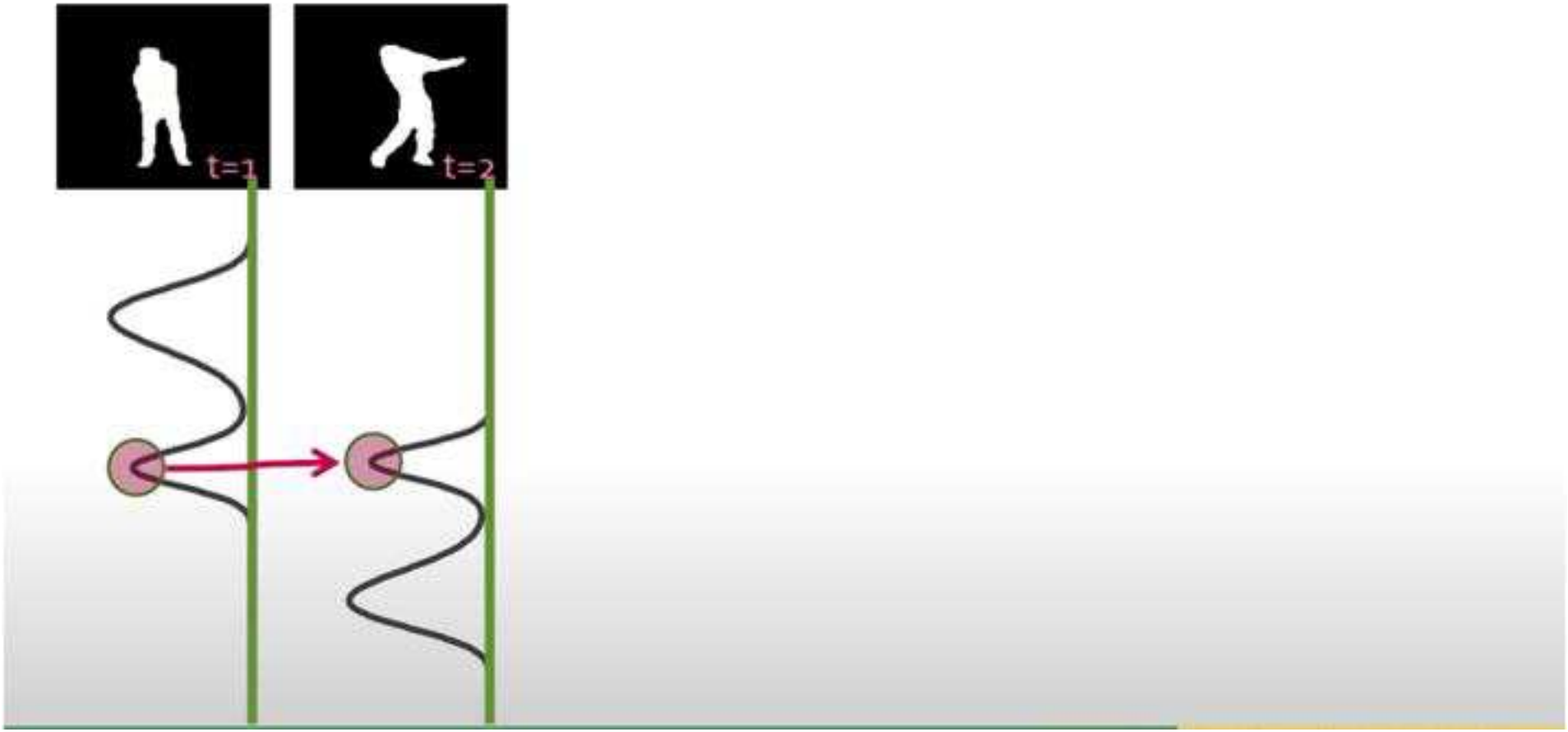
Пример



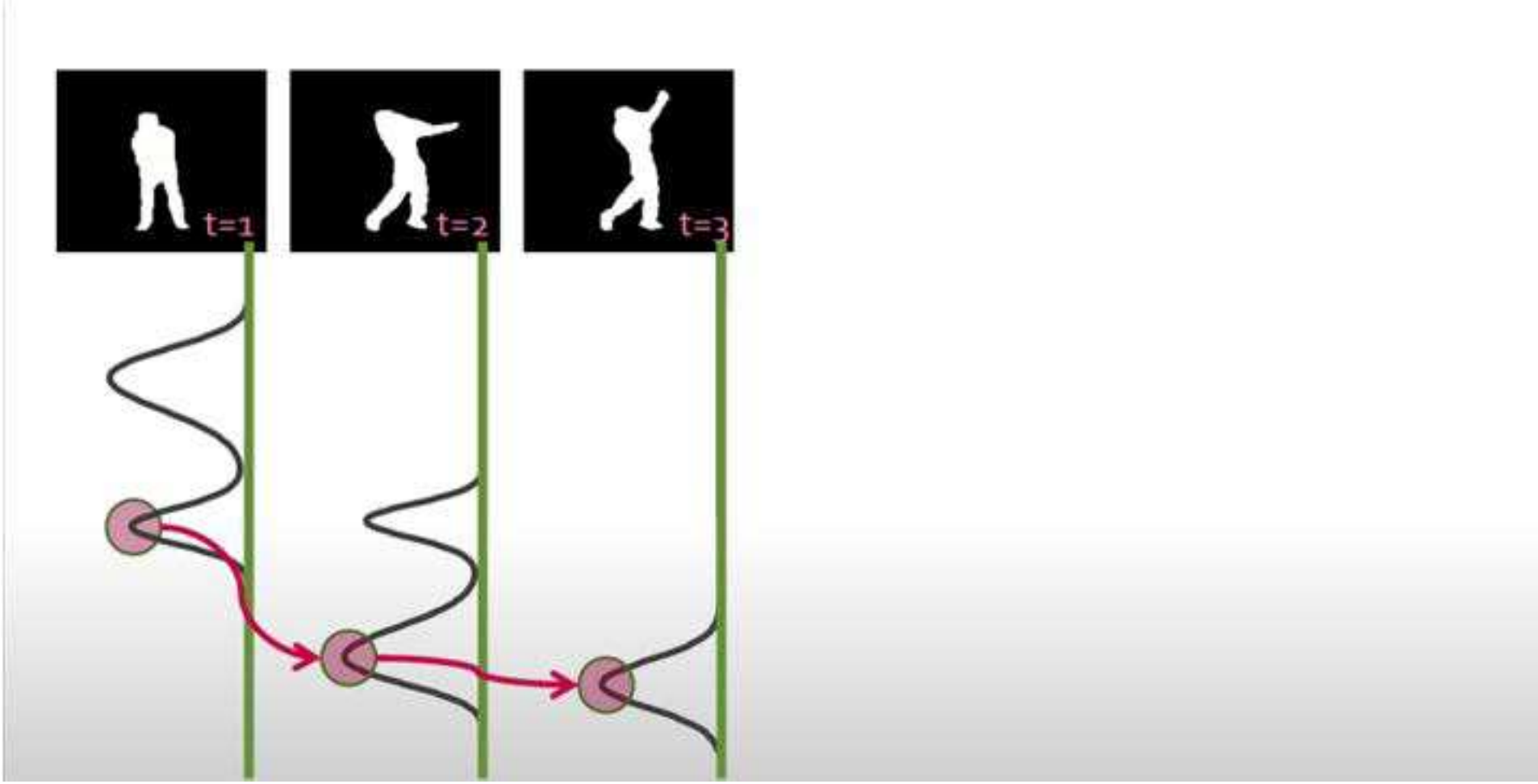
Пример



Пример



Пример



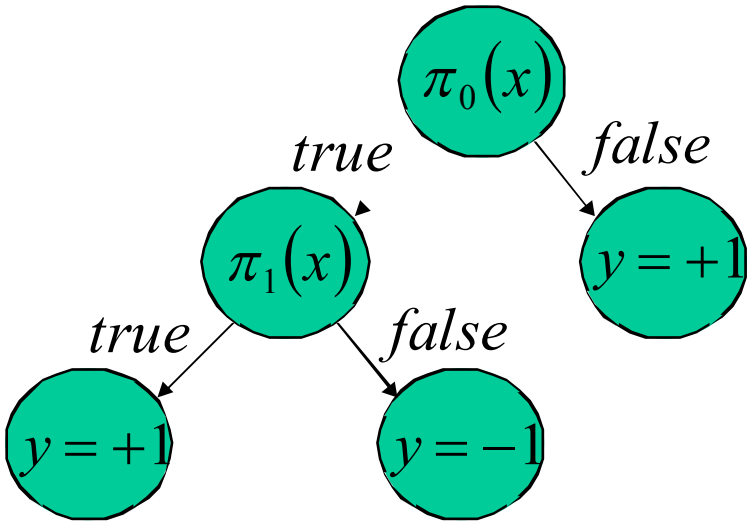
Временная фильтрация

- Методы временной фильтрации выходят за пределы нашего курса
- Скорее, задача из области оптимального управления

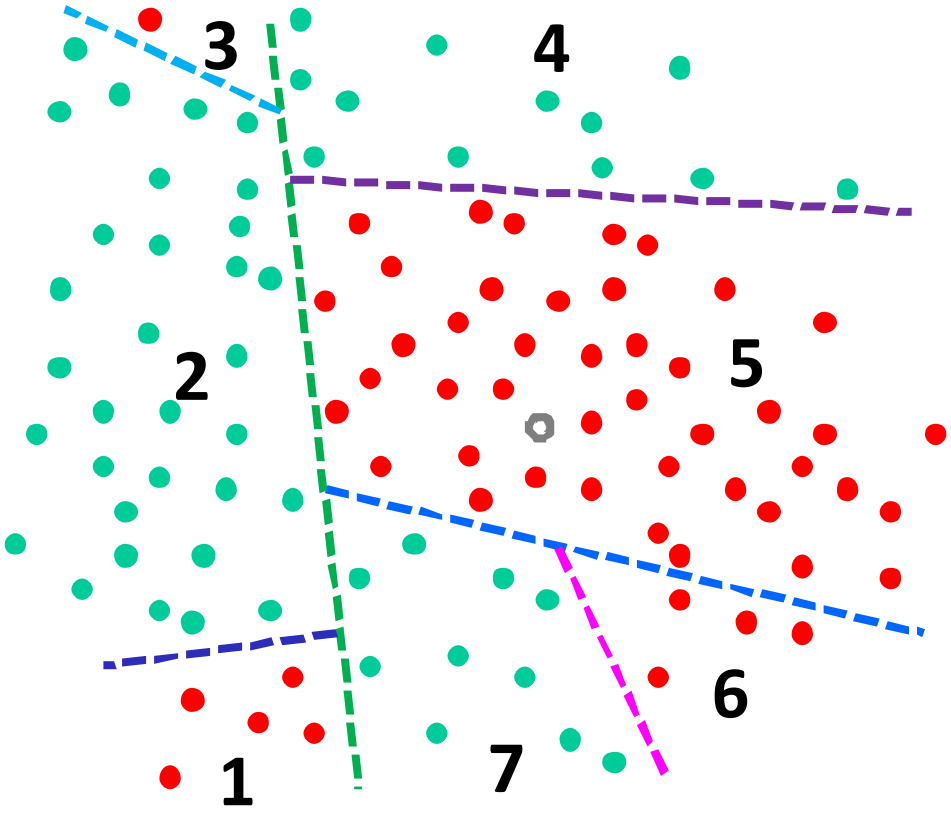
- Пример:
  - Kalman filter
  - Particle filter
  - Hidden Markov Model
  - Gaussian process
  - И т.д.

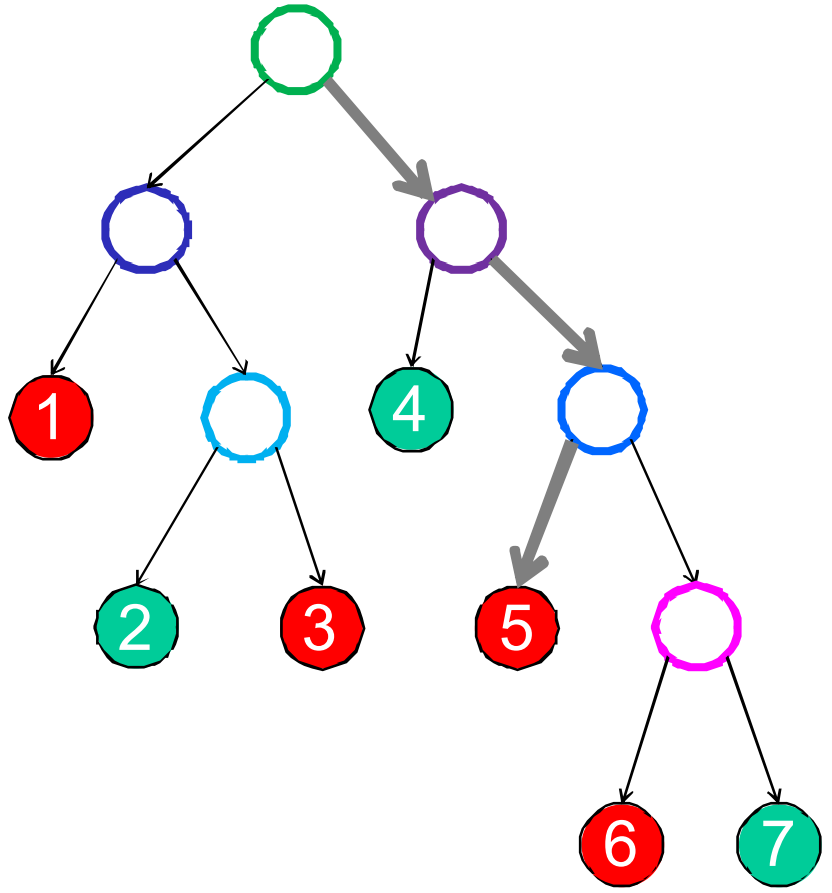
Решающие деревья

- Classification trees
- Двоичное дерево
- Узлы:
  - Помечены некоторым предикатом $\pi : X \to bool$
- Связи:
  - Помечены $\left\{ \begin{array}{l} true(1) \\ false(0) \end{array} \right\}$
- Листья:
  - Помечены ответами из Y



Пример решающего дерева





Slide by Victor Lempitsky

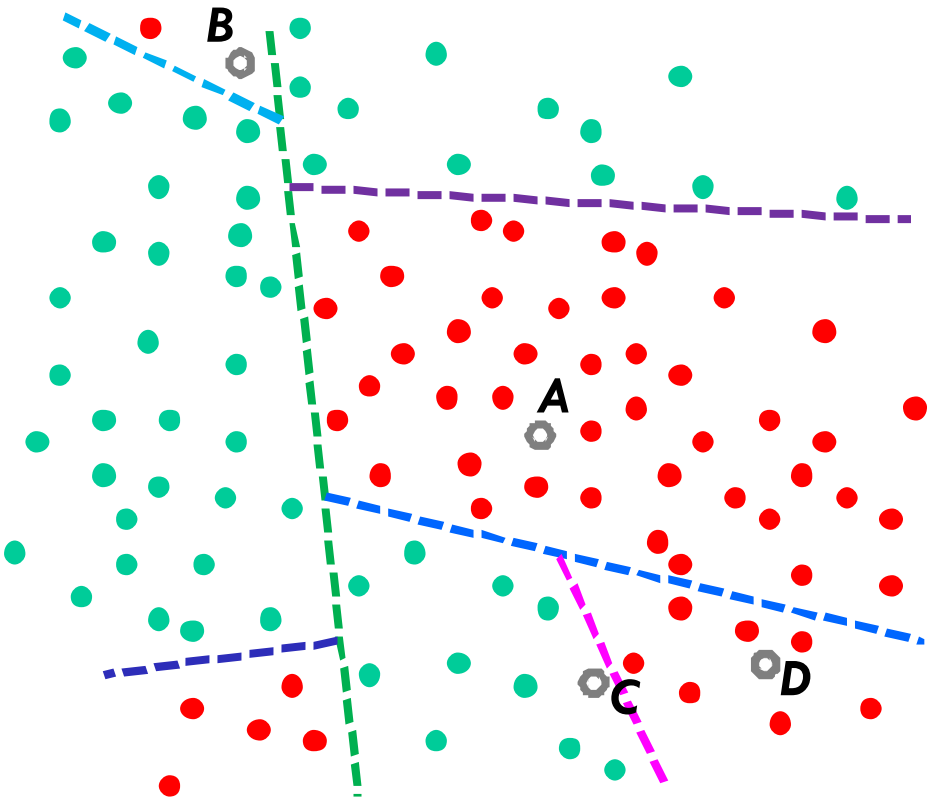
Обучение дерева решений

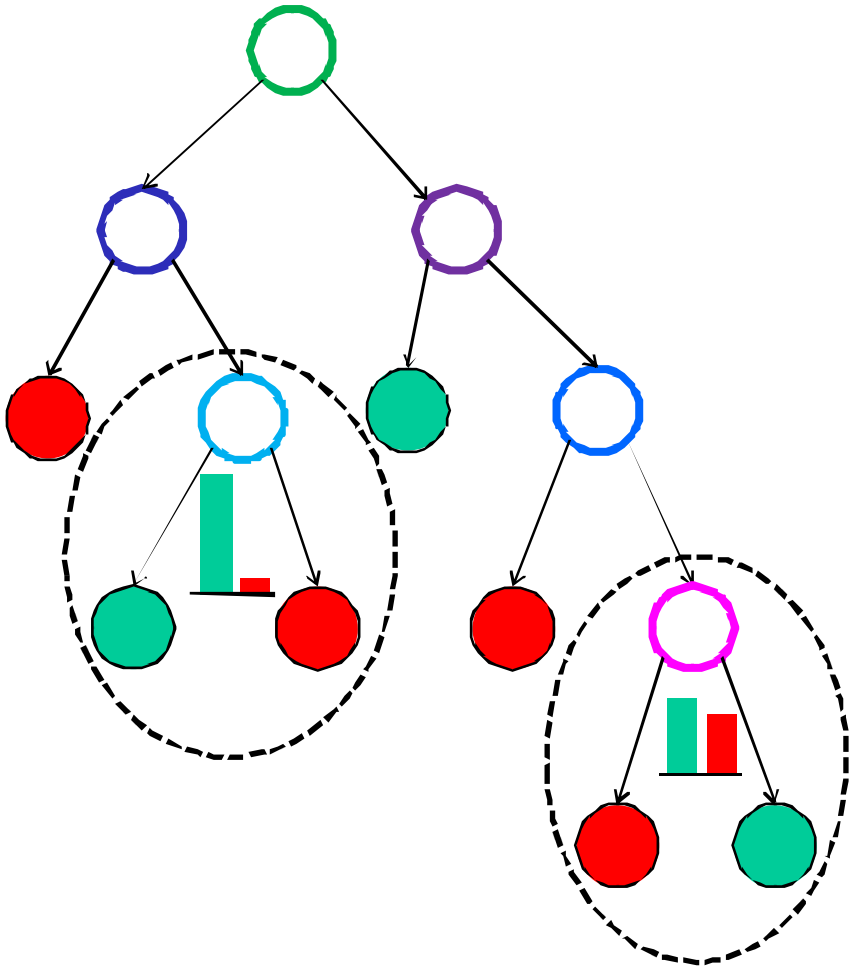
```
function Node = Обучение_Вершины( {(x,y)} )        {
      if {y} одинаковые
            return Создать_Лист(y);
      test = Выбрать_лучшее_разбиение( {(x,y)} );
      {(x0,y0)} = {(x,y) | test(x) = 0};
      {(x1,y1)} = {(x,y) | test(x) = 1};
      LeftChild = Обучение_Вершины( {(x0,y0)} );
      RightChild = Обучение_Вершины( {(x1,y1)} );
      return Создать_Вершину(test, LeftChild, RightChild);
}

//Обучение дерева
function main()                {
      {(X,Y)} = Прочитать_Обучающие_Данные();
      TreeRoot = Обучение_Вершины( {(X,Y)} );
}
```

Slide by Victor Lempitsky

Переобучение и обрезка дерева





Slide by Victor Lempitsky

Свойства решающих деревьев

- Плюсы
  - \+ Просто и наглядно
  - \+ Легко анализируемо
  - \+ Быстро работает
  - \+ Легко применяется для задач со множеством классов и к регрессии
- Минусы
  - – Чувствительны к выбросам в обучающей выборке
  - – Низкая обобщающая способность (предсказание)
  - – Требуют сложных алгоритмов «обрезания» для борьбы с переобучением

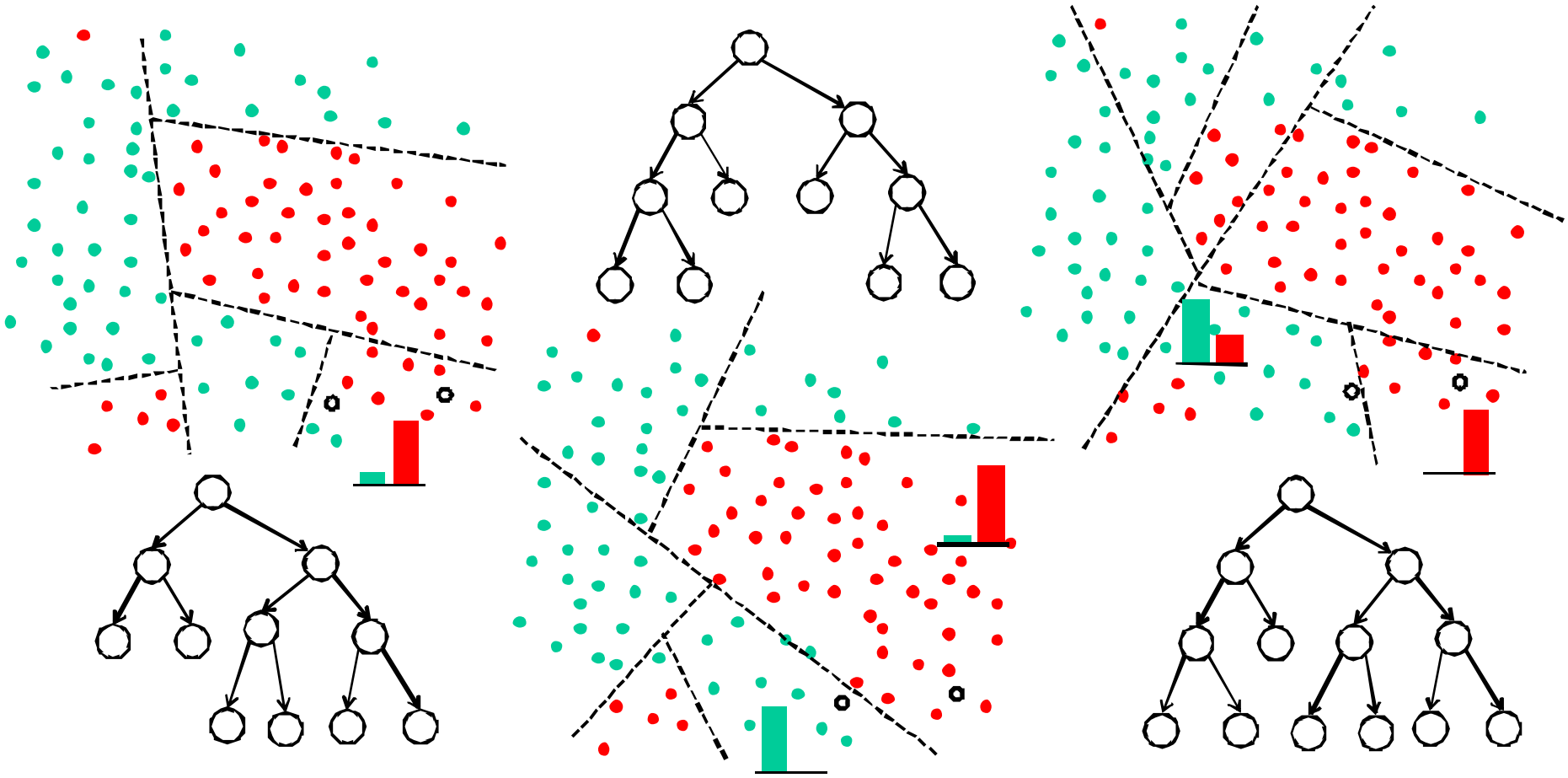
Комитетные методы

- Если взять множество правил (экспертов), с некоррелированной ошибкой (ошибаются в разных местах), то их комбинация может быть работать во много раз лучше

- Такие методы называются ***комитетными***

- ***Бустинг*** – один из примеров комитетных методов, но и на базе деревьев мы можем построить комитет

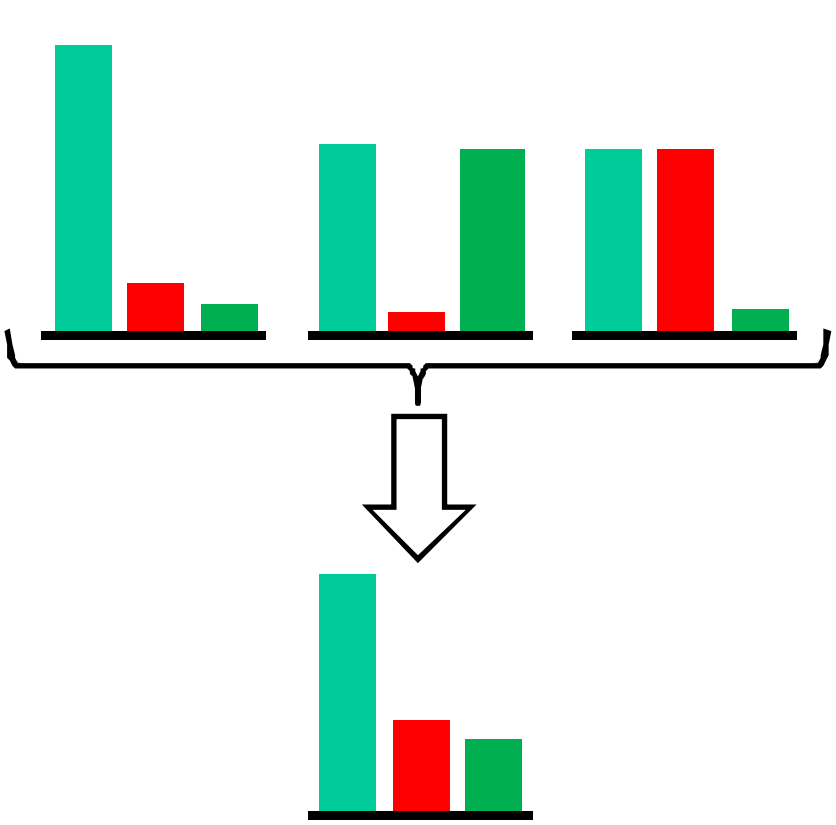
От дерева к лесу

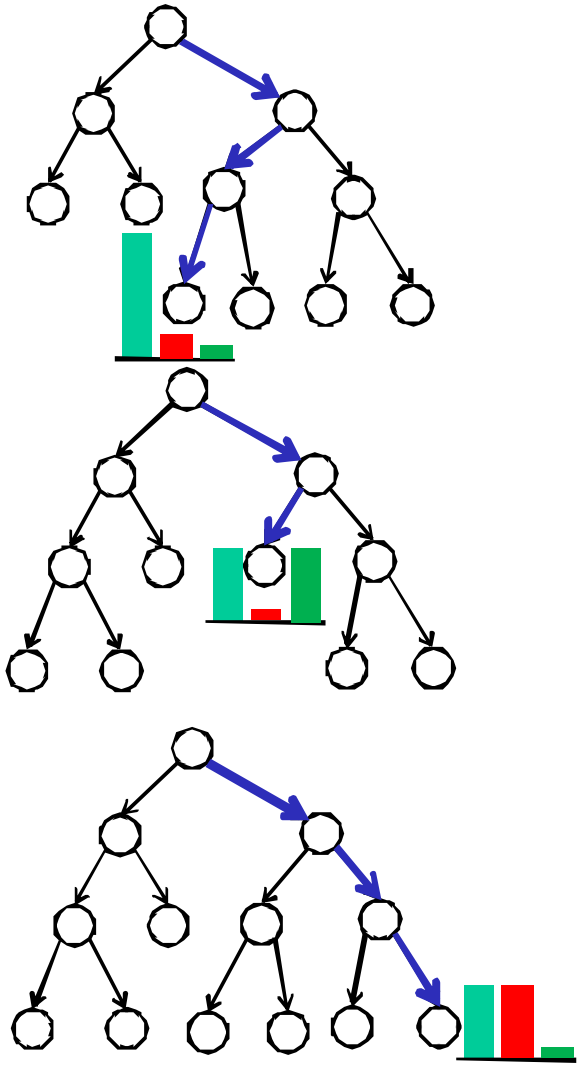


1. Yali Amit, Donald Geman: *Shape quantization and recognition with randomized trees.* Neural Computation, 1997.
2. Leo Breiman: *Random forests.* Machine Learning, 2001.

Slide by Victor Lempitsky

Решающий лес - применение



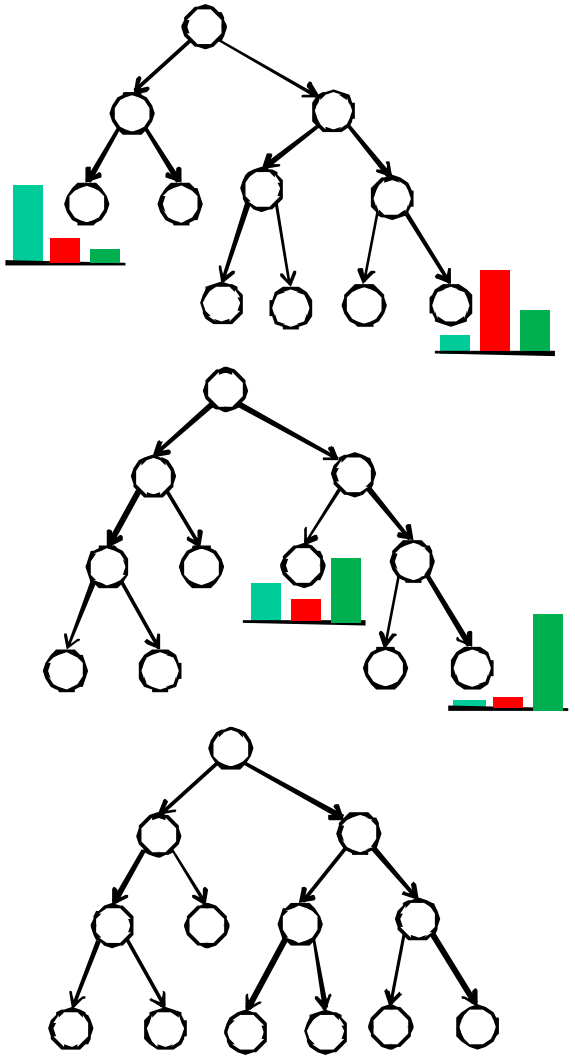


Slide by Victor Lempitsky

Решающий лес - обучение

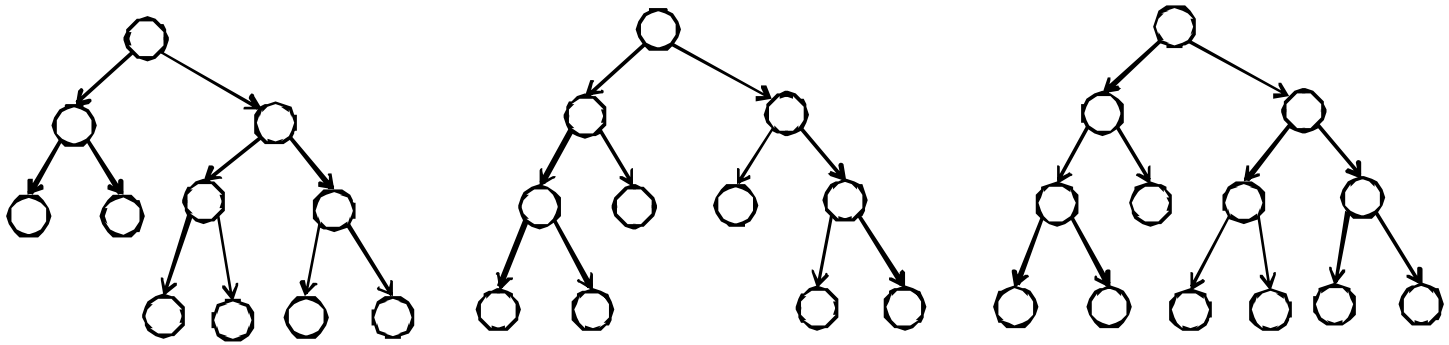
```
function Node = Обучение_Вершины( {(x,y)}, Level)      {
        if {y} одинаковые или Level == maxLevel
                return Создать_Лист(Распределение y);
        {tests} = Создать_N_Случайных_Разбиений({(x,y)},N);
        test = Выбрать_лучшее_разбиение_из({tests});
        {(x0,y0)} = {(x,y) | test(x) = 0};
        {(x1,y1)} = {(x,y) | test(x) = 1};
        LeftChild = Обучение_Вершины( {(x0,y0)}, Level+1);
        RightChild = Обучение_Вершины( {(x1,y1)}, Level+1);
        return Создать_Вершину(test, LeftChild, RightChild);
}

//Обучение леса
function main()   {
        {X,Y} = Прочитать_Обучающие_Данные();
        for i = 1 to N
                {Xi,Yi} = Случайное_Подмножество({X,Y});
                TreeRoot_i = Обучение_Вершины({Xi,Yi});
        end
}
```



Slide by Victor Lempitsky

Решающий лес (Random Forest)



1. Один из самых эффективных алгоритмов классификации
2. Вероятностное распределение на выходе
3. Применим для высоких размерностей пространства признаков
4. Высокая скорость обучения и тестирования
5. Относительная простота реализации
6. Может занимать много памяти при большой глубине дерева

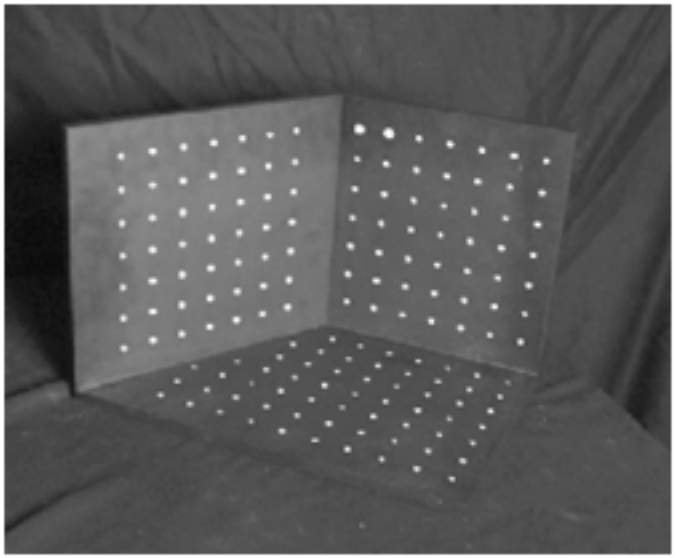
Caruana, R., Niculescu-Mizil, A.: An empirical comparison of supervised learning algorithms, 2006

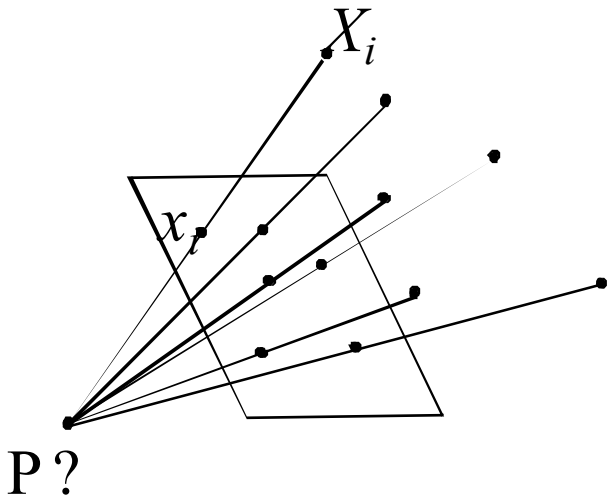
Slide by Victor Lempitsky

Типичное приложение AR

- Общая схема:
  - Выделяем объект сцены, к которому нужно привязывать синтетику
  - Определяем ракурс камеры («регистрация камеры»)
  - Встраиваем синтетический объект

Регистрация камеры





- Если известны несколько пар соответствий 2D и 3D точек, то можно зарегистрировать камеру
- Методы подробно рассматриваются во второй части курса
- Надо сфокусироваться на быстром и надежном выделении и отслеживании объектов и точек

Отслеживание

- Идея
  - Отслеживание объекта через сопоставление ключевых точек между изображениями
  - Задачу сопоставления ключевых точек для заданного объекта можно представить как задачу классификации ключевых точек

V. Lepetit, P. Lagger, and P. Fua, Randomized Trees for Real-Time Keypoint Recognition. CVPR 2005

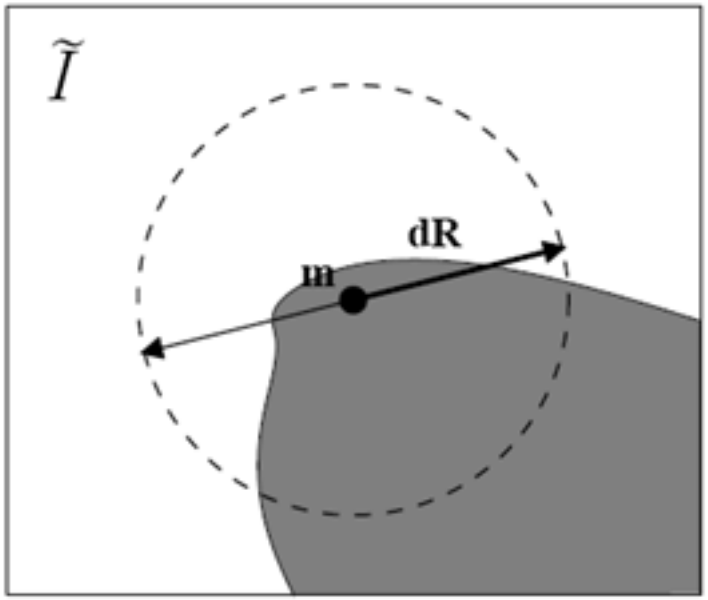
Схема

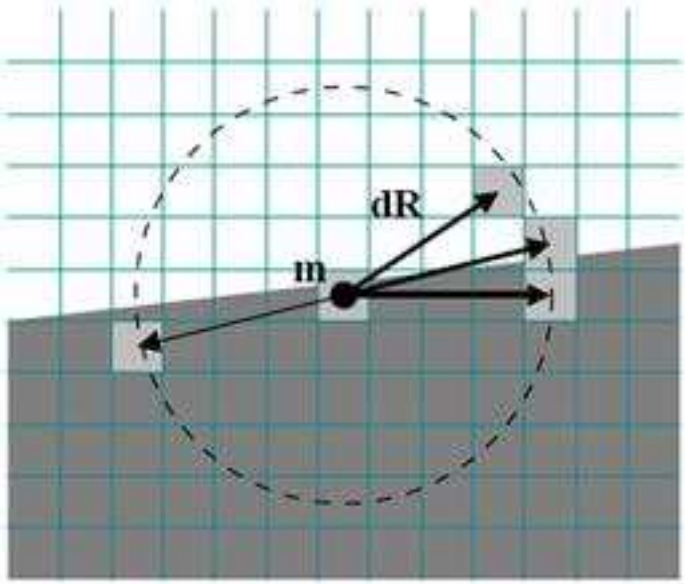
- Идея:
  - N точек с номерами от 1 до N
  - Сделаем классификатор, который ставит номер {0,N}

- Обучим классификатор ключевых точек
  - Возьмём исходное изображение
  - Найдём на нём ключевые точки
  - Синтезируем обучающую выборку патчей по ним
  - Отберём наиболее надежные точки
  - Обучим классификатор

- Слежение
  - Найдём ключевые точки
  - Классифициируем их
  - Зарегистрируем камеру по ключевым точкам

Yet Another Keypoint Detector (YAKT)





Сканируем окружность вокруг точки, проверяя условие – является ли точка ключевой:

$$\begin{array}{lrcll} \text{If} & |\tilde{I}(\mathbf{m}) - \tilde{I}(\mathbf{m}+\mathbf{dR}_\alpha)| & \leq & +\tau & \text{and} \\ \text{if} & |\tilde{I}(\mathbf{m}) - \tilde{I}(\mathbf{m}-\mathbf{dR}_\alpha)| & \leq & +\tau & \text{then } \mathbf{m} \text{ is not a keypoint,} \end{array}$$

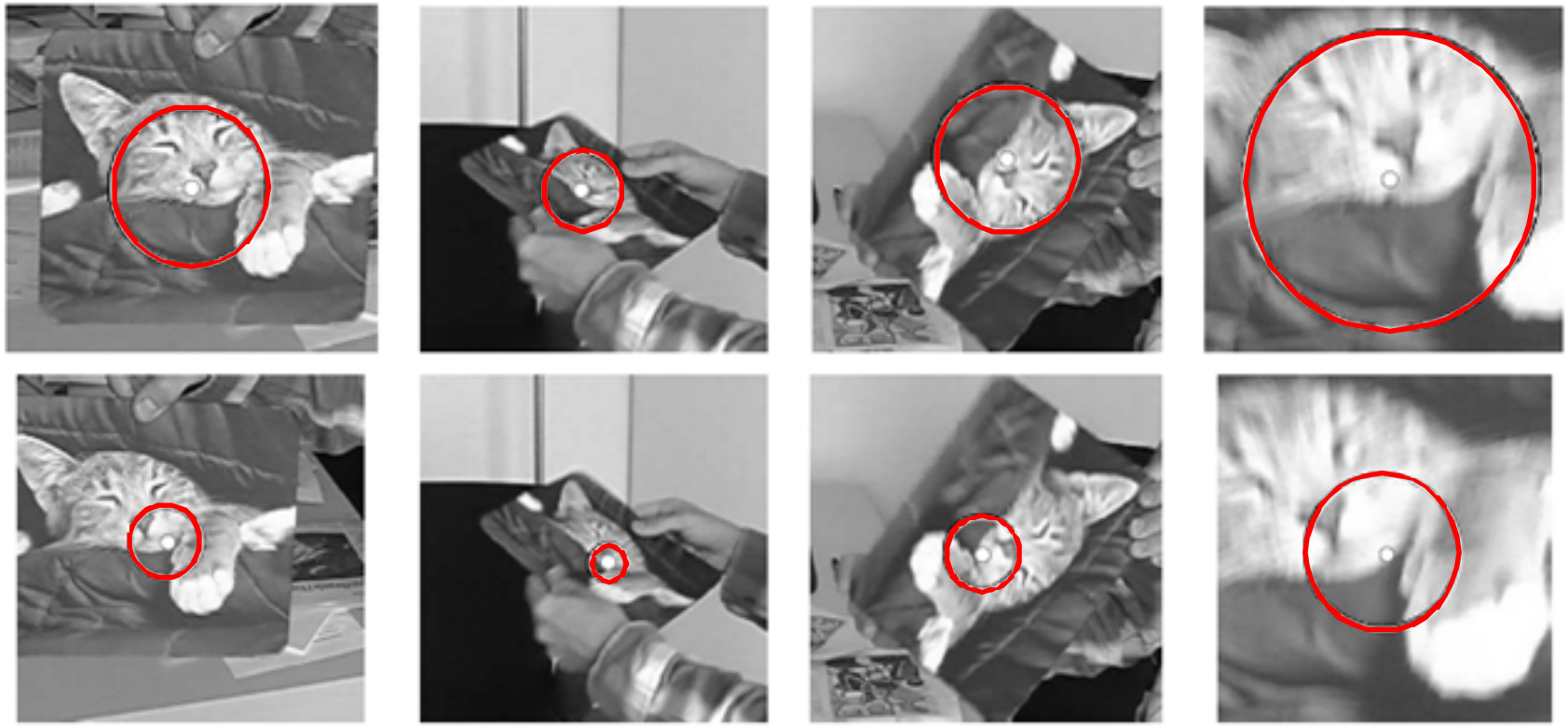
$$\mathrm{LoG}(\mathbf{m}) \approx \sum_{\alpha\in[0;\pi[} \tilde{I}(\mathbf{m}-\mathbf{dR}_\alpha) - \tilde{I}(\mathbf{m}) + \tilde{I}(\mathbf{m}+\mathbf{dR}_\alpha)$$

- аппроксимация LoG для оценки характерного масштаба

$$\alpha_{\mathbf{m}} = \underset{\alpha\in[0;2\pi]}{\operatorname{argmax}} |\tilde{I}(\mathbf{m}) - \tilde{I}(\mathbf{m}+\mathbf{dR}_\alpha)|.$$

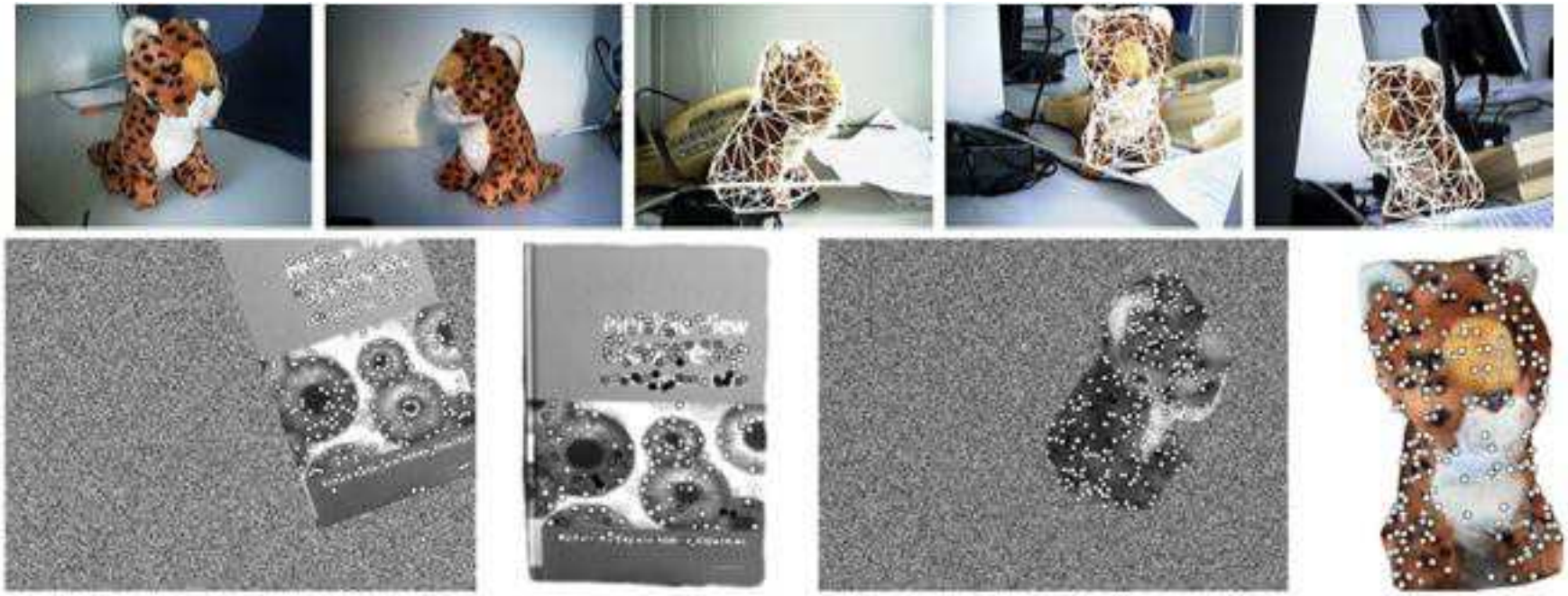
- оценка ориентации фрагмента

YAKT - результаты



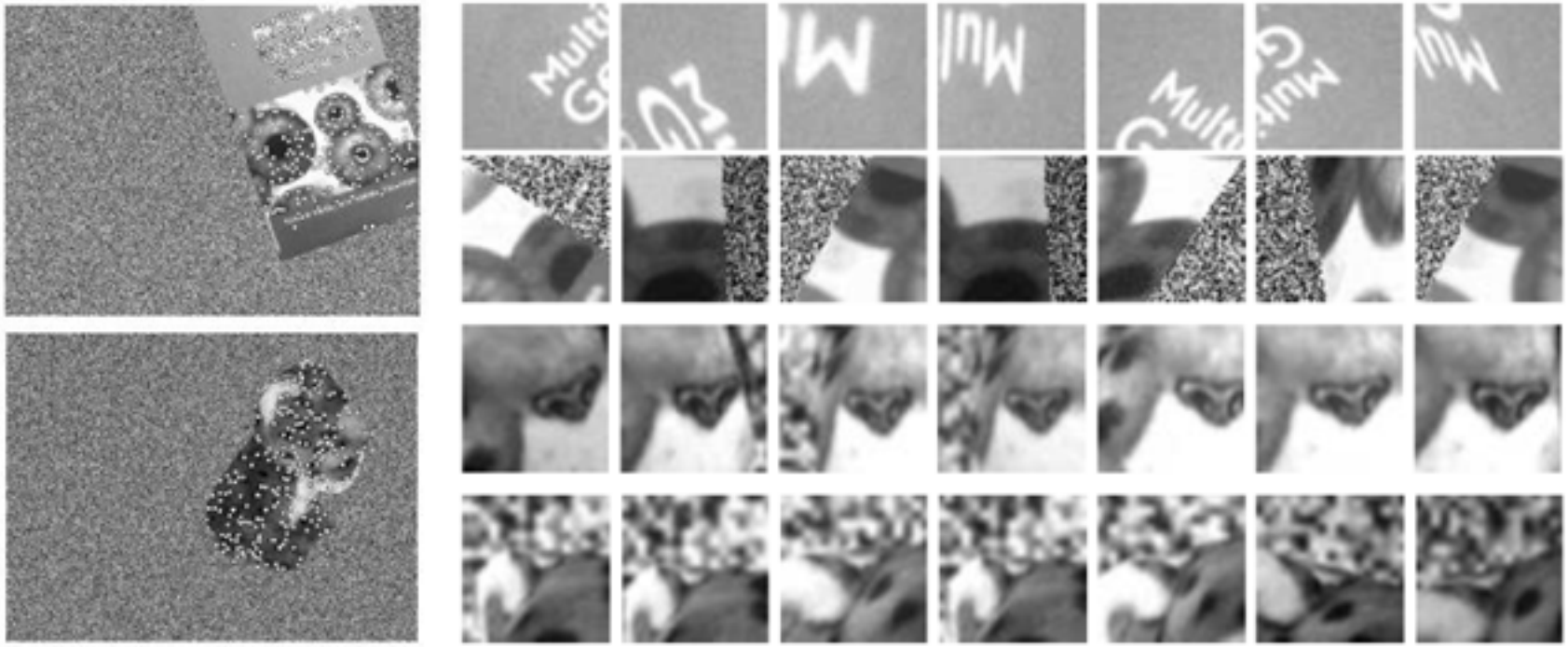
- Работает достаточно неплохо при изменении ракурсов и масштабов

Синтез обучающей выборки



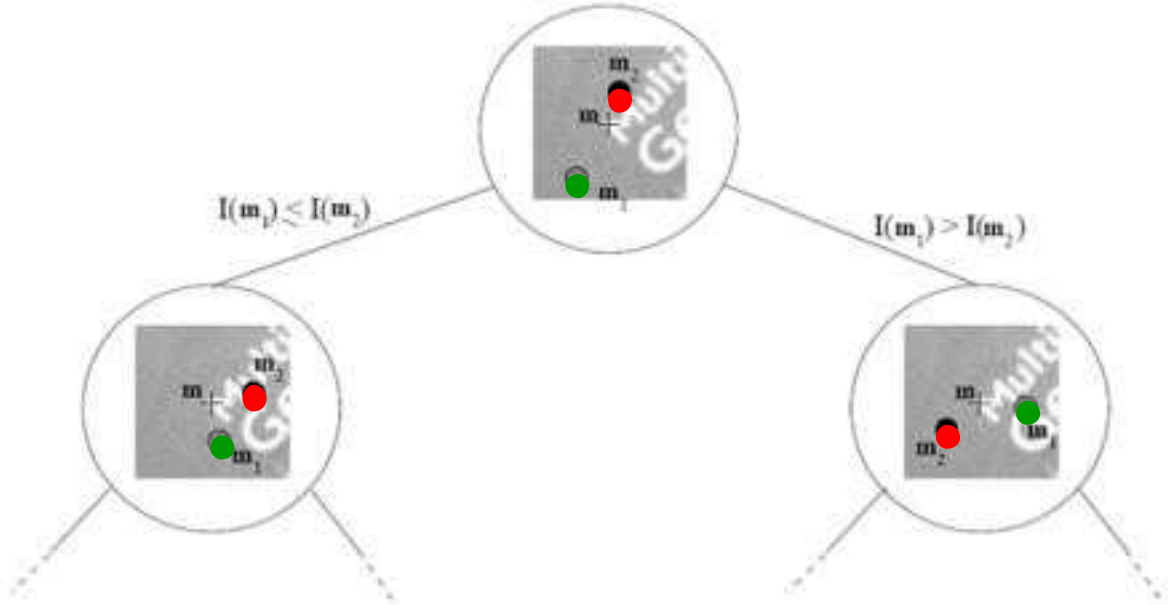
- Как получить обучающую выборку изображений с других ракурсов, есть их нет?
  - Построить 3D модель и визуализировать с других ракурсов!
- Для плоских или гладких объектов
  - Приближаем окрестность точки плоскостью
- Для сложных объектов
  - Как-нибудь (хоть вручную) строим трёхмерную модель

Выбор надежных точек



- Хотим найти k=200 надежных точек, которые находятся надежно на других ракурсах
  - Ищем ключевые точки на исходном кадре
  - Строим множество новых ракурсов
  - Считаем, сколько раз точка с исходного кадра нашлась на новых кадрах (знаем, где она должна быть)
  - Отбираем лучшие
  - Сразу получаем обучающую выборку для классификатора!

Признаки и классификатор



- Классификатор – рандомизированный решающий лес
- Признаки: сравнение двух точек в окрестности по яркости

$$C_2(\mathbf{m_1}, \mathbf{m_2}) = \begin{cases} \text{If } \mathrm{I}_\sigma(\mathbf{p}, \mathbf{m_1}) \le \mathrm{I}_\sigma(\mathbf{p}, \mathbf{m_2}) & \text{go to left child} \\ \text{otherwise} & \text{go to right child} \end{cases}$$

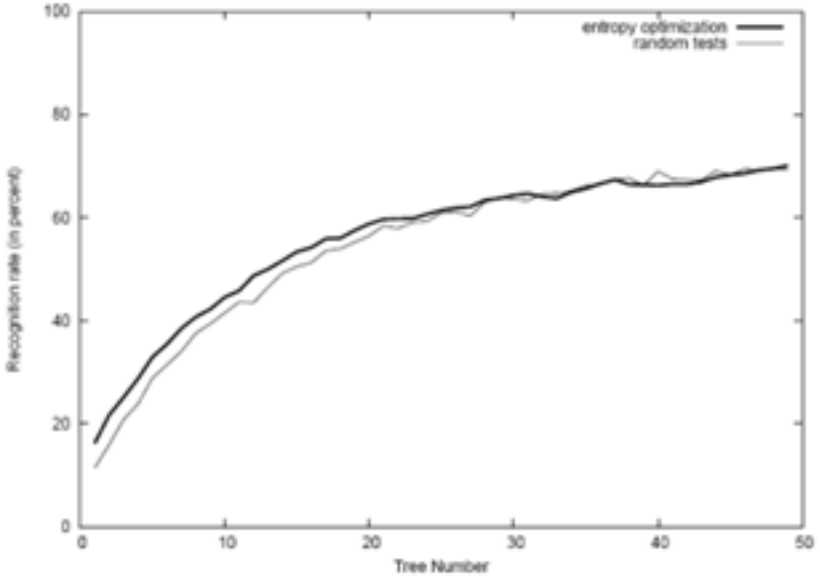
- Для окрестности 32*32 пикселей количество признаков 2^{19}
- Фрагмент можем нормализовать по ориентации

Обучение классификатора

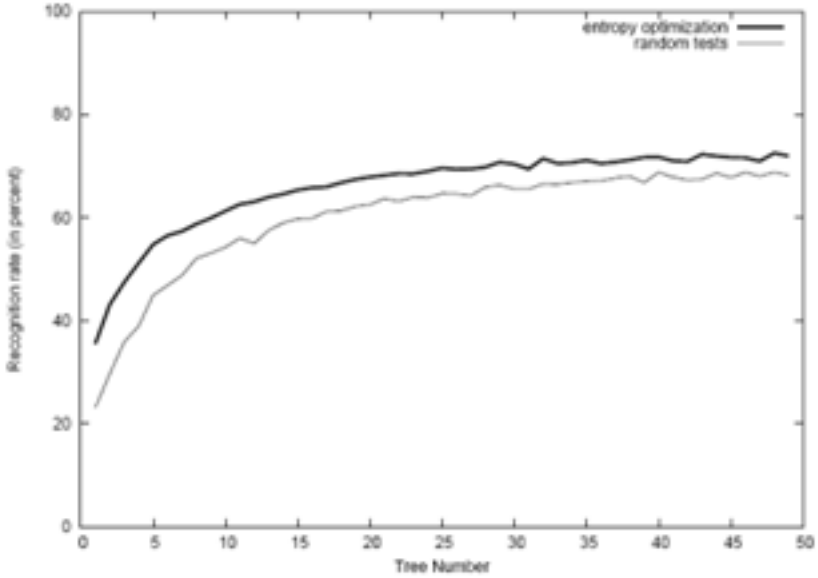
- Классический подход
  - Генерируем n тестов для каждой вершины, выбираем наилучший тест
  - n_1=10, n_d=100d
  - Строим дерево до тех пор, пока примеров в вершину приходит достаточно много (>10)
- «Экстремально-случайный»
  - Для каждой вершины берём случайный тест
  - Строим до максимальной глубины

- Качество классификации R:
  - Отношение правильно распознанных фрагментов к общему числу

Обучение классификатора



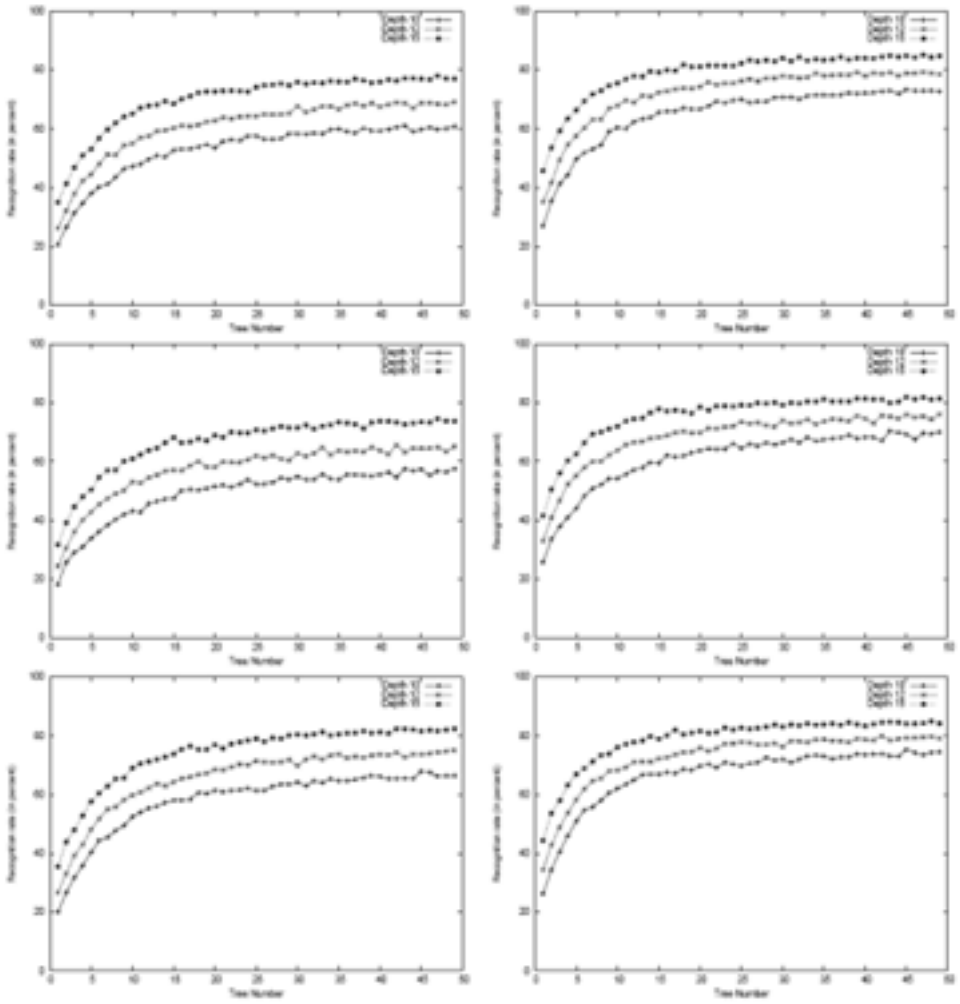
Сравнение двух подходов по точности без нормализации фрагмента по ориентации



Сравнение двух подходов по точности при использовании нормализации фрагмента по ориентации

- Выбрали вариант с нормализацией, поскольку по скорости/размеру классификатора он оказался предпочтительнее
- Обучение экстремально случайное, занимает несколько секунд (десятки минут для другого варианта)

Оценки обучения и признаков



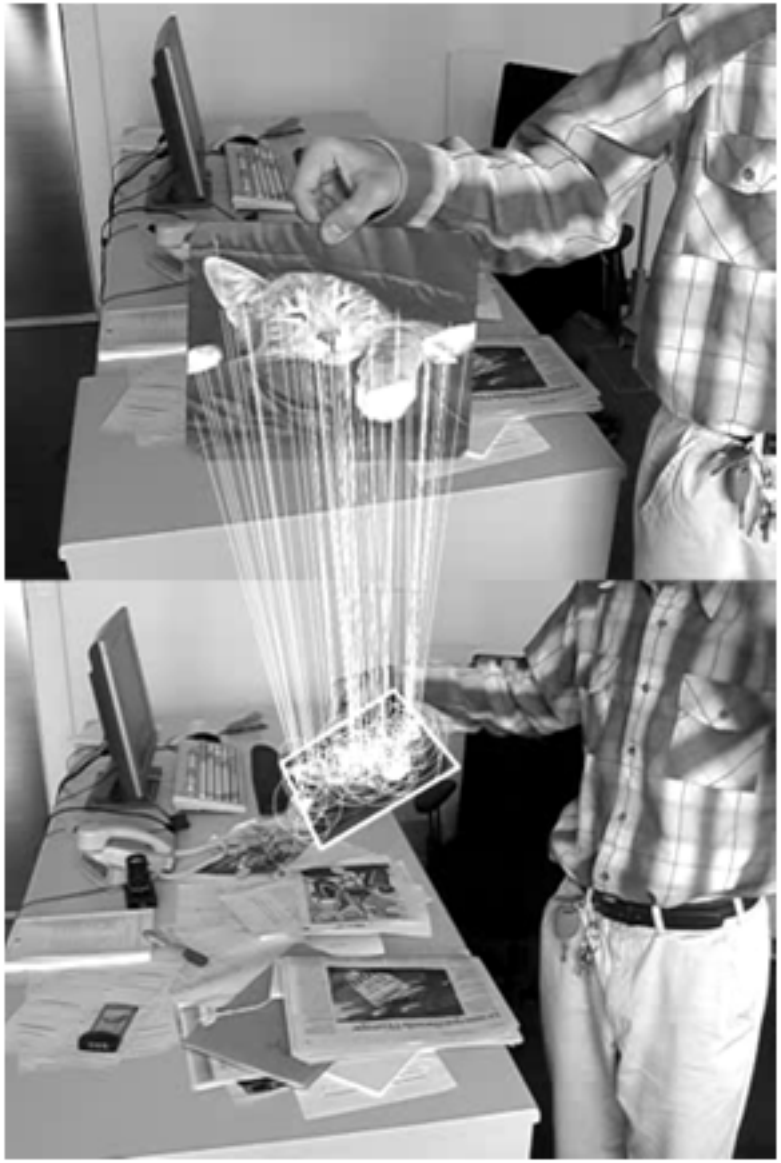
C2
признаки

C4 признаки (сравнение 4х точек – два градиента)

Ch признаки (вычисление SIFT и двух случайных параметров из него)

Сравнение простых признаков с более сложными показывают достижение похожей точности и насыщение точности при лесе из 20 деревьях для всех признаков

Результат работы



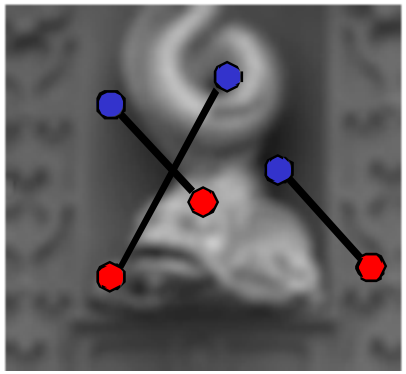
Выводы

Эта красивая работа учит нас следующему:

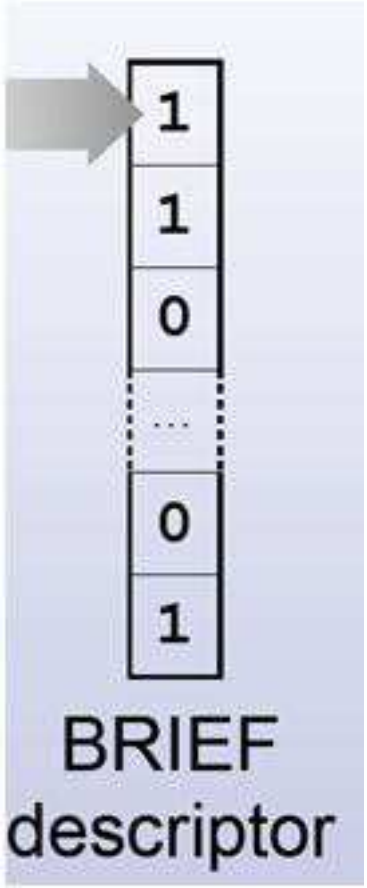
- Можно обучать классификатор на синтезированных данных при недостатке данных
- Рандомизированные деревья можно очень быстро обучить и они быстро работают
- Задачу сопоставления можно решить как задачу классификации достаточно быстро и эффективно

Код доступен:
http://cvlab.epfl.ch/software/bazar/index.php

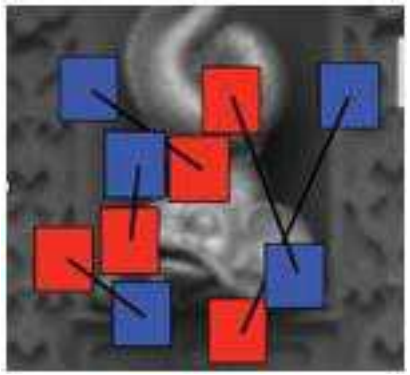
BRIEF



Сглаживание

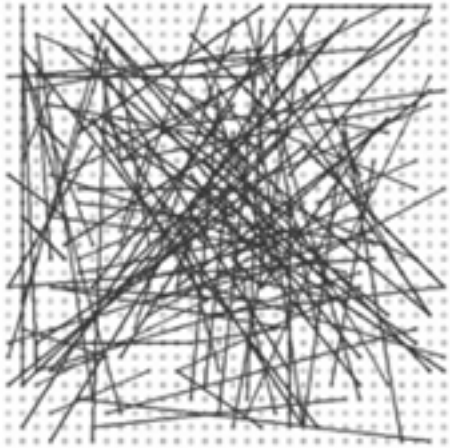


Можем использовать интегральные изображения

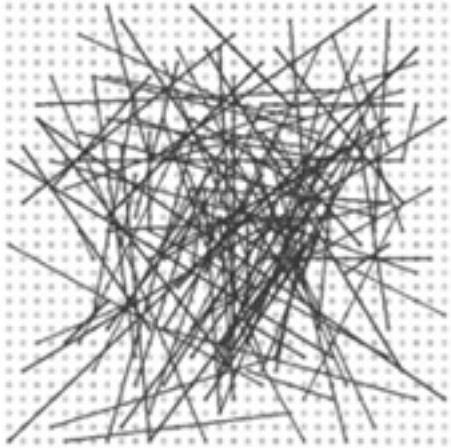


M. Calonder, V. Lepetit, C. Strecha, and P. Fua, BRIEF: Binary Robust Independent Elementary Features. ECCV, 2010

Варианты

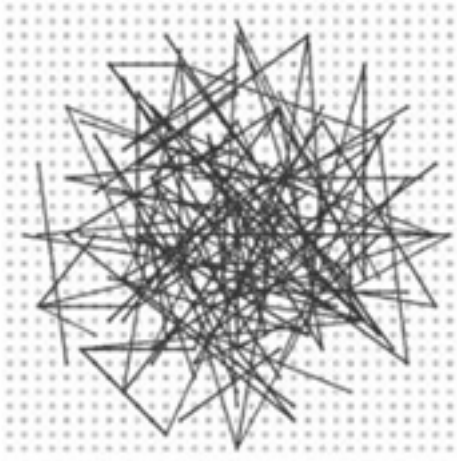


Выбор по равномерному распределению

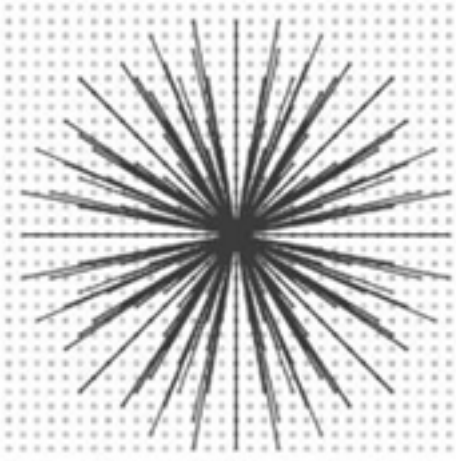


Выбор по нормальному распределению

Локализованное нормальное распределение

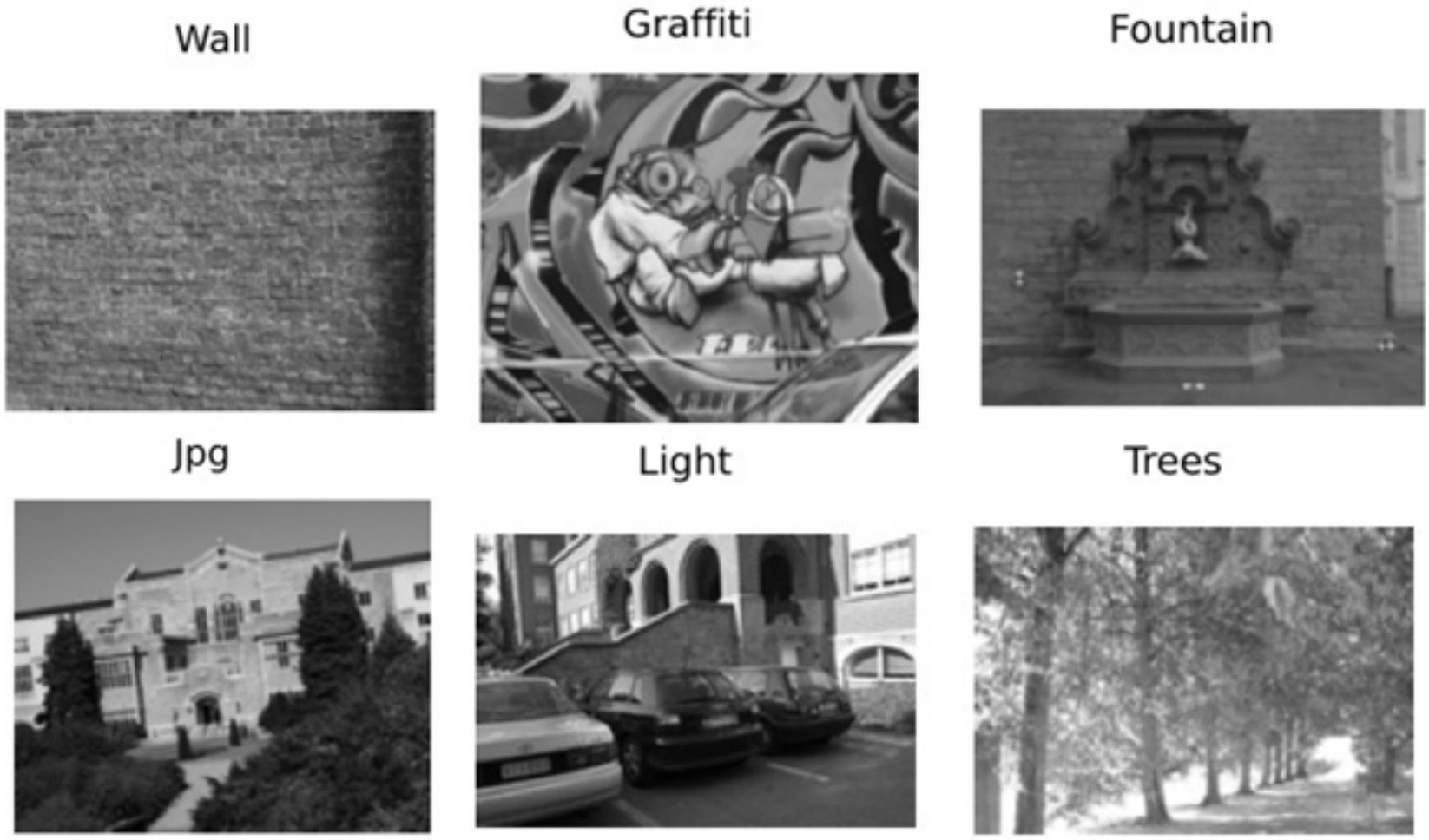


Равномерное распределение в полярных координатах



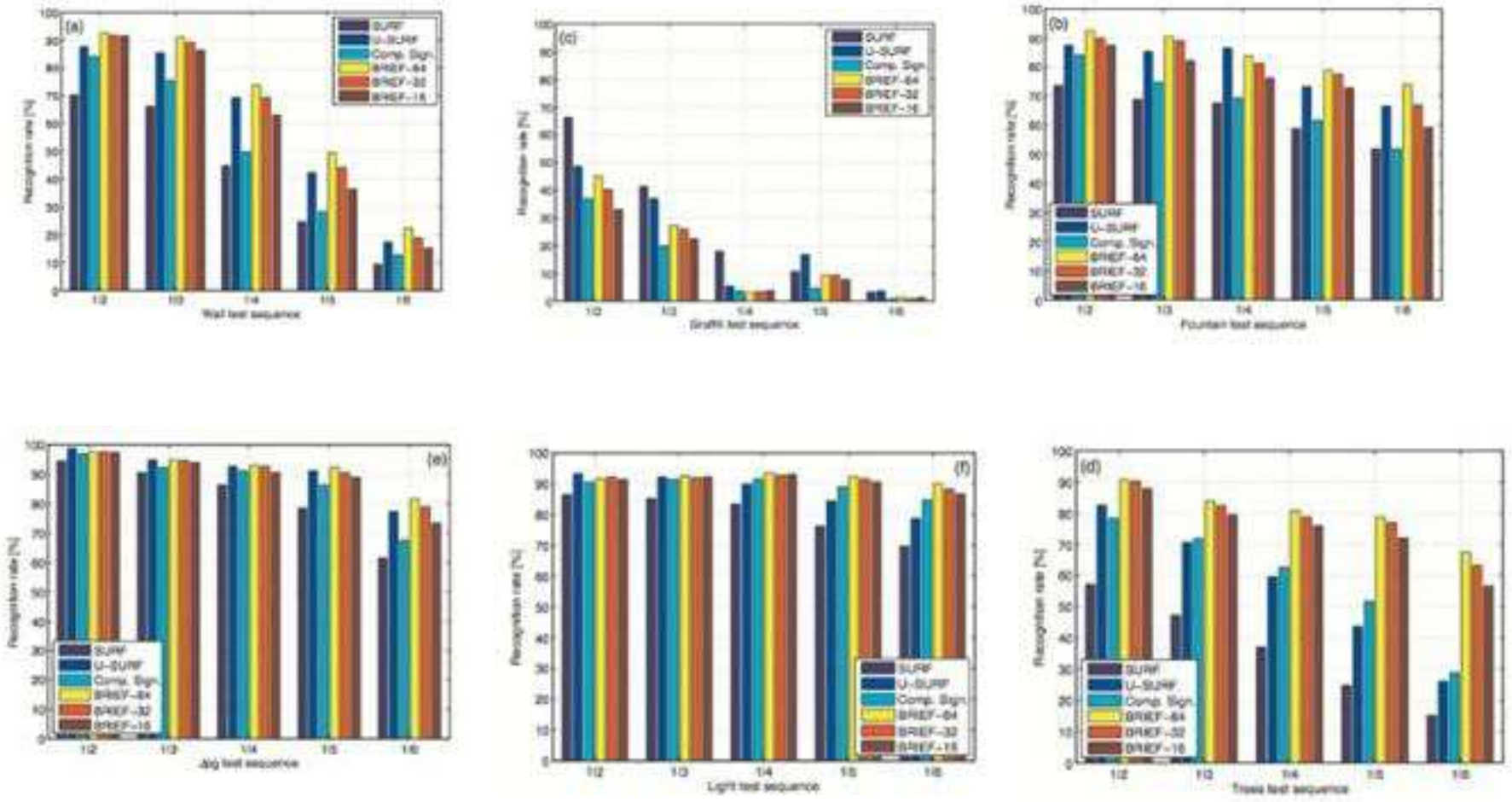
Равномерно по сетке в полярных координатах

Эксперименты



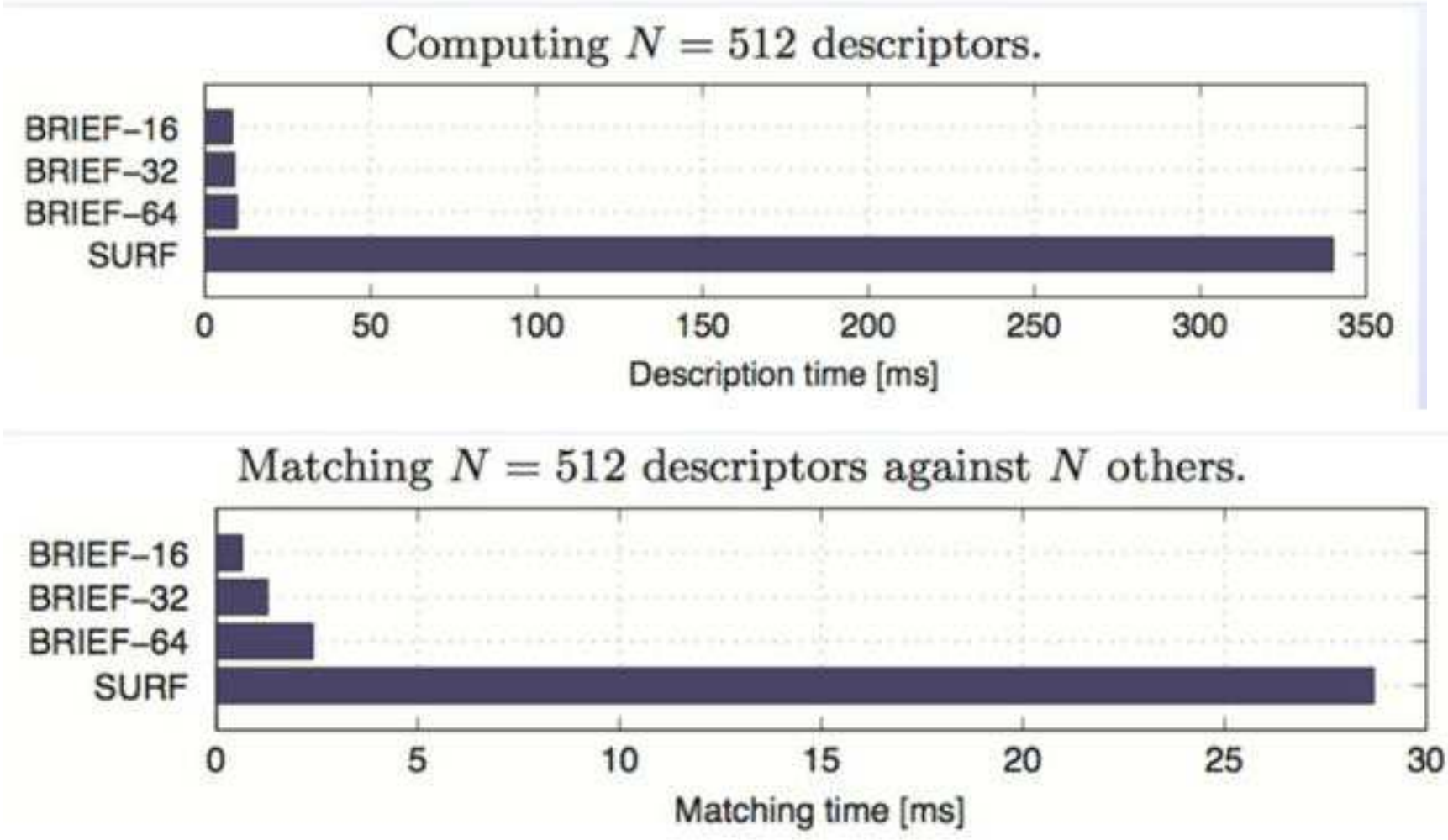
Будем распознавать категории изображений по ключевым точкам с разными дескрипторами

Эксперименты



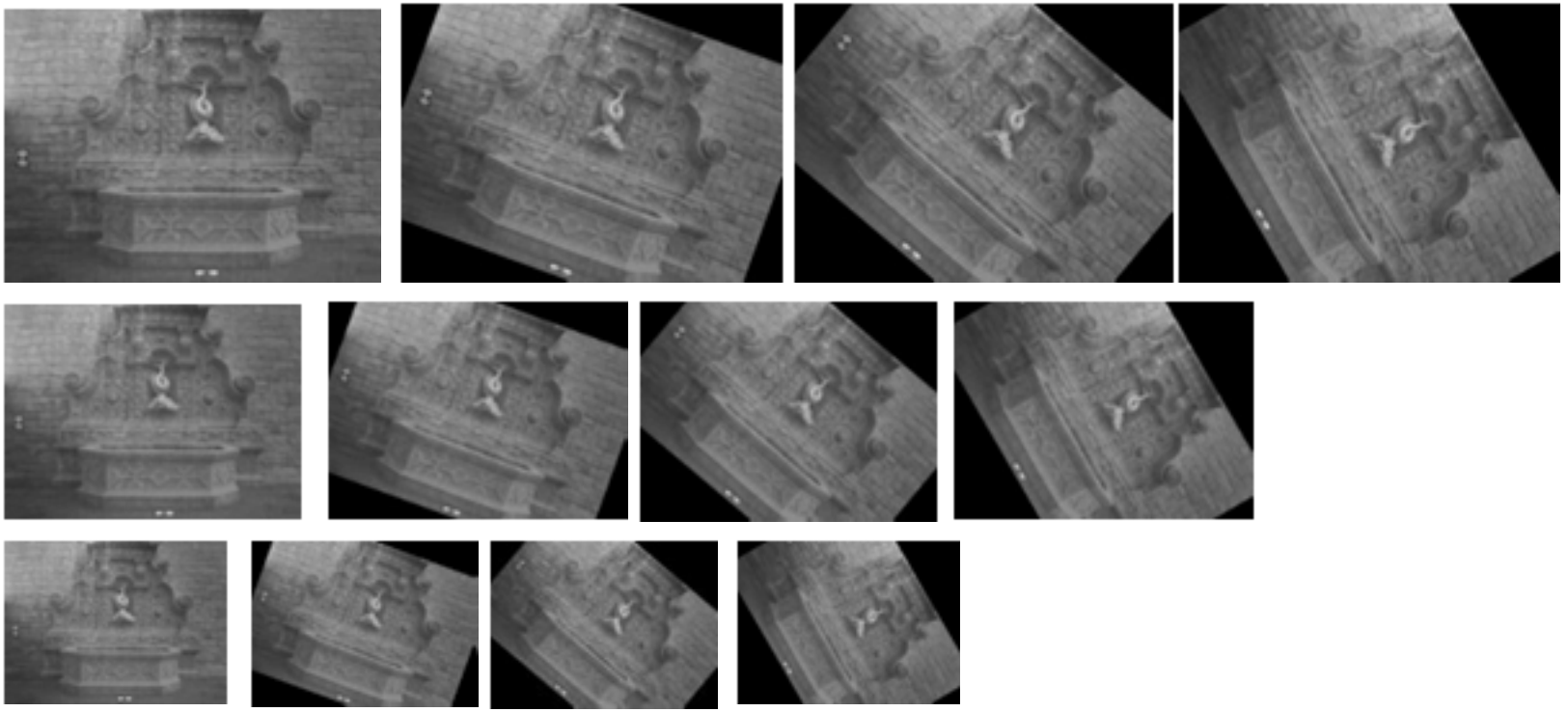
Эксперименты показывают, что BRIEF справляется не хуже, а часто и лучше обычных дескрипторов

Скорость работы

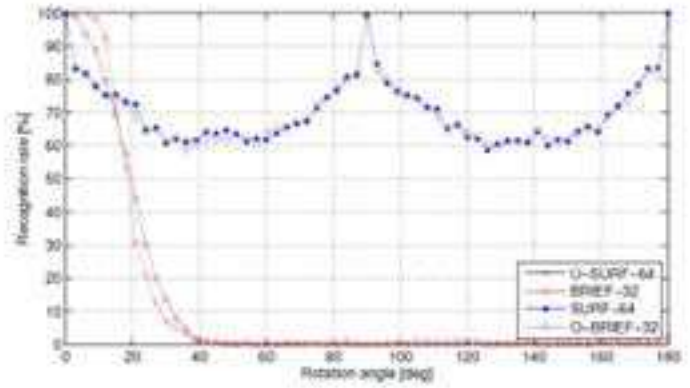


Сравнение BRIEF по скорости вычисления и сопоставления с SURF – быстрым вариантом SIFT (на основе гистограмм ориентации градиентов)

Добавление инвариантности



- Дескриптор изначально не очень устойчив к повороту и масштабу
- Размножим выборку: несколько поворотов и масштабов
- Будем сравнивать тестовое изображение с каждым из синтетических



Пример работы

AR-настольные игры

Калибровка доски

Локализация области фигур

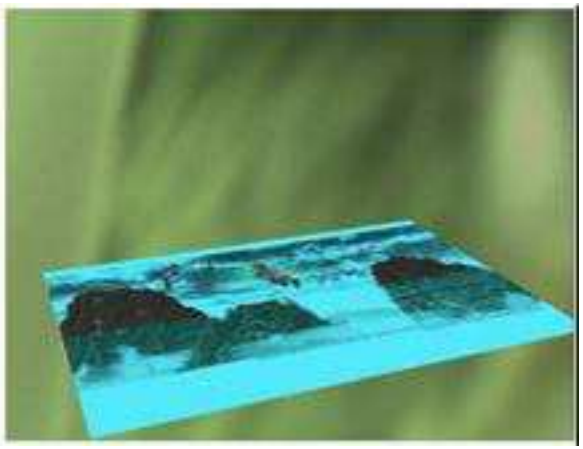
Выделение фигур VJ

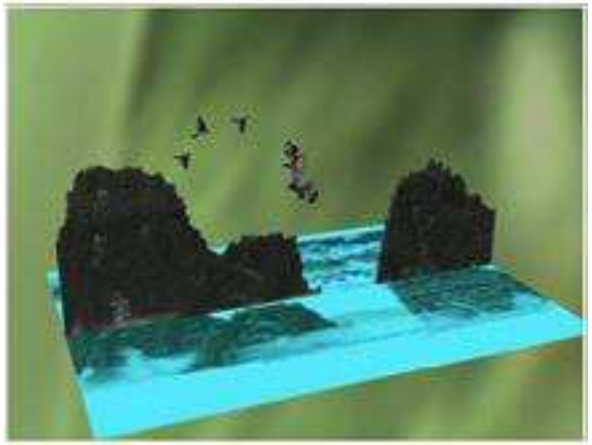
Дополненная сцена

E. Molla and V. Lepetit, Augmented Reality for Board Games. In *Proceedings of the International Symposium on Mixed and Augmented Reality*, 2010.

Видео-пример

AR-книги

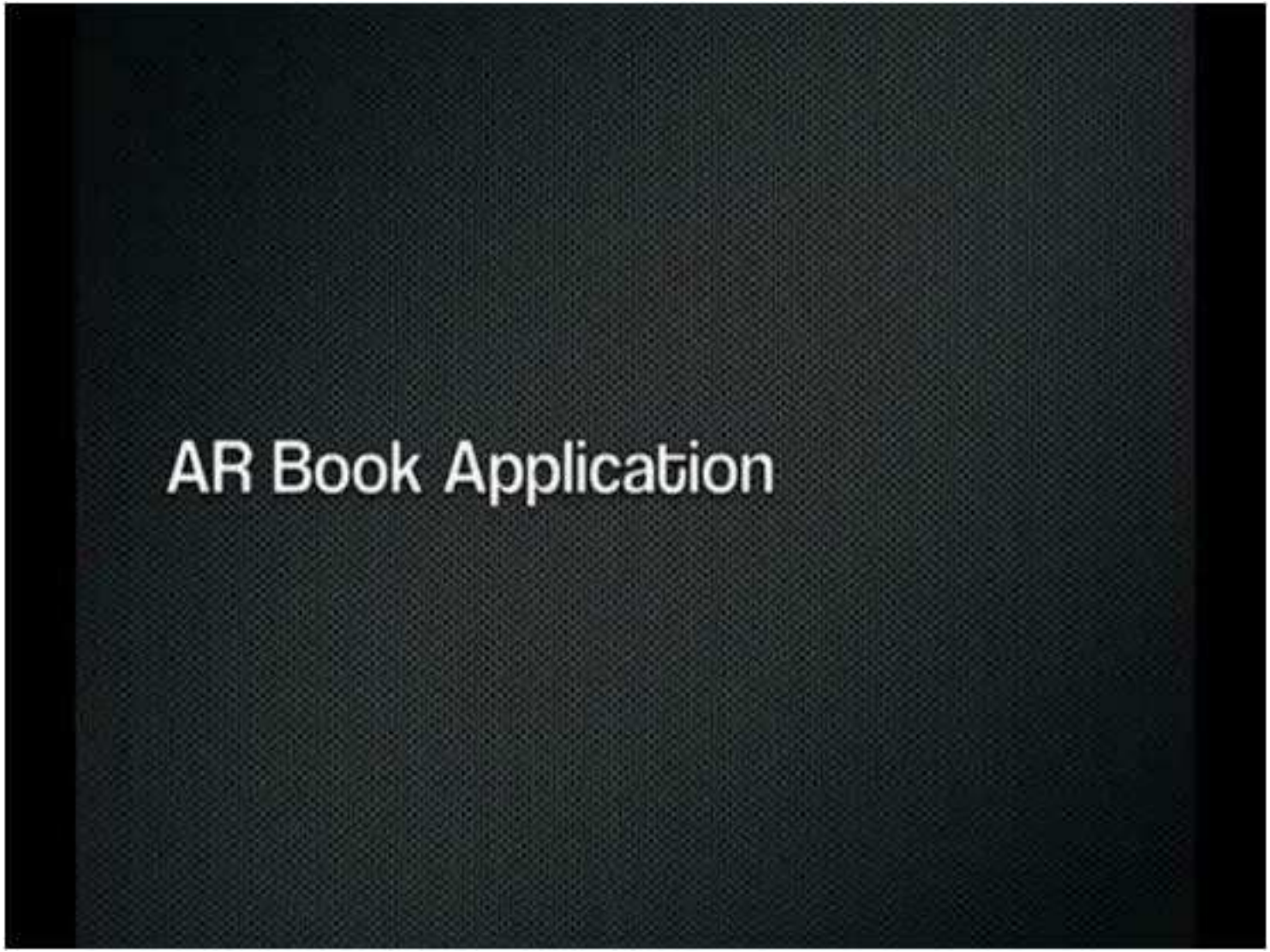




Множество иллюстраций с разным контентом – будем параллельно распознавать и отслеживать (2 потока обработки)

K. Kim, V. Lepetit, W. Woo, Scalable Real-time Planar Targets Tracking for Digilog Books. *Computer Graphics International*, 2010

AR-иллюстрации

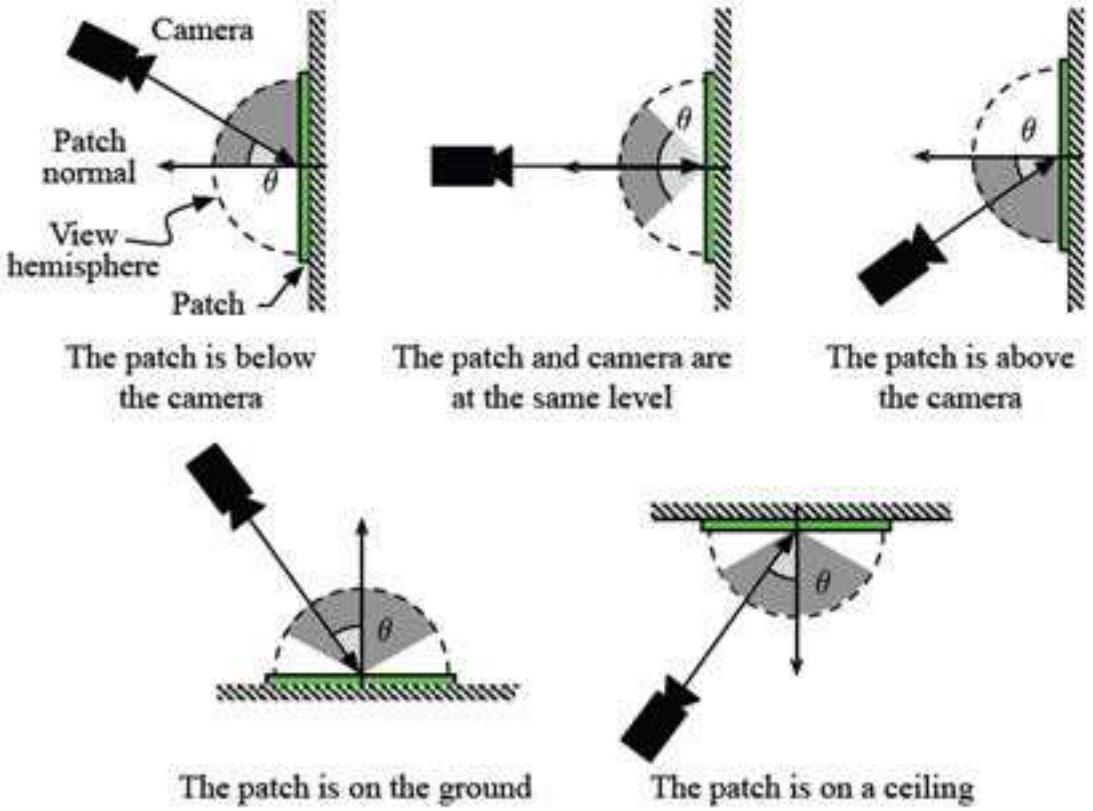


Подход на основе шаблонов

Попробуем отслеживать один объект («шаблон»), а не множество точек

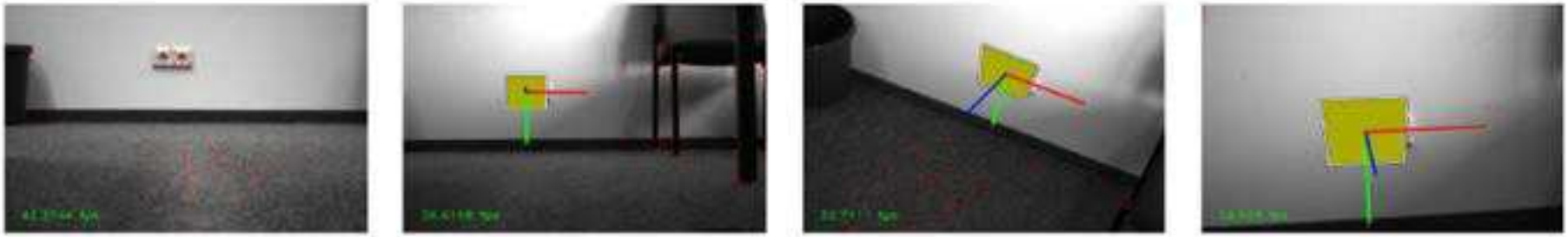
W. Lee, Y. Park, V. Lepetit, and W. Woo, Point-and-Shoot for Ubiquitous Tagging on Mobile Phones. In *Proceedings of the International Symposium on Mixed and Augmented Reality*, 2010

Определение ракурса



- По ориентации телефона определяем на какой поверхности пользователь выбрал шаблон
- Зная ориентацию камеры относительно плоскости синтезируем фронтальный ракурс, на котором инициализируем фрагмент

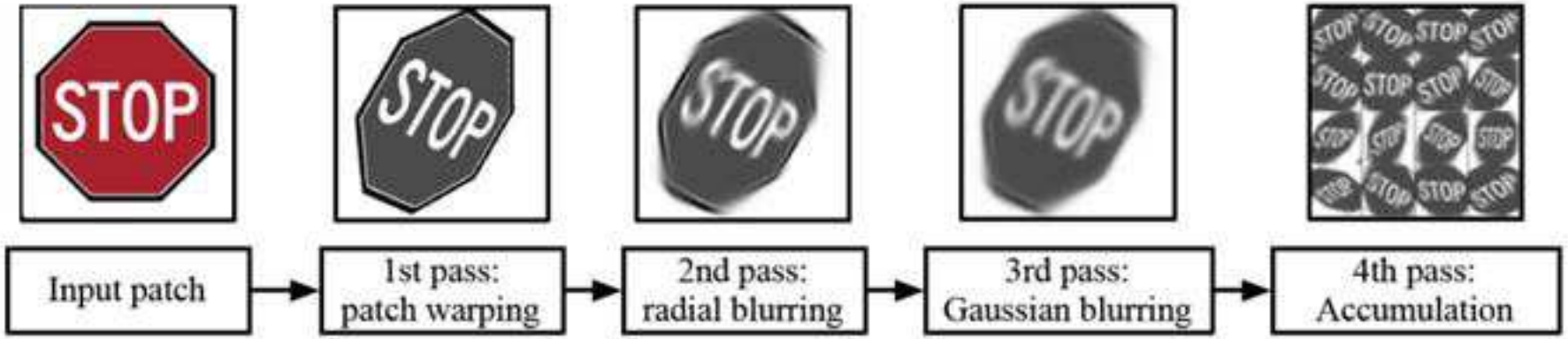
Отслеживание плоских фрагментов



- Задача – отслеживать плоский фрагмент, одновременно определяя его ориентацию относительно камеры («регистрация»)
- Идея:
  - Синтезируем несколько «усреднённых шаблонов» этого фрагмента для новых ракурсов, путем усреднения нескольких изображений с близкими ракурсами
  - Для быстрого построения «усреднённых шаблонов» предложена специальная процедура, не требующая действительно синтезировать и смешивать кучу видов
  - Набор «усреднённых шаблонов» - своеобразный дескриптор точки, одновременно содержащий информацию о ракурсе

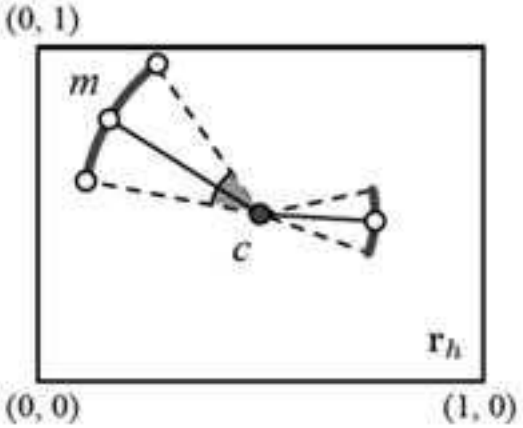
S. Hinterstoisser, O. Kutter, N. Navab, P. Fua, and V. Lepetit, Real-Time Learning of Accurate Patch Rectification. In *Proceedings of the Conference on Computer Vision and Pattern Recognition*, 2009

Построение «средних шаблонов»



- Исходный фрагмент – 128*128 пикселей
- Трансформация, затем радиальный смаз, затем гауссов смаз
- Затем шаблон уменьшается до 32*32 пикселей
- 225 видов, всего около 900kb памяти
- 0.3с на обучение на PC, 6-7с на телефоне

Радиальный смаз



Усреднение пикселей вдоль дуги

Результаты

- Слежение: поиск наилучшего фрагмента и уточнение камеры с помощью алгоритма ESM-Blur
- 10-15 кадров/с

Y. Park et. al. ESM-Blur: Handling and Rendering Blur in 3D Tracking and Augmentation. ISMAR 2009.

Пример работы

Резюме

- Рандомизированные методы часто показывают высокую скорость и качество работы
  - Рандомизированный решающий лес (Random Forest)
  - Рандомизированные дескрипторы (BRIEF)

- Простых признаков (попарных сравнений пикселей) может оказаться достаточно

- При нехватке данных можно их синтезировать и на них обучить алгоритм

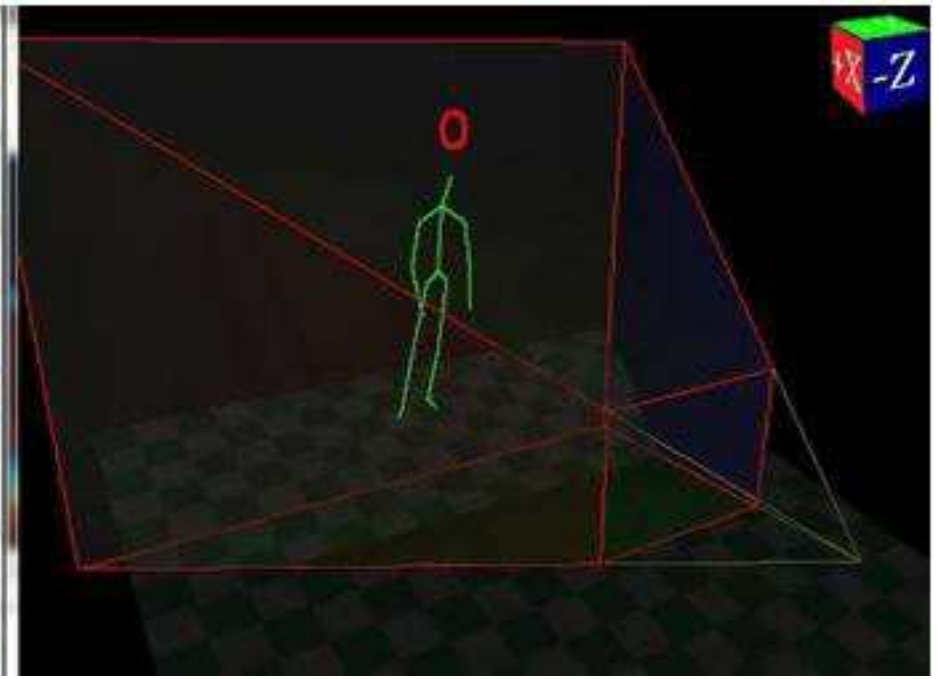
Kinect

Управление жестами

Одна из самых старых задач компьютерного зрения, пришедшая из научной фантастики

Оценка 3D позы человека

Изображение с камеры



«Скелет» человека

Это только часть задачи. Нам нужно ещё интерпретировать позу и движения человека

Первые работы (1983)

Model-based vision: a program to see a walking person

David Hogg

For a machine to be able to 'see', it must know something about the object it is 'looking' at. A common method in machine vision is to provide the machine with general rather than specific knowledge about the object. An alternative technique, and the one used in this paper, is a model-based approach in which particulars about the object are given and this drives the analysis. The computer program described here, the WALKER model, maps images into a description in which a person is represented by the series of hierarchical levels, i.e. a person has an arm which has a lower-arm which has a hand. The performance of the program is illustrated by superimposing the machine-generated picture over the original photographic images.

Keywords: vision, machine perception, WALKER model

INTRODUCTION

Vision systems, both natural and artificial, require knowledge about the perceived objects, although the role played by this knowledge in the analytical process is unclear. Many techniques of machine vision seek to generate 3D structural descriptions without invoking object specific knowledge. An alternative is to adopt the 'model-based' approach wherein particular knowledge about the objects being sought drives the analysis.

This paper is concerned with a computer program that understands TV image sequences depicting a person walking through an arbitrary environment (Figure 1). The program maps given image sequences into a description in which the human body is represented by a collection of connected cylinders corresponding to its parts. It is supposed that such a 3D structural description would be both necessary and sufficient for many everyday tasks to be performed effectively. For example, touching someone's arm or deciding whether several people are marching in step all appear to require a grasp of 3D structure whether perceived visually or otherwise. Each output description is an instance of an abstract 3D model for a class of human walkers, henceforth called the WALKER model, itself an input to the program (Figure 2).

Descriptions generated by the program are sufficiently detailed to determine a pictorial reconstruction of the person from the perspective of the original imaging device. By superimposing these reconstructions over the original images a close indication of the program's performance is visible to the human observer. When presented with the sequence depicted in Figure 1, the program generates as part of its output the sequence shown in Figure 3. The program copes with the enormous local ambiguity in an image by weighing evidence from across the image in support of a large number of possible interpretations. As a consequence, the program's performance should degrade gracefully for increasingly difficult image sequences in which the walker may be obscured or occluded to the camera.

Visual problem

The visual problem can be divided broadly into two parts, namely, what should be described and how can such descriptions be derived from a time-varying 2D image. It is impossible to divorce these two issues from one another since the difficulty of deriving a description from an image is bound to depend on the things being described. Moreover, certain representations may be required solely as intermediate descriptions for the interpretative process itself.

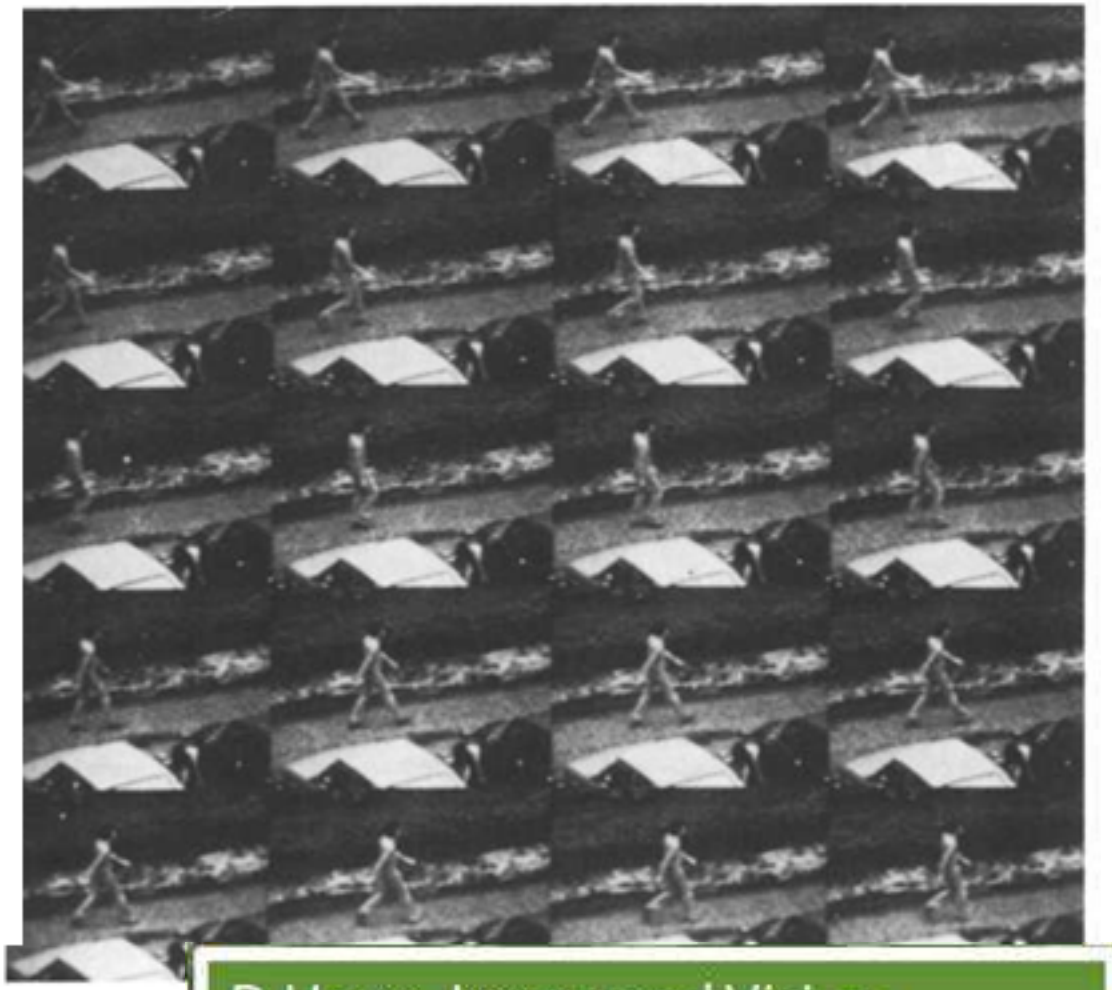
The question of what should be represented must depend on the visual system's function within a cognitive machine whose ultimate goal may be far removed from the visual world. This paper takes a non-controversial stand in accepting the usefulness of 3D structural descriptions as an interface to a larger system and instead concentrates on the second issue of how to generate such a description from an image.

General-knowledge inference

Much of the current work in computer vision is concerned with the generation of 3D descriptions using only general-

School of Engineering and Applied Sciences, University of Sussex, Brighton, Sussex, UK.
This research originated in this paper was carried out while the author was an SERC funded research student in the Cognitive Studies Programme at the University of Sussex

vol 1 no 1 February 1983 0262-8856/83/010005-20$03.00

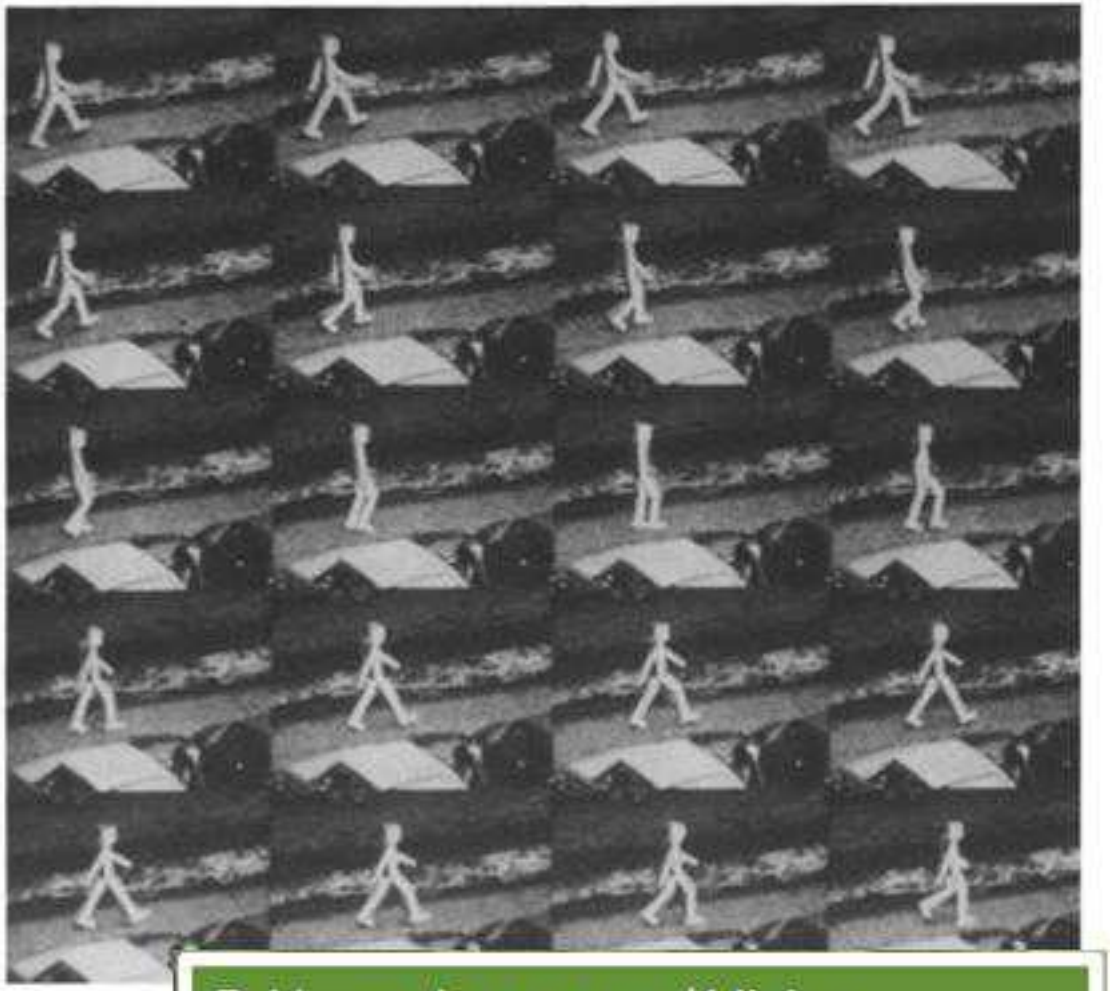


D Hogg, Image and Vision Computing, Vol 1 (1983)

Первые работы (1983)

Model-based vision: a program to see a walking person

David Hogg



D Hogg, Image and Vision Computing, Vol 1 (1983)

Pfinder (People Finder) 1995



- Вычитание фона для получения маски человека
  - Одна Гауссиана для каждого пикселя
- Моделирование человека как несколько «блобов»
  - модель Гауссиана
  - параметры пикселя - (x,y,Y,U,V)
  - пиксель человека должен принадлежать одному из блобов человека

Christopher Wren, Ali Azarbayejani, Trevor Darrell, Alex Pentl Pfinder: Real-Time Tracking of the Human Body, PAMI 1997

PFinder

- Инициализация модели
  - «Стартовые» позы – оценка контура, локализация частей тела и моделирование блобов в этих областях

- На каждом кадре:
  - Получение маски переднего плана
  - Для каждого пикселя оценка логарифма правдоподобия принадлежности к каждому блобу

$$d_k = -\frac{1}{2}(\mathbf{y}-\boldsymbol{\mu}_k)^T\mathbf{K}_k^{-1}(\mathbf{y}-\boldsymbol{\mu}_k) - \frac{1}{2}\ln|\mathbf{K}_k| - \frac{m}{2}\ln(2\pi)$$

$$s(x,y) = \mathrm{argmax}_k(d_k(x,y))$$

  - Обновление моделей блобов
  - Предсказание/сглаживание фильтром Калмана

Pfinder: демо

Подходы по изображениям

Okada & Stenger 2008

Navaratnam *et al.* 2007

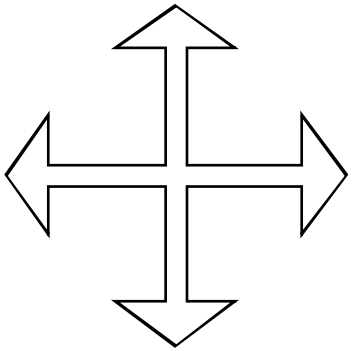
Реконструкция по фотографиям

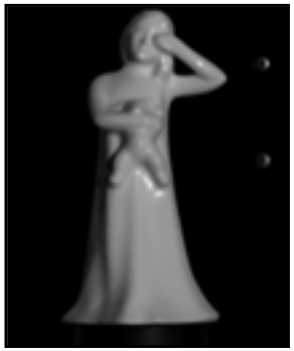
Оценка 3D позы близка к задаче 3D реконструкции по изображениям. Для деформирующегося объекта без текстуры (человек в обычной одежде) только по изображениям крайне сложна.

жесткий

с текстурой

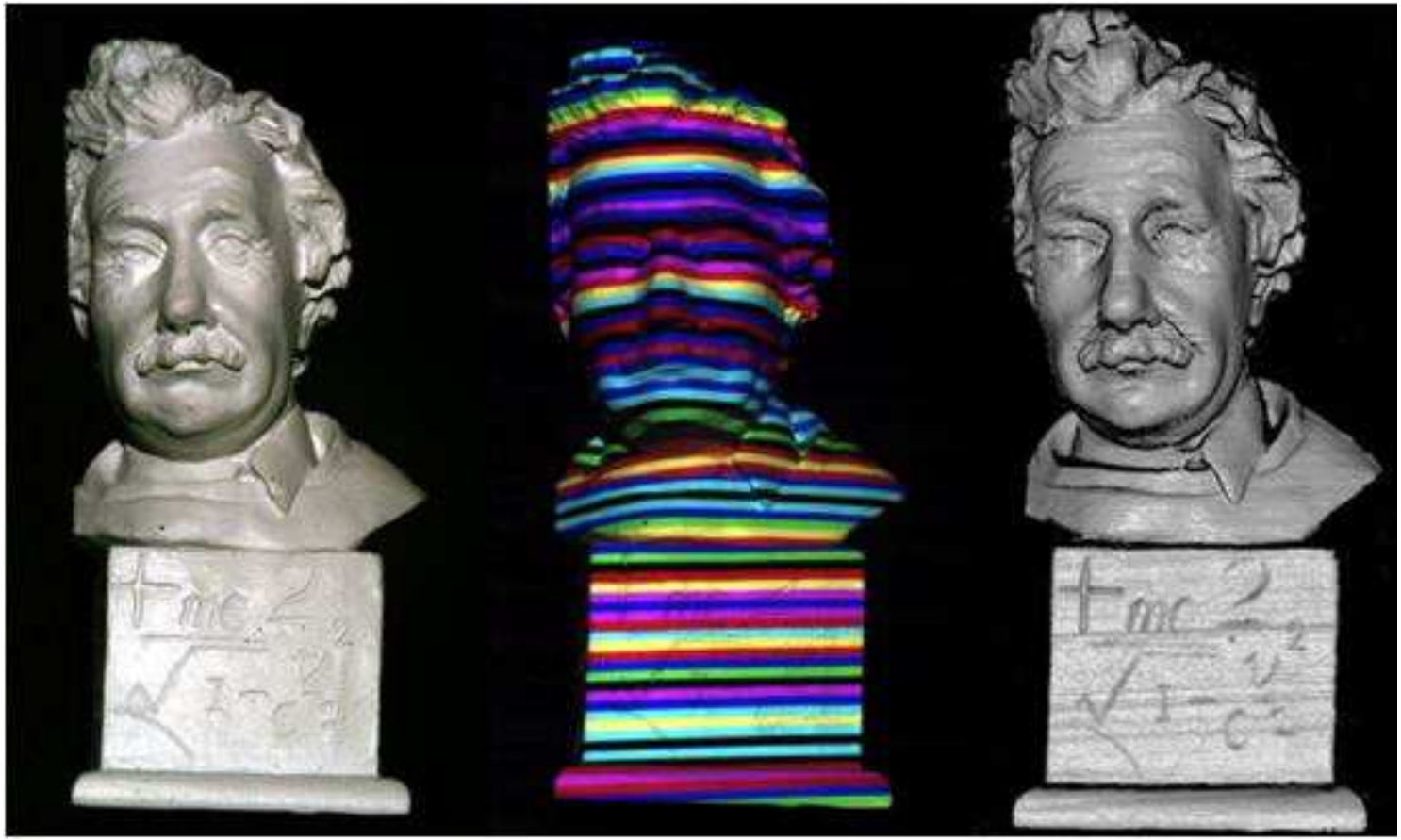
деформирующийся





без текстуры

Структурированный свет



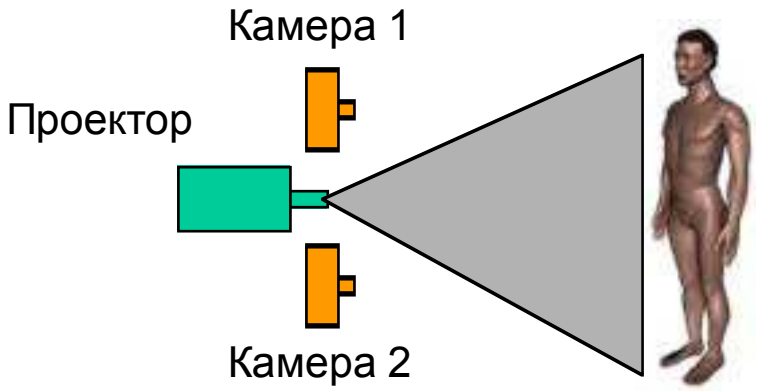
Специальной подсветкой мы можем свести задачу к более простой – стерео-реконструкции текстурированных объектов

[Zhang02]

«Активное стерео»



- Проецируем специальный «шаблон» на объект («структированный свет»)
- Шаблон даёт «текстуру» по всей поверхности объекта
- Решаем задачу стерео либо по 2м камерам, либо с калиброванным проектором
- Подсветка может быть в видимом диапазоне, а может быть ИК

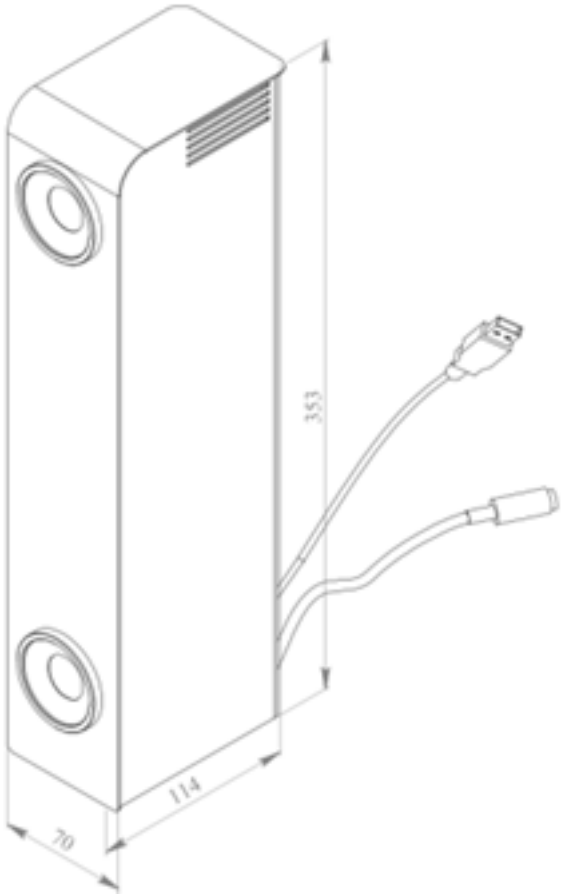


KINECT

Пример 3D камеры

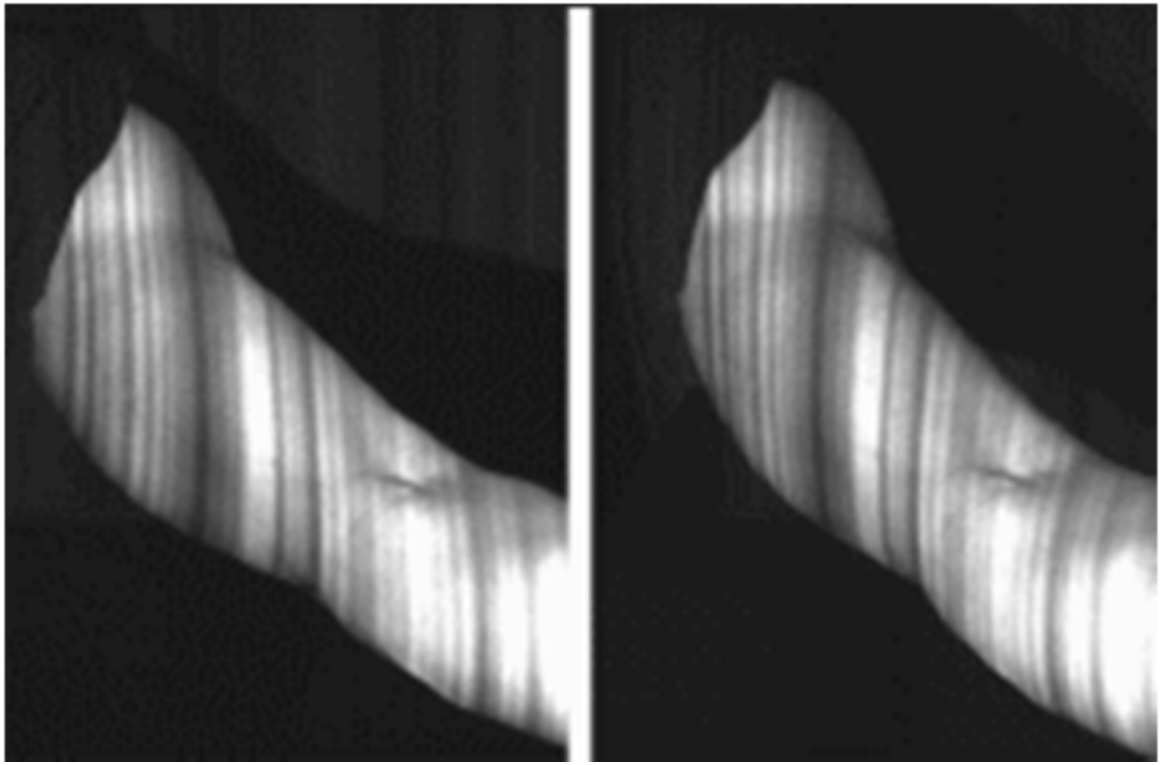
Спецификация

Модель	TDSL-1.1	TDSM-1.1
Размеры, HxDxW	353 x 114 x 70 мм	266 x 114 x 70 мм
Вес	2.3 кг	1.9 кг
Питание	12В, 36Вт	12В, 36Вт
Интерфейс	1xUSB2.0	1xUSB2.0
Точность, режимы		
однокадровый, до	0.3 мм	0.15 мм
многокадровый, до	0.1 мм	0.05 мм
Разрешение, режимы		
однокадровый, до	200'000 точек	200'000 точек
многокадровый, до	неограничено	неограничено
3D форматы	.ply, .obj, .stl, .wrl	.ply, .obj, .stl, .wrl
Рабочая дистанция	0.8 – 1.6 м	0.4 – 1.0 м
Поле зрения, HxW	41x32°	30x21°
Время экспозиции	0.1мс	0.1мс
Частота съемки	0 — 15fps	0 — 15fps
Скорость объекта, до	30 км/ч	30 км/ч
Источник света	вспышка (не лазер)	вспышка (не лазер)

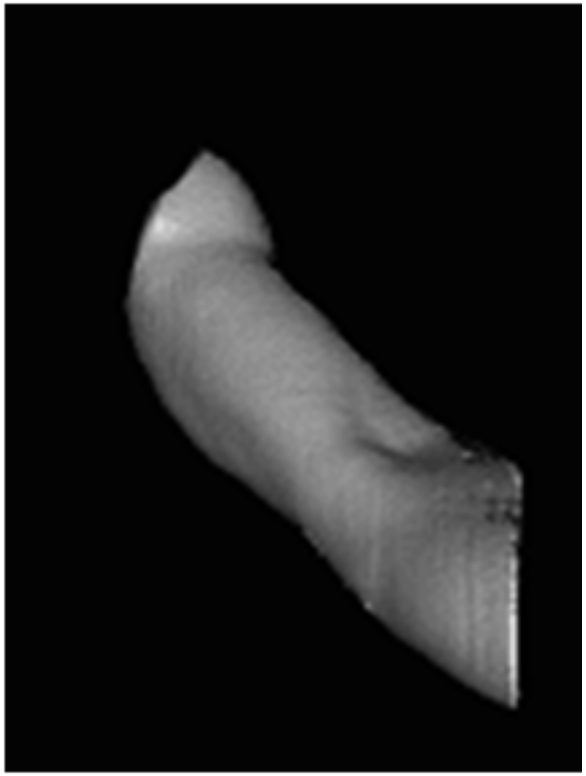


http://artec-group.ru

Пример реконструкции

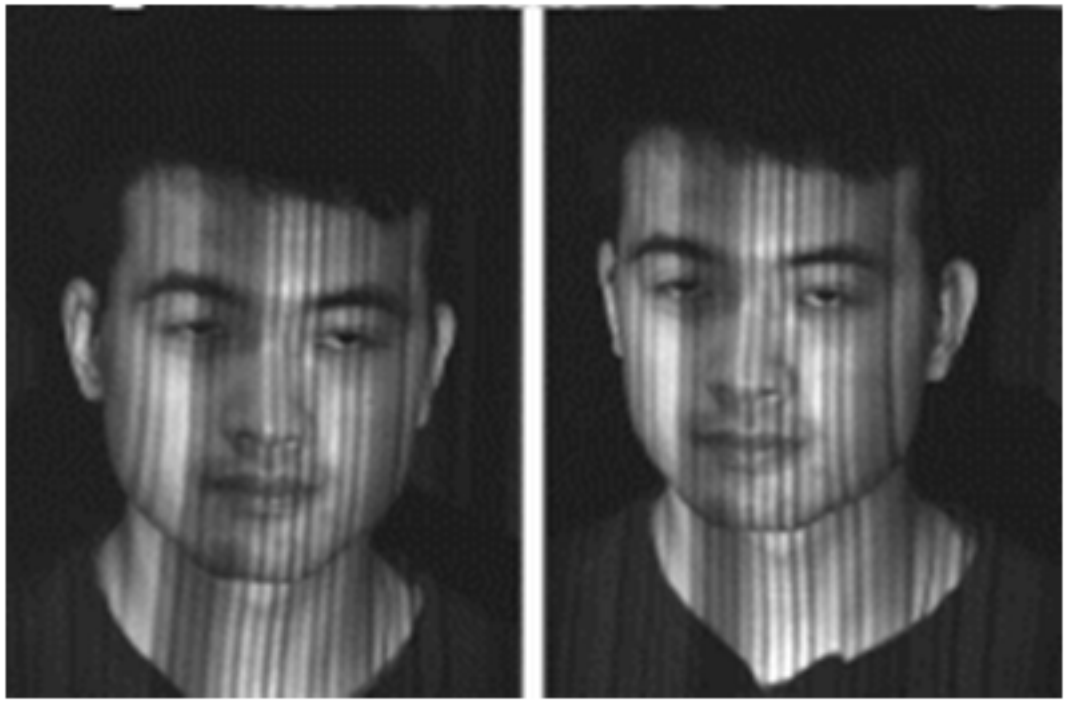


Исходные видео-потоки

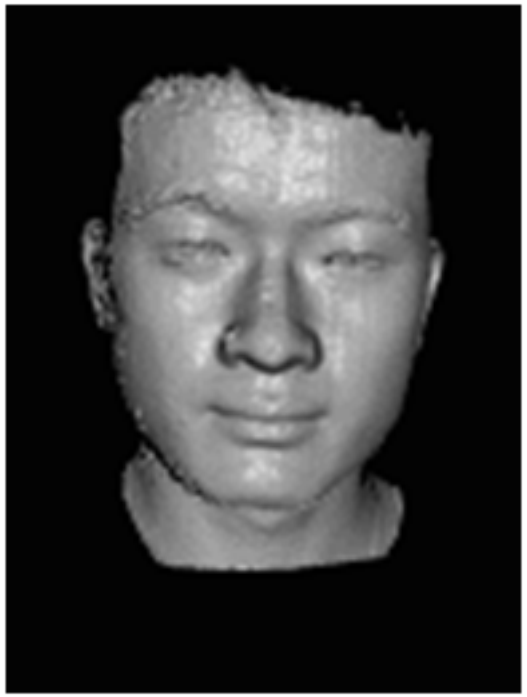


Реконструкция

Пример реконструкции

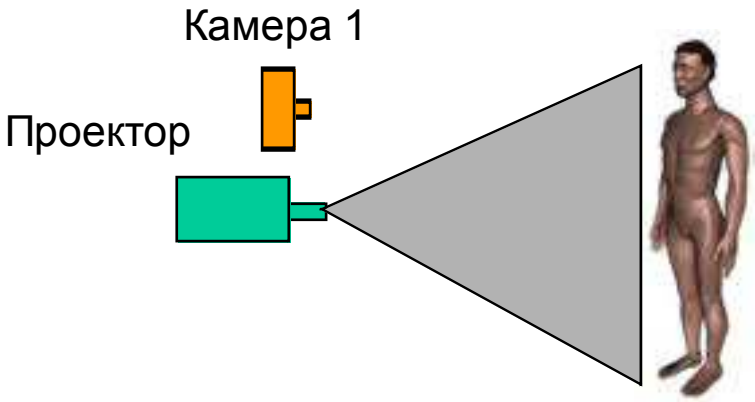


Исходные видео-потоки

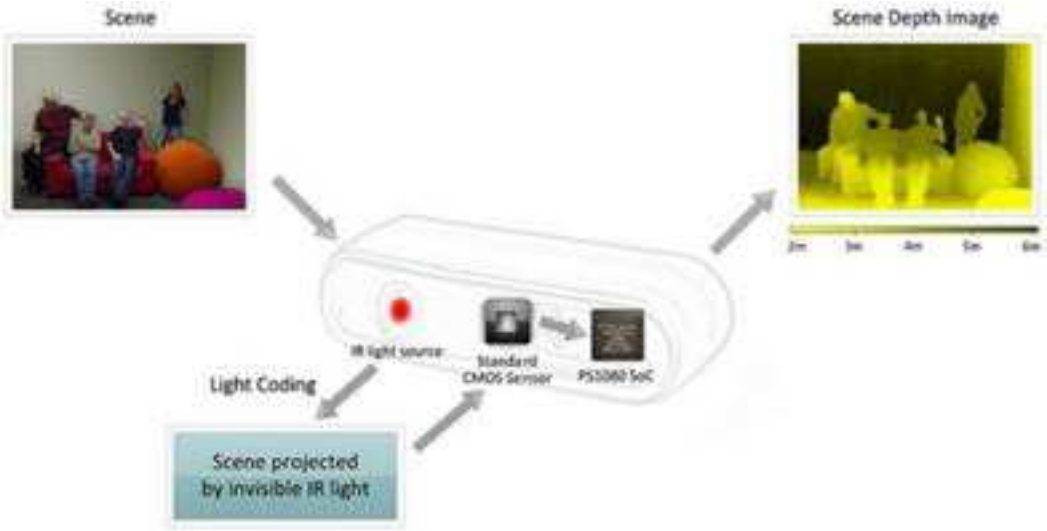


Реконструкция

Kinect



- Технология компании PrimeSense
- Лицензирована Microsoft
- Kinect – разработка Microsoft
- Внутри ещё фазированный микрофон



«Система на кристале» (SoC)

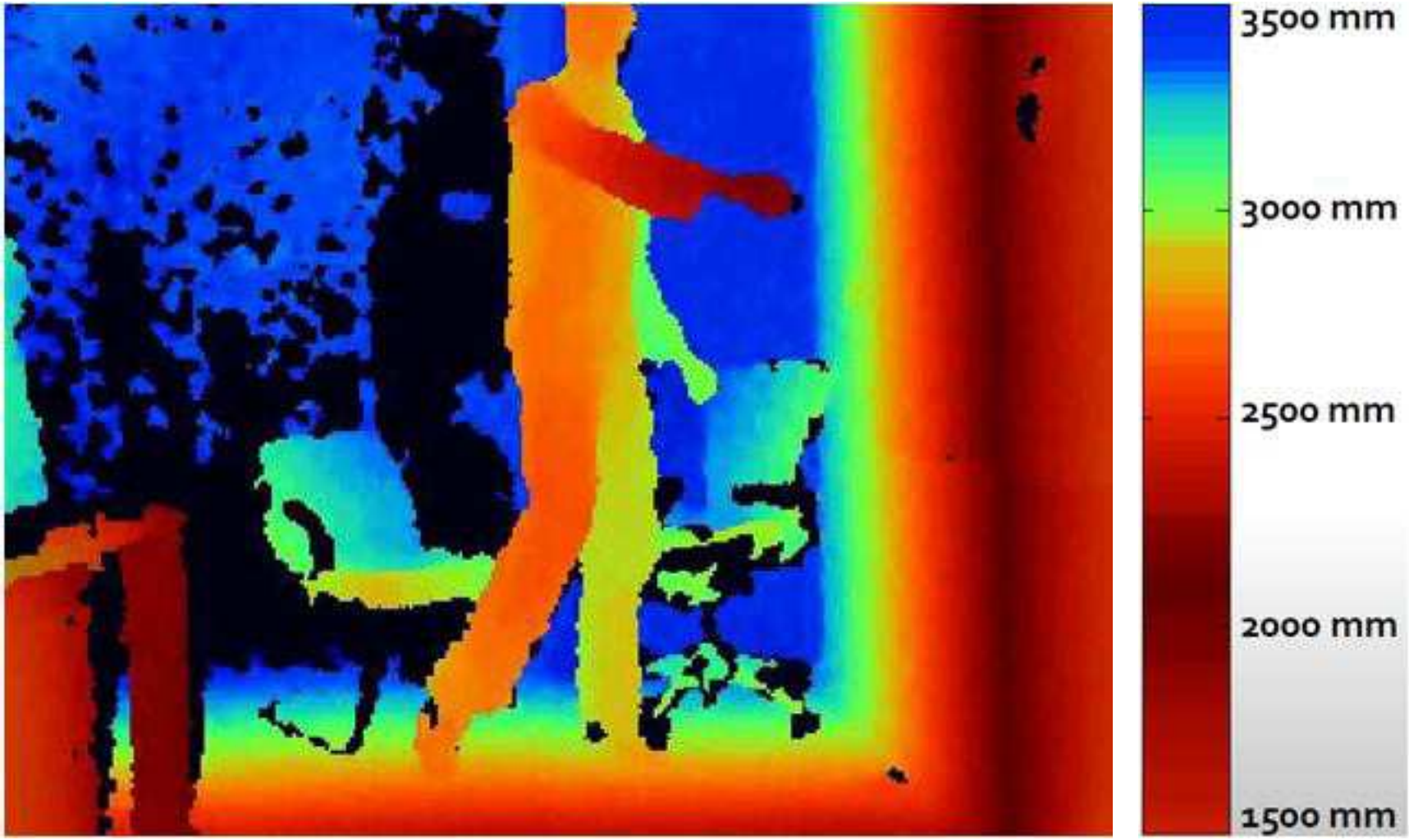
Технология PrimeSense

Хитрая структурированная подсветка

Характеристики сенсора

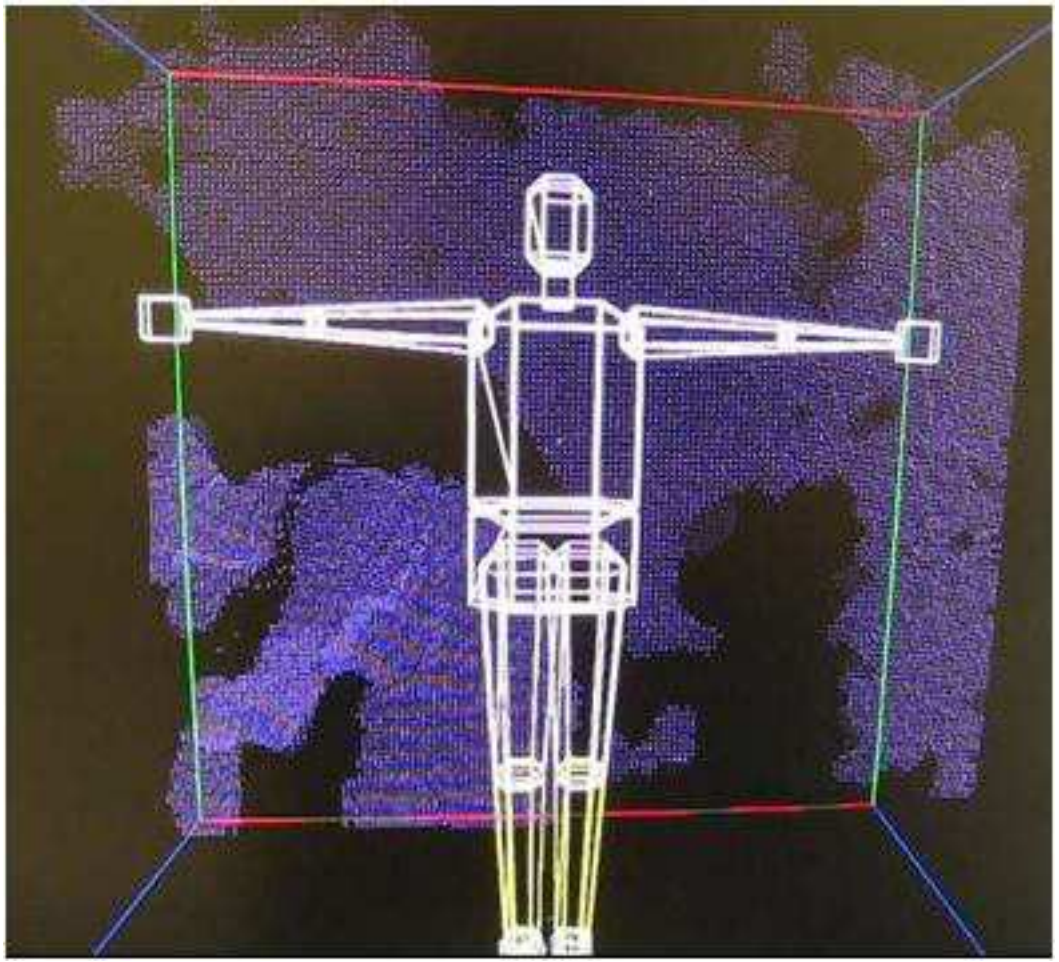
Property	Spec
Field of View (Horizontal, Vertical, Diagonal)	58° H, 45° V, 70° D
Depth image size	VGA (640x480)
Spatial x/y resolution (@ 2m distance from sensor)	3mm
Depth z resolution (@ 2m distance from sensor)	1cm
Maximum image throughput (frame rate)	60fps
Operation range	0.8m - 3.5m
Color image size	UXGA (1600x1200)
Audio: built-in microphones	Two mics
Audio: digital inputs	Four inputs
Data interface	USB 2.0
Power supply	USB 2.0
Power consumption	2.25W
Dimensions (Width x Height x Depth)	14cm x 3.5cm x 5cm
Operation environment (every lighting condition)	Indoor
Operating temperature	0°C - 40°C

Карта глубины

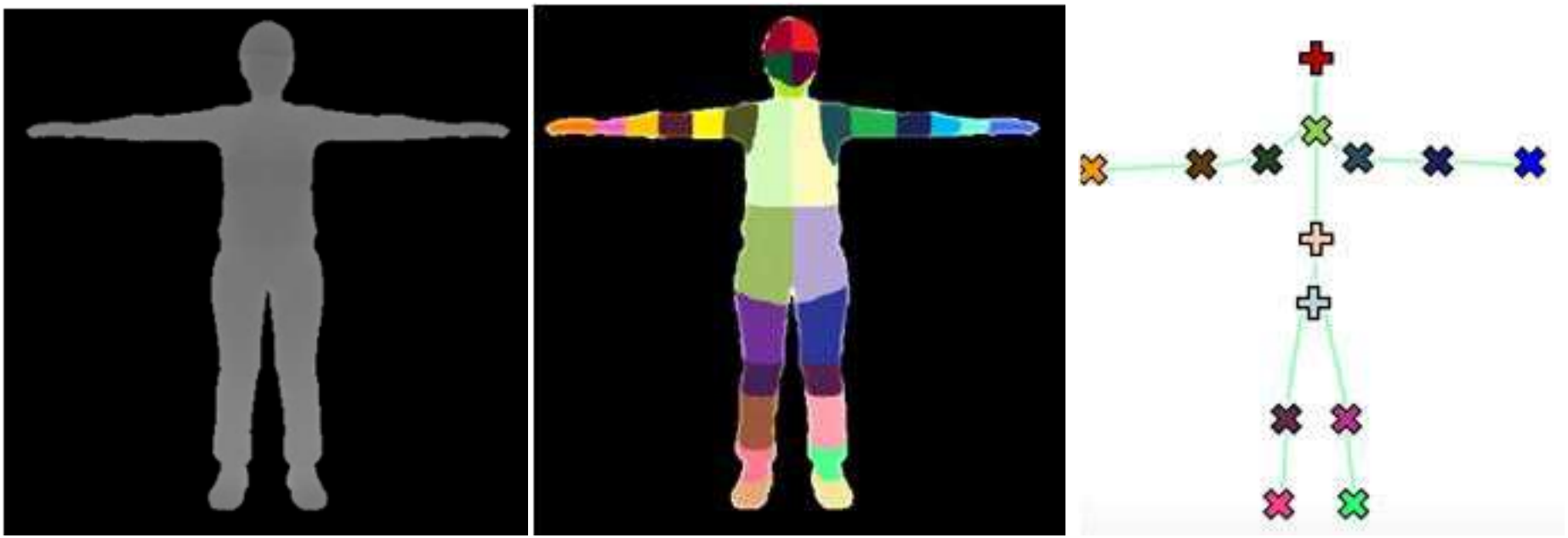


Прототипы

- Прототип, Сентябрь 2008
- Очень неплох:
  - Реальное время
  - Точные
  - Разные позы
- Но....
- Требуется инициализация



Идея решения задачи



Сформулируем задачу не как задачу оценки позы, а как задачу попиксельной разметки изображения человека на части тела

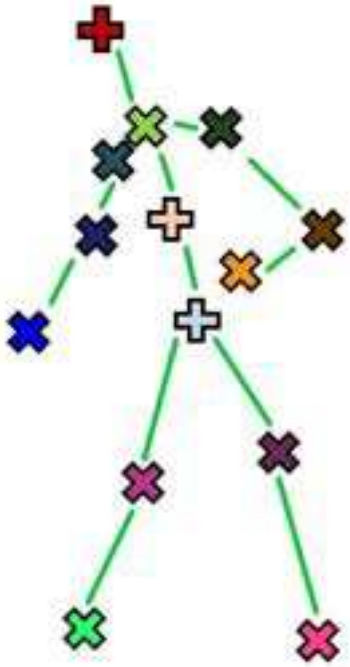
- Всего определили 31 часть
- Меньше частей оказывалось хуже!

J. Shotton et al. Real-Time Human Pose Recognition in Parts from Single Depth Images, CVPR 2011

Данные для обучения - Мосар

- Реальная съемка в домашних условиях
  - Никогда исследователям не предоставлялась!
  - Валидационная выборка
- Дополнительная съемка и использование стандартных баз движения человека

Получение данных

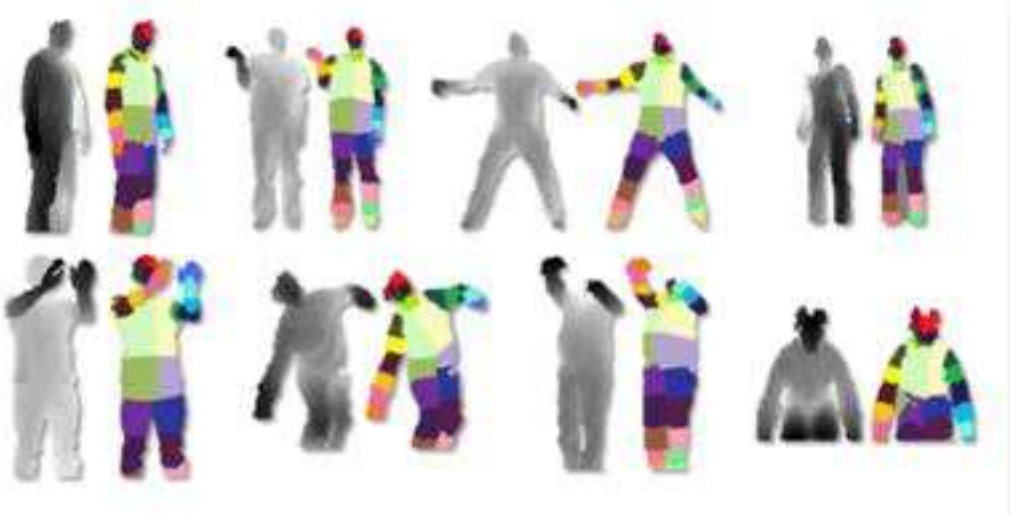


- Mocap дает позу человека
- Анимация фигуры человека с помощью ПО (MotionBuilder, например)
- Визуализация карты глубины

Получение разметки

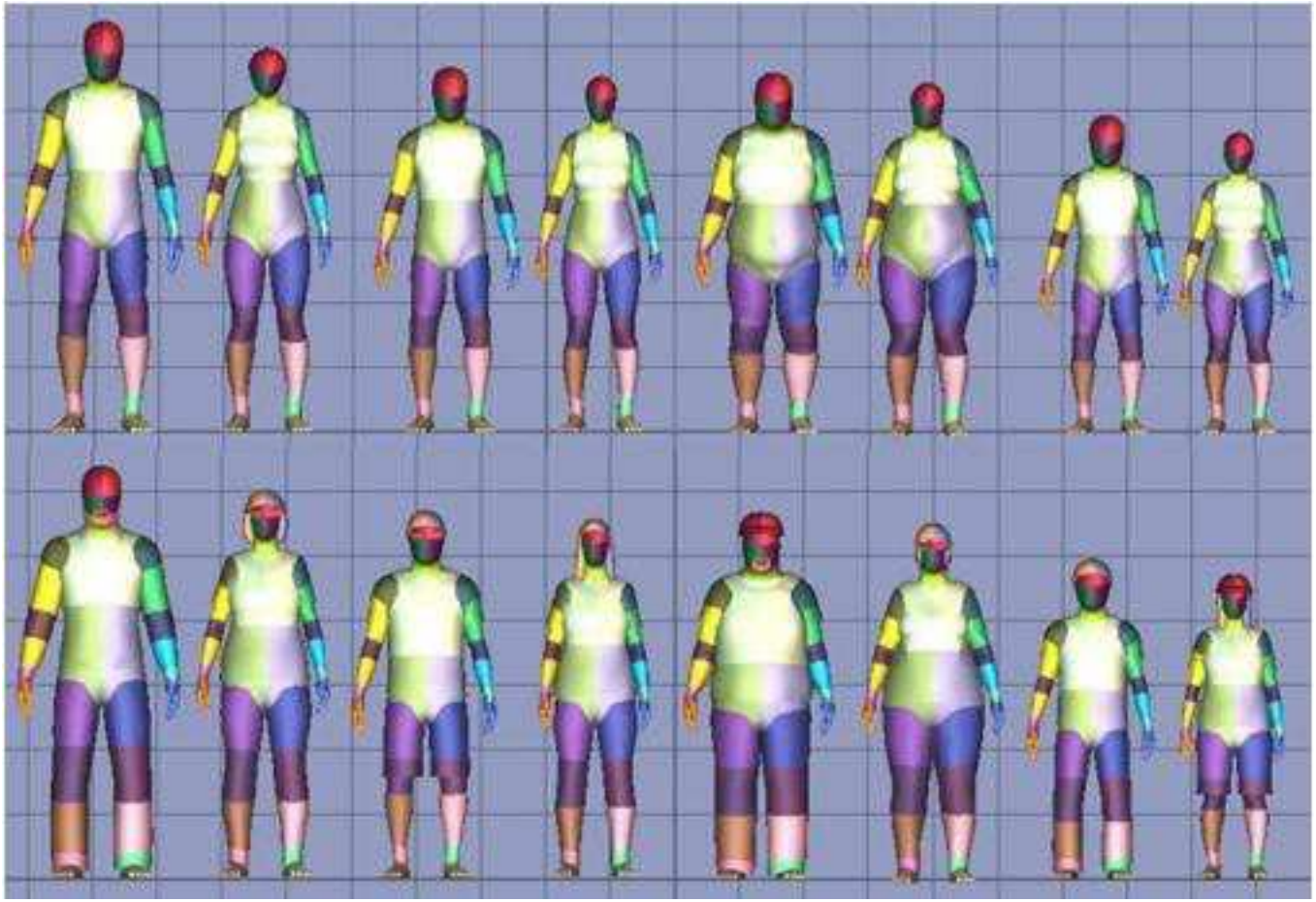
Поскольку визуализируем синтетическую модель, то можем её точно разметить

Данные Mocap



- 500к изображений из нескольких сотен видеопоследовательности
- Прореживается, отсекая ближайшие $\max_j \|p_1^j - p_2^j\|_2$
- Порог – 5 см, остаётся 100к
- Оказалось, нужно дополнительно снимать, чтобы наполнить недостаточно заполненные области пространства поз

Вариации моделей



- 15 стандартных фигур, вариации в параметрах, вариации в одежде

Реалистичность данных

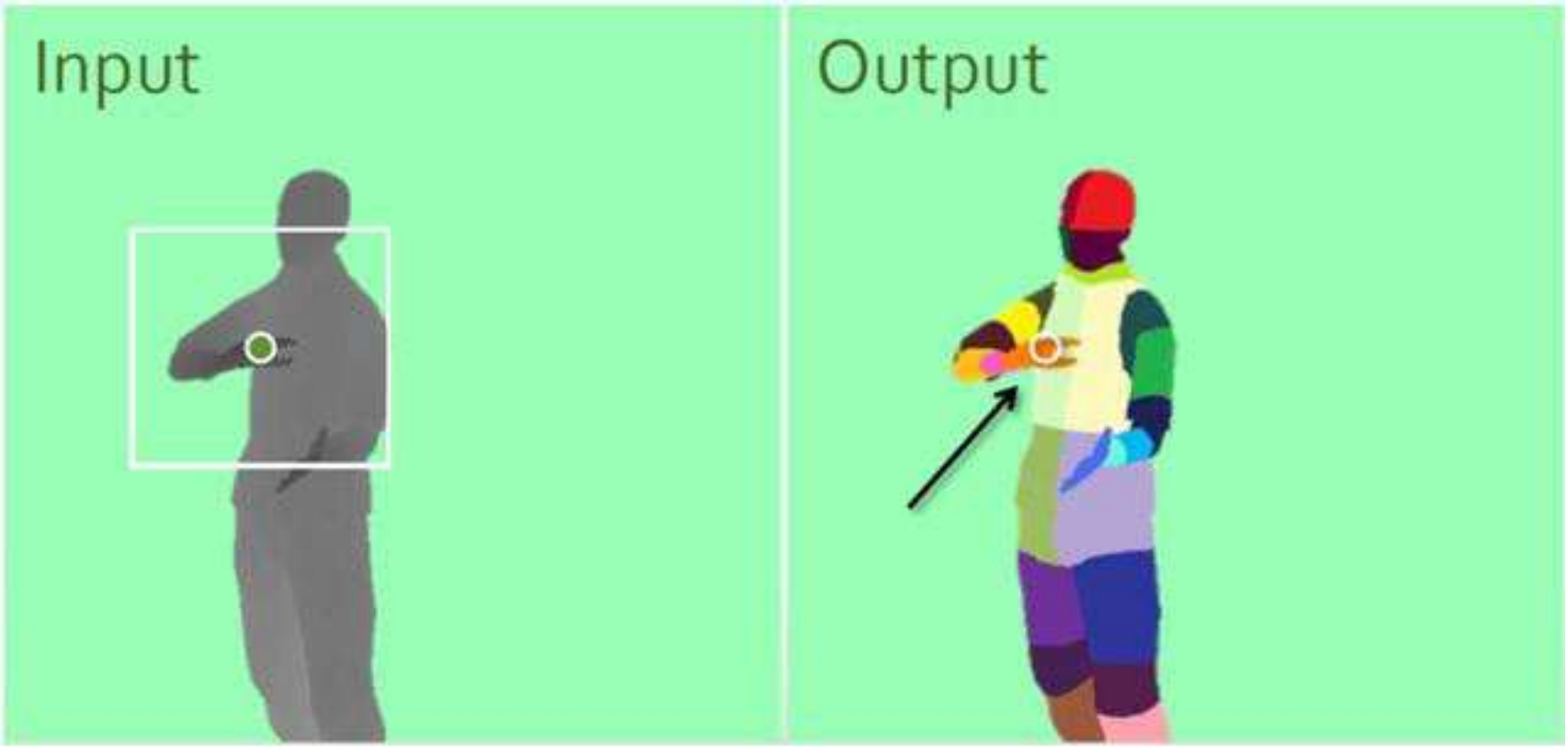
Синтетические данные

- Реалистичные
- Слишком чистые и хорошие

Искусственно испорченные:

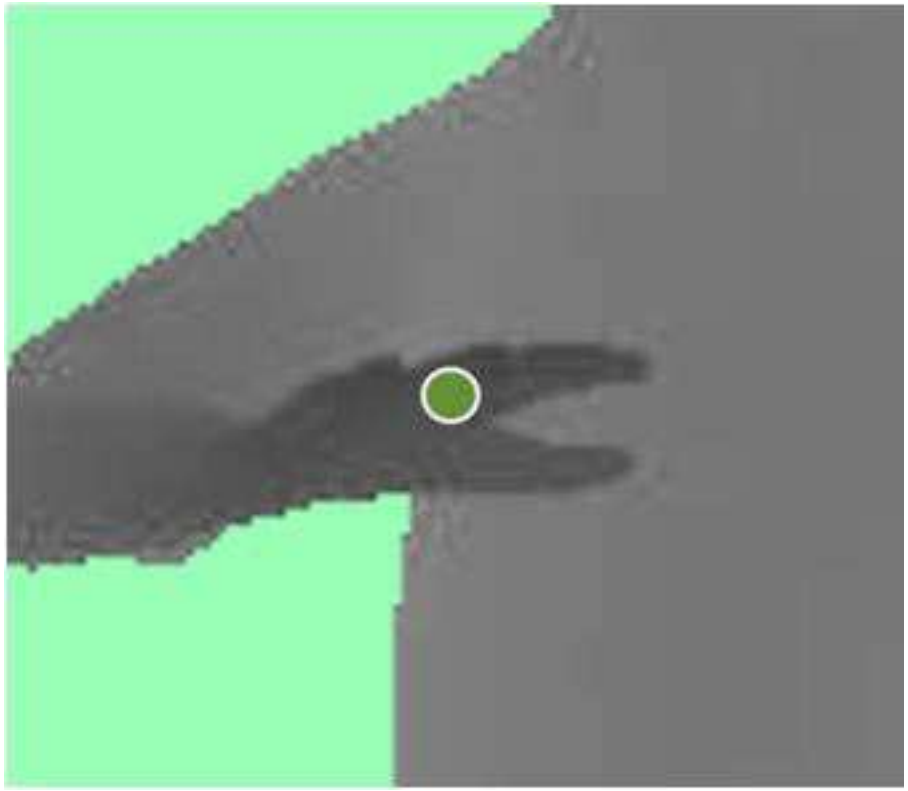
- Пропадающие пикселы на волосах
- Шумы и сниженное разрешение
- Резкие края
- Перекрытия

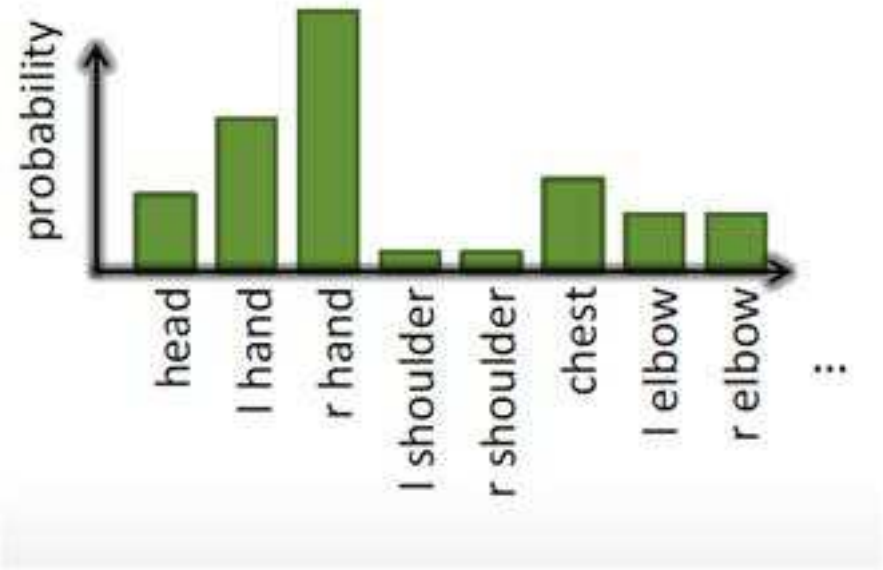
Попиксельная классификация



- Каждый пиксель классифицируется независимо от других
- Признаки вычисляются по карте глубине в некоторой окрестности

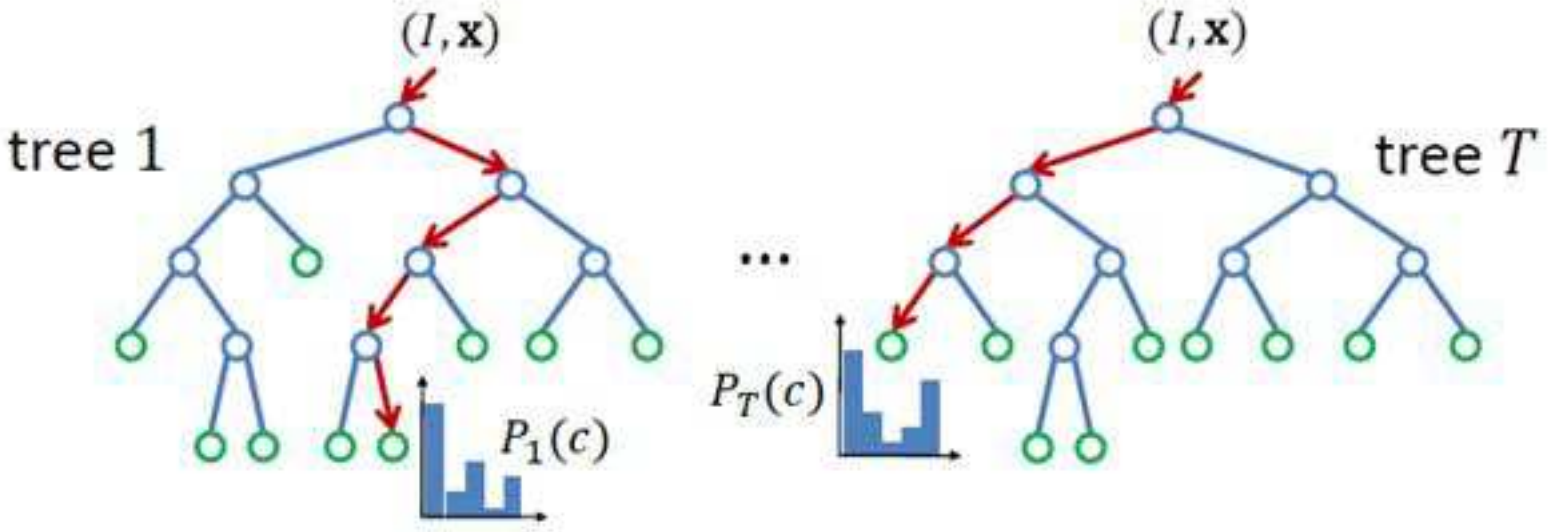
Многоклассовая классификация





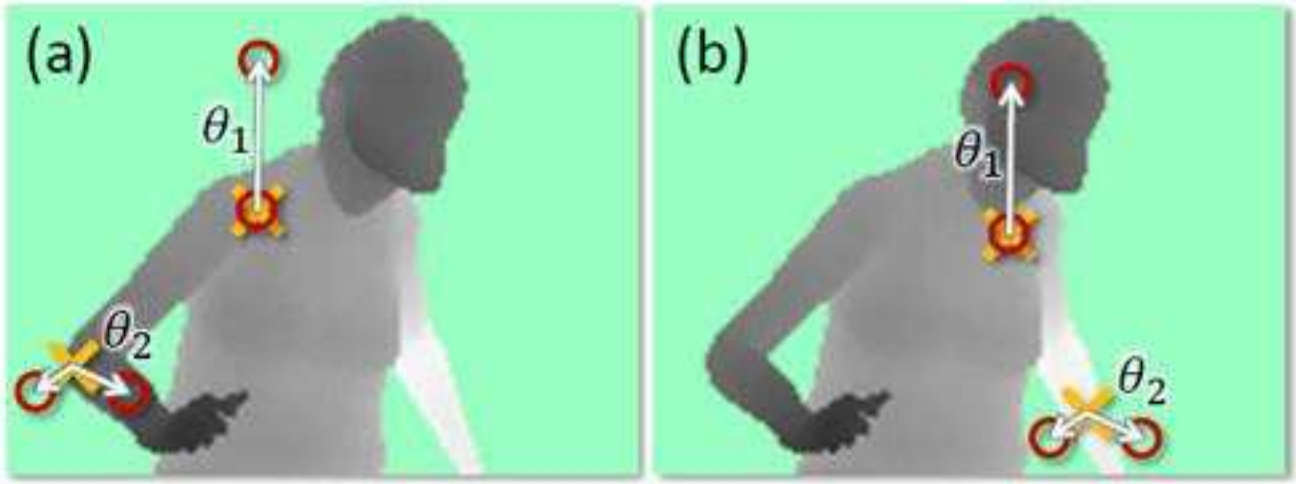
- Для каждого пикселя хотим получить вероятность каждой метки
- Для этого хорошо подойдёт случайный лес

Классификация



- Будем классифицировать каждый пиксель карты глубины с помощью случайного леса
- Параллелизация – каждый пиксель обрабатывается независимо
- Построим 3-6 деревьев глубины до 20

Признаки



$$f_\theta(I, \mathbf{x}) = d_I\left(\mathbf{x} + \frac{\mathbf{u}}{d_I(\mathbf{x})}\right) - d_I\left(\mathbf{x} + \frac{\mathbf{v}}{d_I(\mathbf{x})}\right)$$

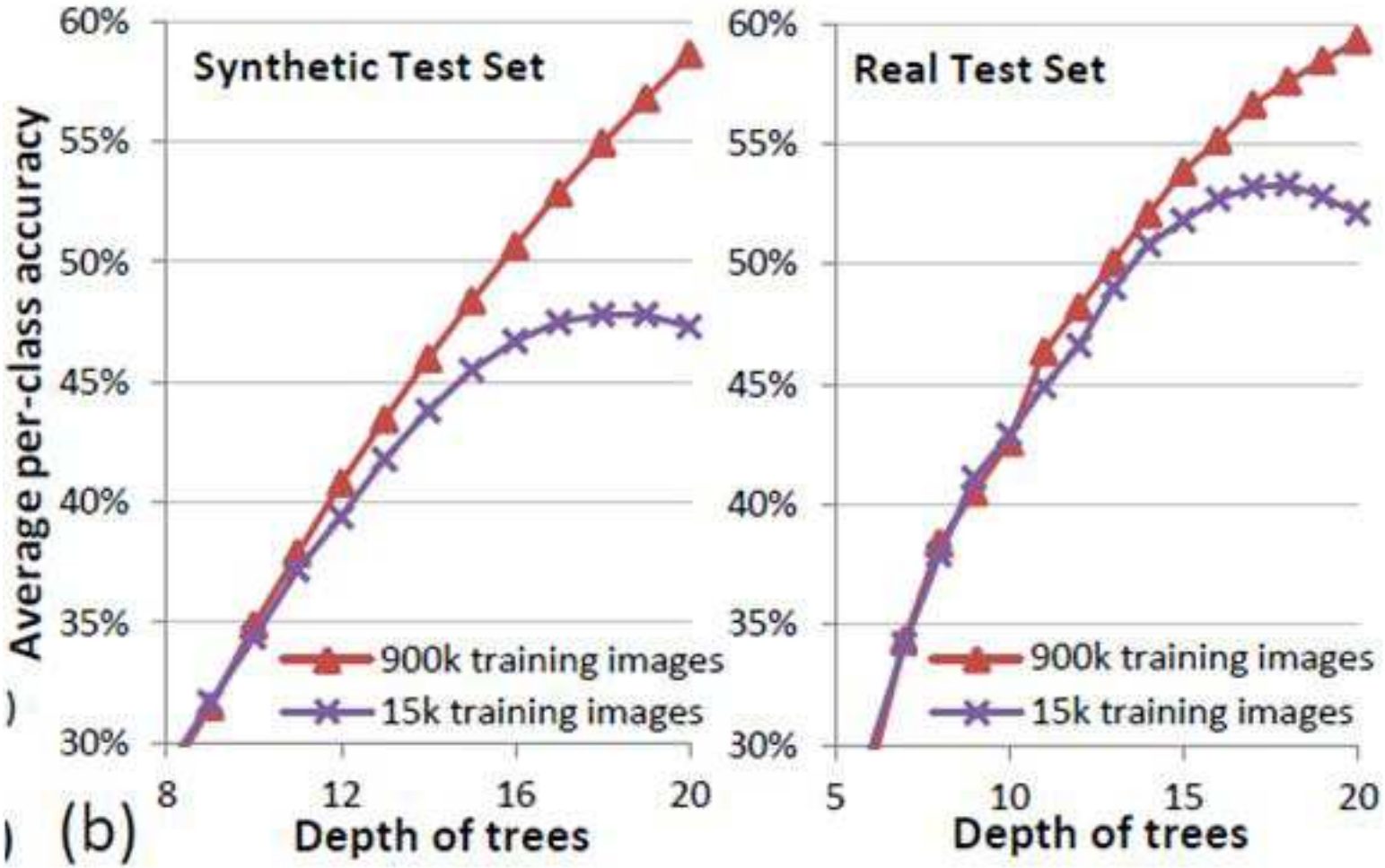
- Признак параметризуется векторами **u** и **v**
- Сверхскоростные признаки – 3 пикселя, 5 арифметических операций

Обучение и тестирование

- Обучение
  - Каждое дерево обучается на своей выборке случайно синтезированных изображений
  - С каждого изображения берется случайные 2000 пикселей
  - При обучении каждой вершины случайно генерируется набор классификаторов – признаков и порогов
  - Из набора классификаторов выбирается наилучший
  - Дерево строится до заданной глубины

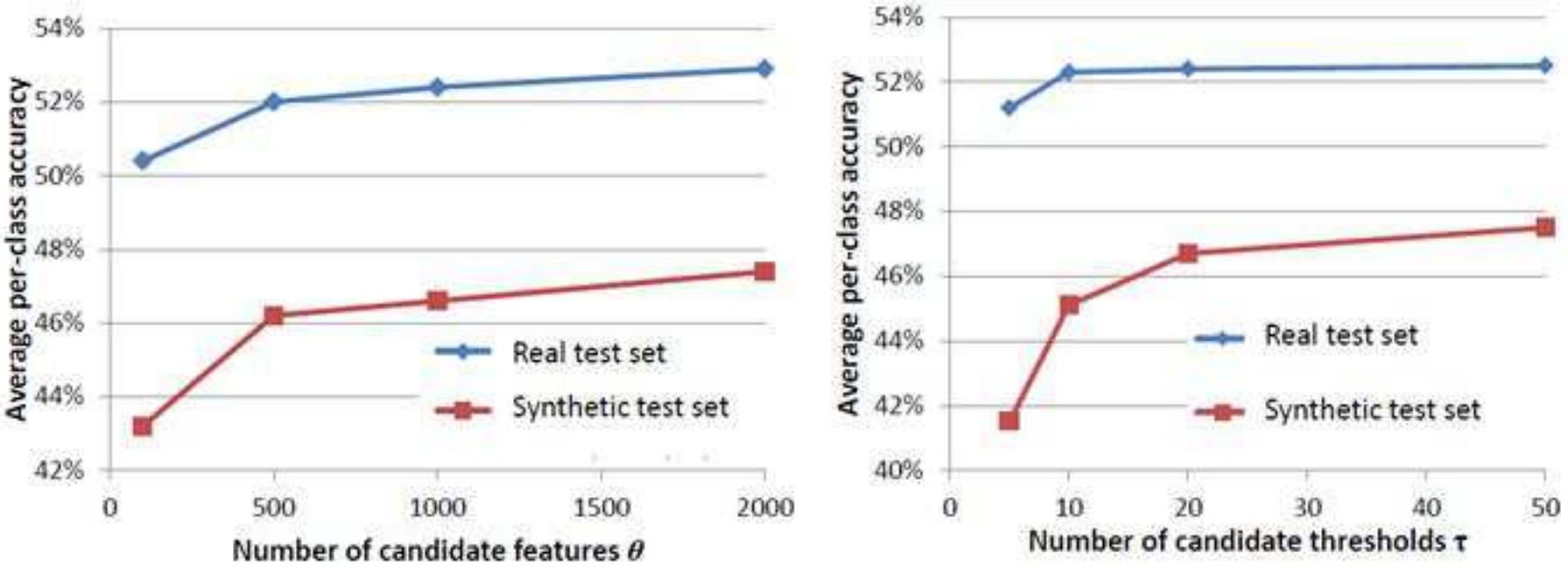
- Тестирование:
  - 5000 синтетических изображений
  - 8000 реальных изображений 15 человек

Эксперименты



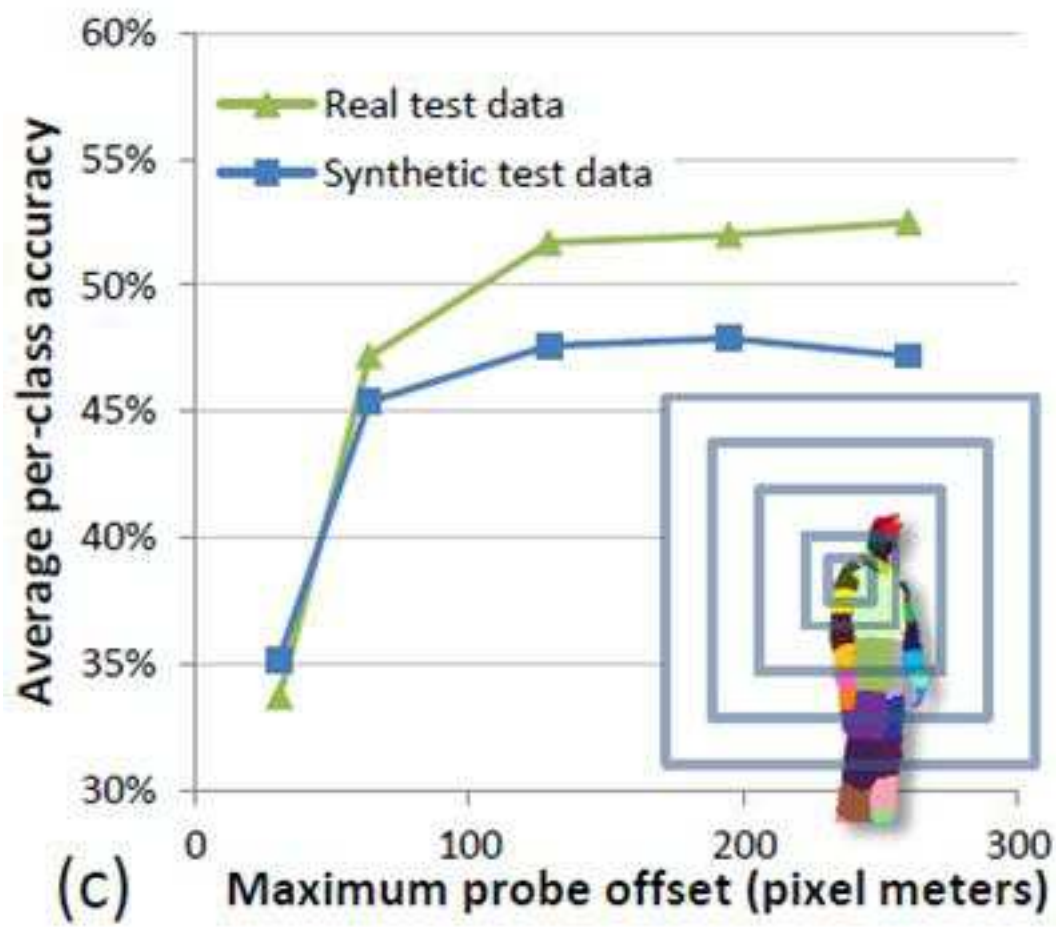
Большая синтетическая выборка позволяет справится с переобучением

Эксперименты



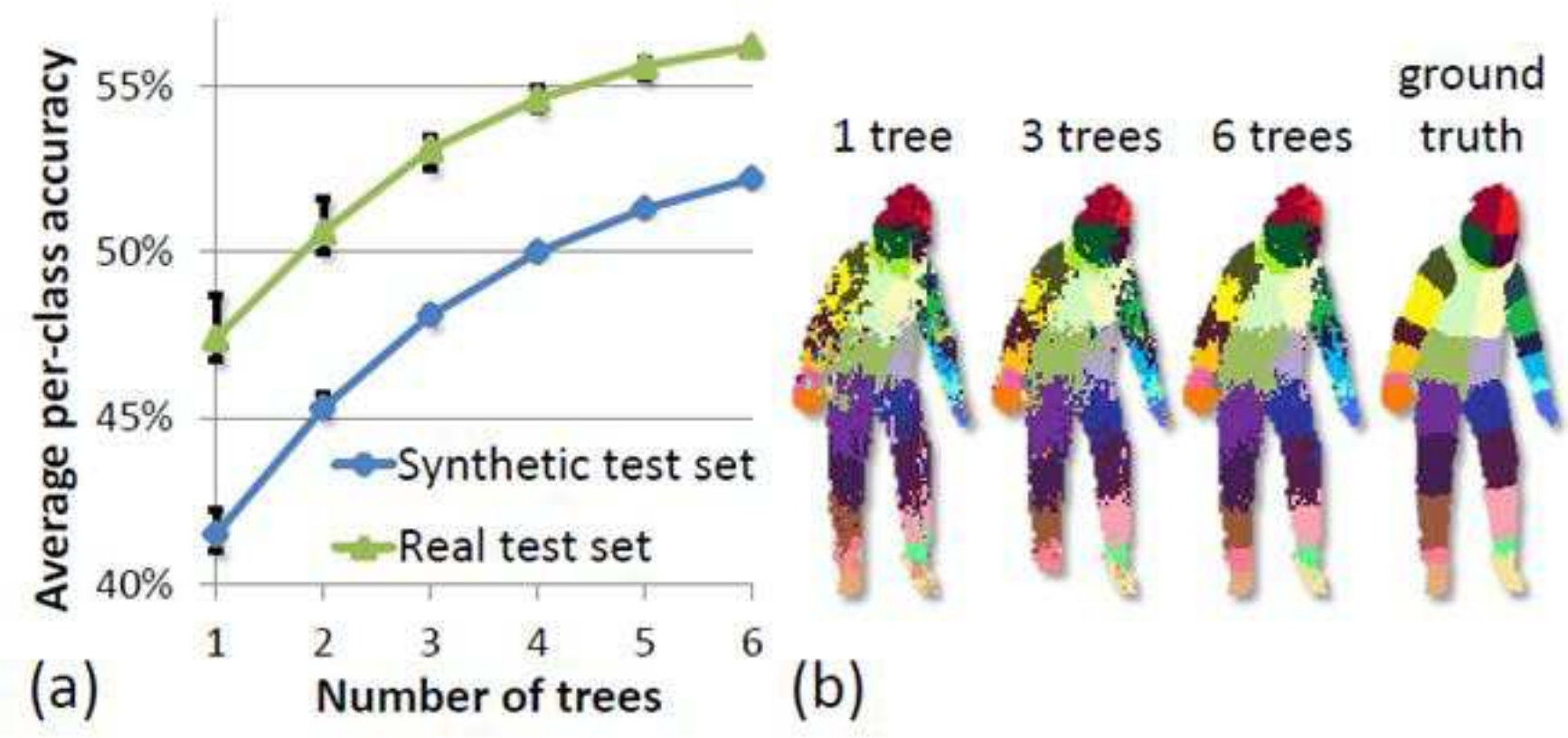
При увеличении числа проверяемых признаков и порогов быстро наступает насыщение и прекращается рост точности

Эксперименты



Для попиксельной разметки частей тела оказалось нужно анализировать только небольшую окрестность

Эксперименты

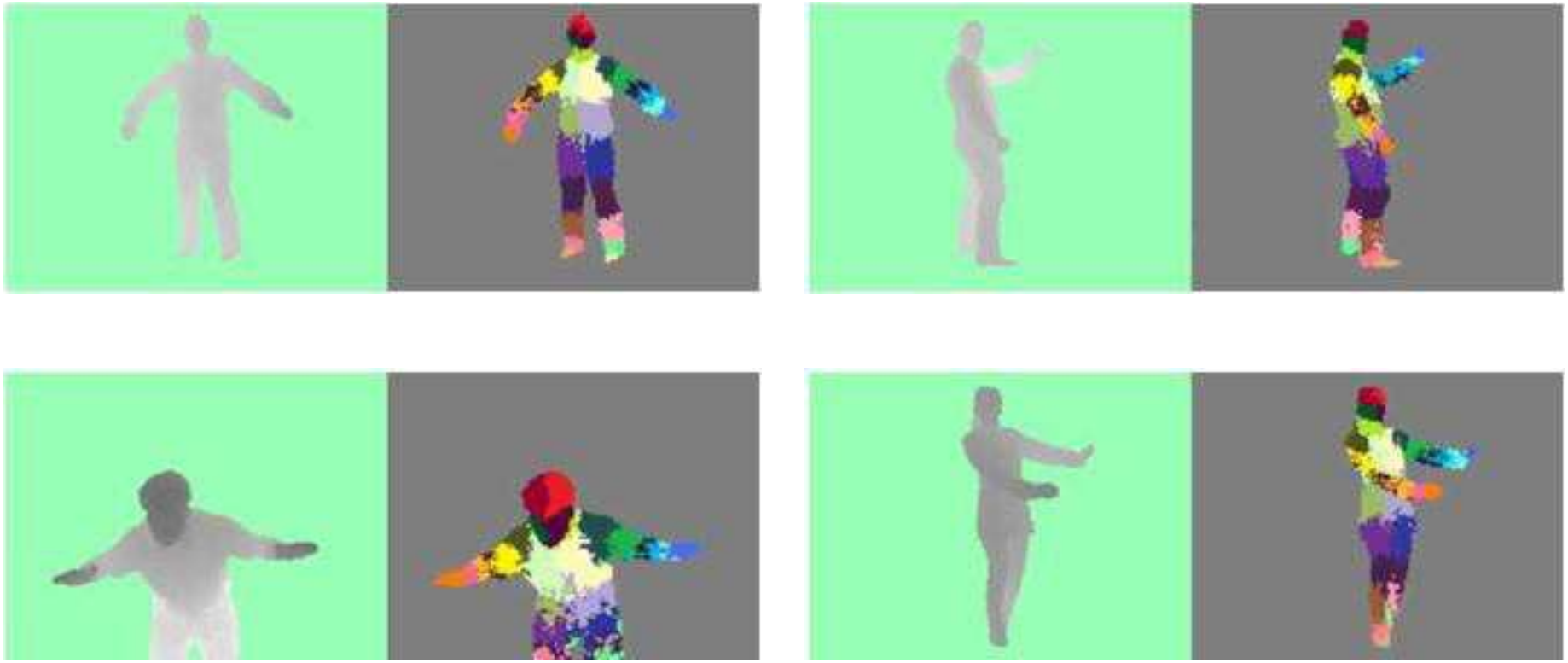


Увеличение числа деревьев повышает точность, но «вычислительная цена» больше, чем при увеличении глубины дерева

Итоговый классификатор

- 3 дерева глубины 20
- 300000 изображений на дерево
- 2000 пикселов на изображении
- 2000 кандидатов-признаков и 50 кандидатов-порогов на признак
- На базе в 1M изображений обучение занимает 1 день на 1000 ядерном кластере
- Скорость работы – 200 кадров / сек на Xbox 360
- Для работы на 30 кадрах / сек требуется всего 15% мощности Xbox 360

Примеры разметки



Оценка точек скелета

- Шаг 1 – Сглаженный поиск моды по областям

$$f_c(\hat{\mathbf{x}}) \propto \sum_{i=1}^{N} w_{ic} \exp\left(-\left\|\frac{\hat{\mathbf{x}} - \hat{\mathbf{x}}_i}{b_c}\right\|^2\right) \quad w_{ic} = P(c|I, \mathbf{x}_i) \cdot d_I(\mathbf{x}_i)^2$$

  - В результате найденные точки (моды) лежат на поверхности тела

- Шаг 2 – сдвиг точек «внутрь тела» вдоль луча от камеры
  - Обучение на 5000 изображениях по сетке

- Шаг 3 – поиск «суставов» для построения скелета

- Шаг 4 – учет ограничений на размеры конечностей, временная фильтрация….

Инфраструктура

http://openni.org/

http://kinectforwindows.org/

- Microsoft Kinect SDK
  - Для MS Kinect
- OpenNI
  - PrimeSense, WillowGarage, ASUS, Side-Kick
  - Для Asus Xtion Pro & Pro Live

Asus Xtion PRO LIVE

Резюме

- Некоторые задачи решаются простыми алгоритмами, но при очень больших объёмах данных и вычислительных мощностях
- Собрать достаточно данных невозможно, поэтому нужно дополнять реальные данные синтетическими
- Суперкомпьютеры применяются и в зрении!