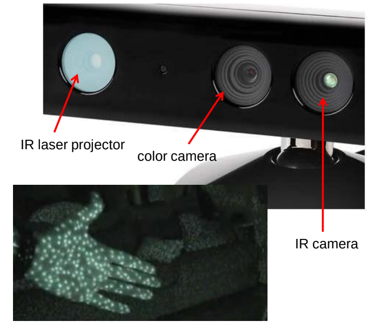
Human Body Recognition and Tracking: How the Kinect Works

Many slides by D. Hoiem

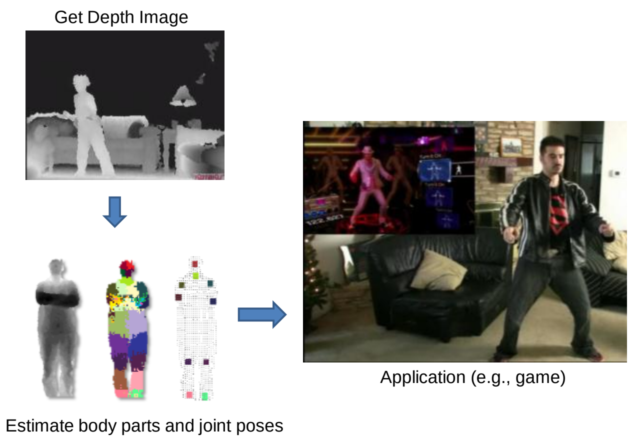
Kinect RGB-D Camera

- Microsoft Kinect (Nov. 2010)
  - Color video camera + laser-projected IR dot pattern + IR camera
- $120 (April 2012)
- Kinect 1.5 due 5/2012

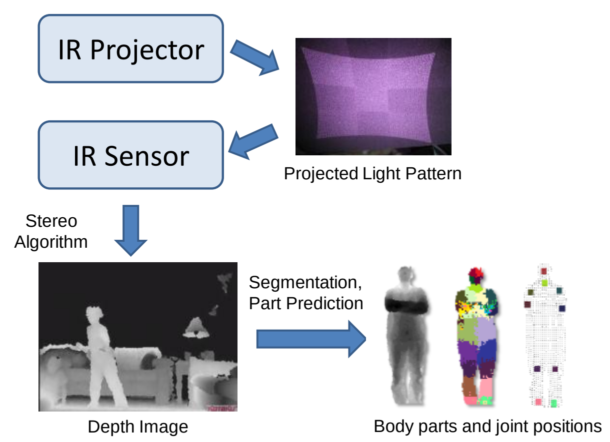
640 x 480, 30 fps



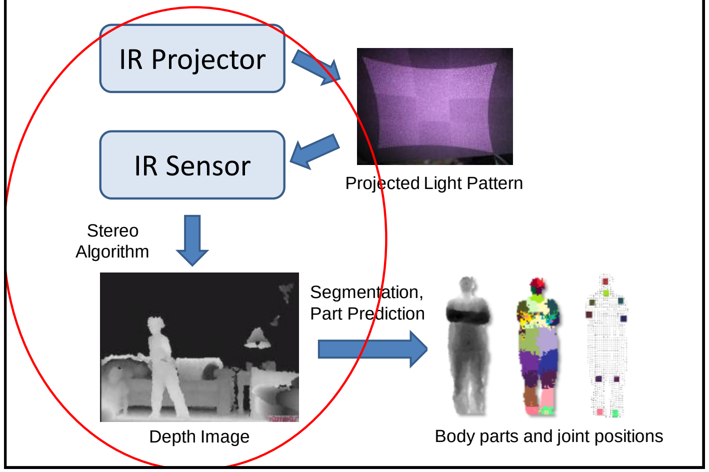
What the Kinect Does



How Kinect Works: Overview



Part 1: Stereo from Projected Dots



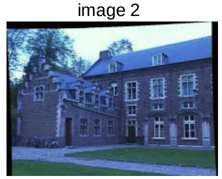
Part 1: Stereo from Projected Dots

1. Overview of depth from stereo
2. How it works for a projector/sensor pair
3. Stereo algorithm used by PrimeSense (Kinect)

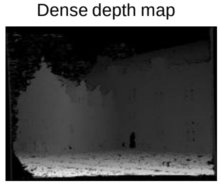
Depth from Stereo Images

image 1

image 2



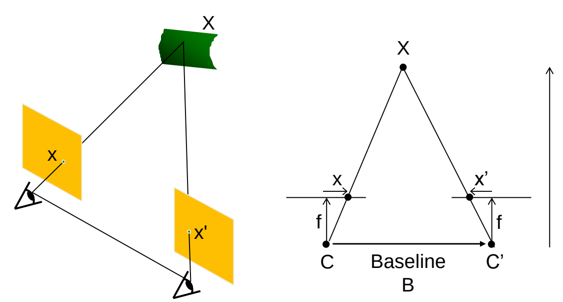
Dense depth map



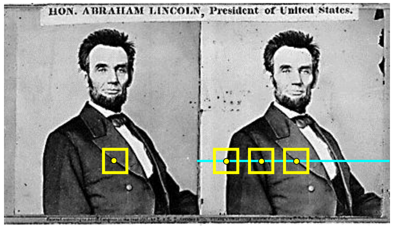
Some of following slides adapted from Steve Seitz and Lana Lazebnik

Depth from Stereo Images

- Goal: recover depth by finding image coordinate x' that corresponds to x

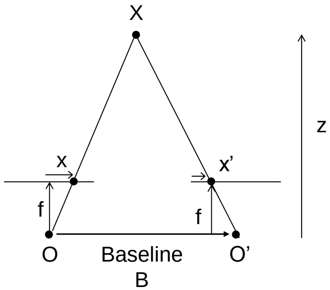


Basic Stereo Matching Algorithm



- For each pixel in the first image
  - Find corresponding epipolar line in the right image
  - Examine all pixels on the epipolar line and pick the best match
  - Triangulate the matches to get depth information

Depth from Disparity

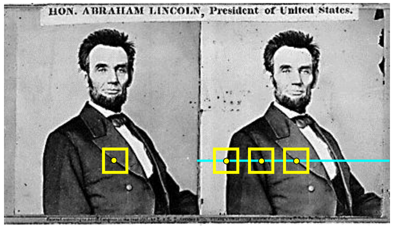


$$\frac{x - x'}{O - O'} = \frac{f}{z}$$

$$disparity = x - x' = \frac{B \cdot f}{z}$$

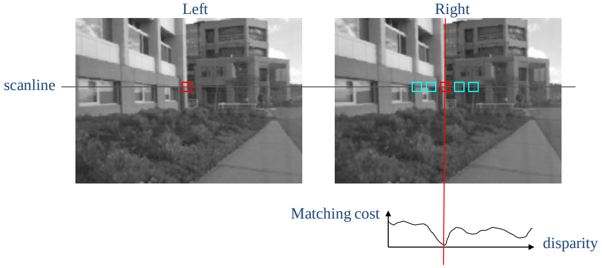
Disparity is inversely proportional to depth, *z*

Basic Stereo Matching Algorithm



- If necessary, rectify the two stereo images to transform epipolar lines into scanlines
- For each pixel x in the first image
  - Find corresponding epipolar scanline in the right image
  - Examine all pixels on the scanline and pick the best match x'
  - Compute disparity x-x' and set depth(x) = fB/(x-x')

Correspondence Search

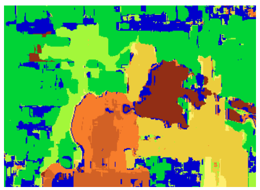


- Slide a window along the right scanline and compare contents of that window with the reference window in the left image
- Matching cost: SSD or normalized correlation

Results with Window Search

Data

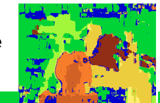
Window-based matching

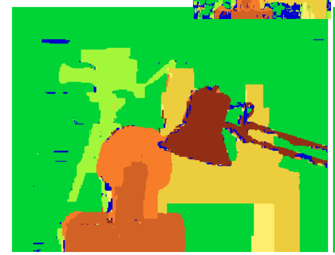


Ground truth

Add Constraints and Solve with Graph Cuts

Before





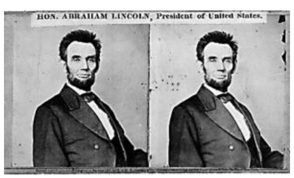
Graph cuts

Ground truth

Y. Boykov, O. Veksler, and R. Zabih, Fast Approximate Energy Minimization via Graph Cuts, PAMI 2001

For the latest and greatest: http://www.middlebury.edu/stereo/

Failures of Correspondence Search



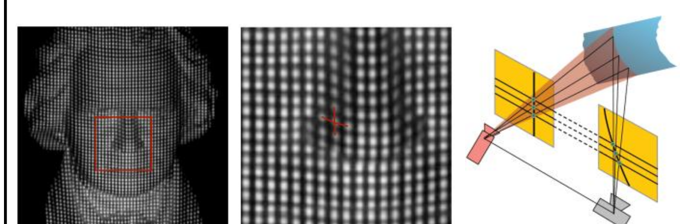
Textureless surfaces

Occlusions, repeated structures

Non-Lambertian surfaces, specularities

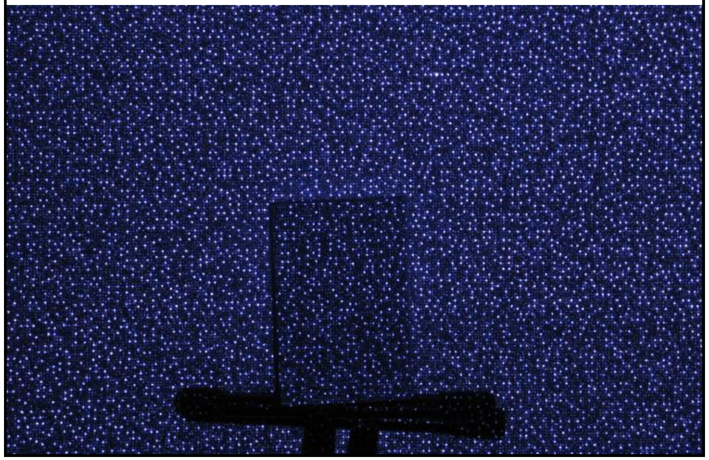
Structured Light

- Basic Principle
  - Use a **projector** to create known features (e.g., points, lines)
- Light projection
  - If we project distinctive points, matching is easy



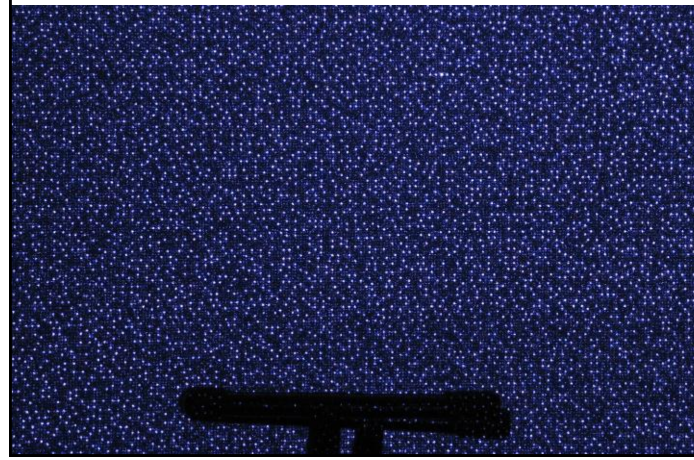
Source: http://www.futurepicture.org/?p=97

Example: Book vs. No Book

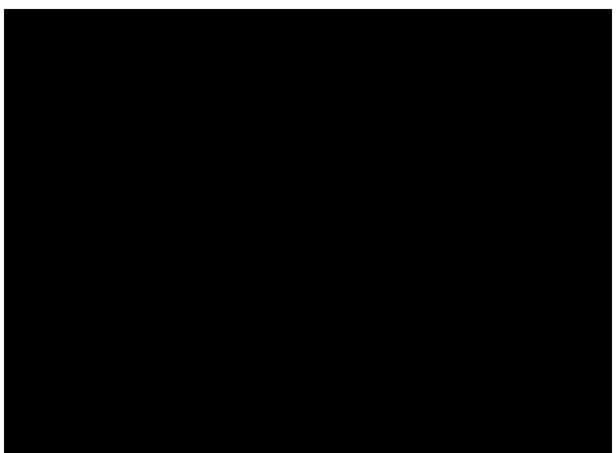


Source: http://www.futurepicture.org/?p=97

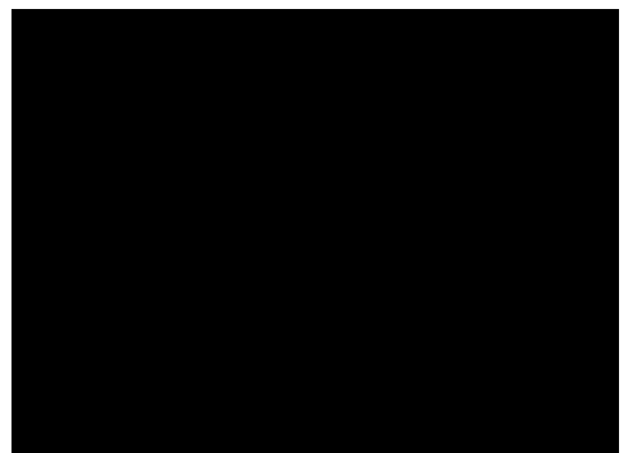
Example: Book vs. No Book



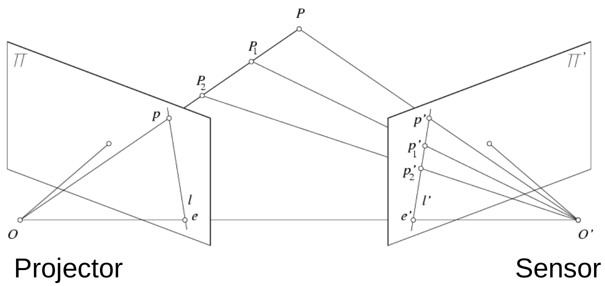
Kinect's Projected Dot Pattern



Projected Dot Pattern (2)



Same Stereo Algorithms Apply



Region-Growing Random Dot Matching

1. Detect dots ("speckles") and label them unknown
2. Randomly select a region anchor: a dot with unknown depth
   a. Windowed search via normalized cross correlation along scanline
      – Check that best match score is greater than threshold; if not, mark as "invalid" and go to 2
   b. Region growing
      1. Neighboring pixels are added to a queue
      2. For each pixel in queue, initialize by anchor's shift; then search small local neighborhood; if matched, add neighbors to queue
      3. Stop when no pixels are left in the queue
3. Stop when all dots have known depth or are marked "invalid"

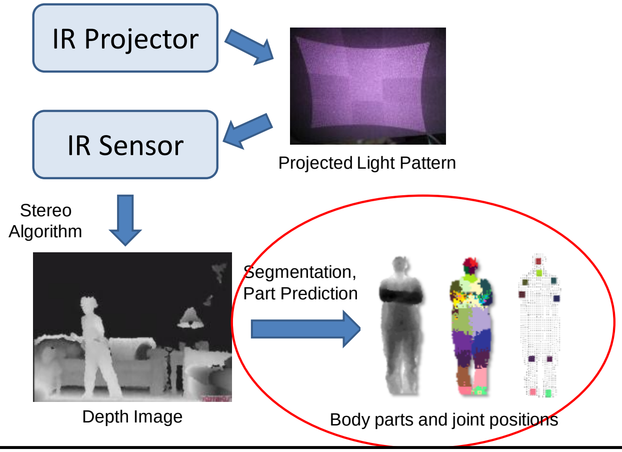
http://www.wipo.int/patentscope/search/en/WO2007043036

Kinect RGB-D Camera

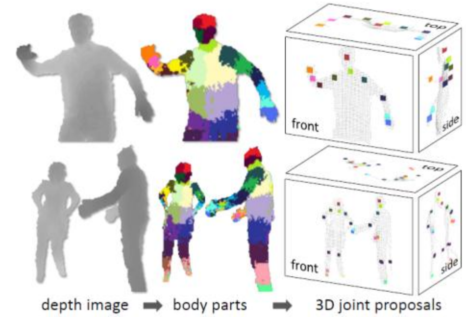
Implementation

- In-camera ASIC computes 11-bit 640 x 480 depth map at 30 Hz
- Range limit for tracking: 0.7 – 6 m (2.3' to 20')
- Practical range limit: 1.2 – 3.5 m

Part 2: Pose from Depth



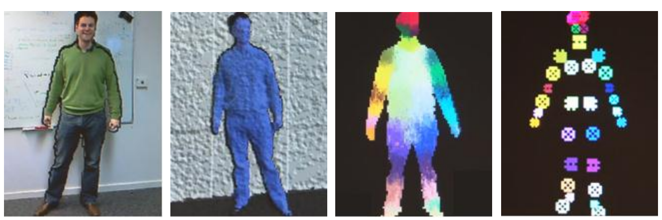
Goal: Estimate Pose from Depth Image



Real-Time Human Pose Recognition in Parts from a Single Depth Image, J. Shotton, A. Fitzgibbon, M. Cook, T. Sharp, M. Finocchio, R. Moore, A. Kipman, and A. Blake, Proc. IEEE Conf. Computer Vision and Pattern Recognition, 2011

Goal: Estimate Pose from Depth Image

Step 1. Find body parts
Step 2. Compute joint positions



RGB | Depth | Part Label Map | Joint Positions

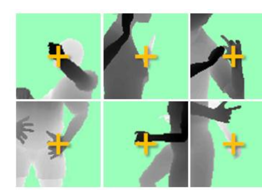
http://research.microsoft.com/apps/video/default.aspx?id=144455

Challenges

- Lots of variation in bodies, orientations, poses
- Needs to be very fast (their algorithm runs at 200 fps on the Xbox 360 GPU)

Pose Examples

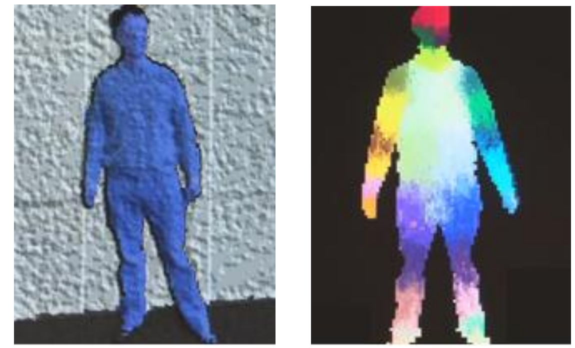
Examples of one part



Finding Body Parts

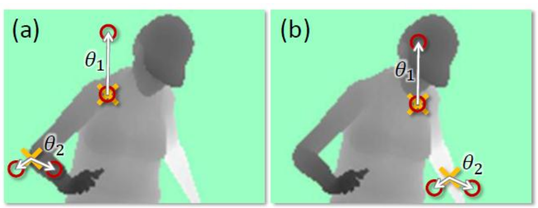
- What should we use for a feature?
  - Difference in depth
- What should we use for a classifier?
  - Random Decision Forests

Extract Body Pixels by Thresholding Depth



Features

- Difference of depth at two pixel
  - Offset is scaled by depth at reference pixel

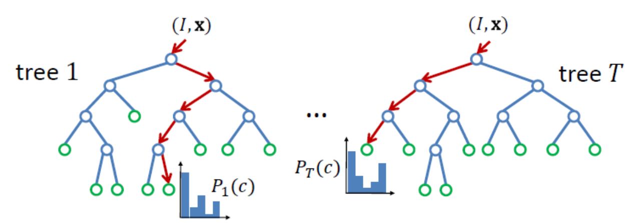


$$f_\theta(I, \mathbf{x}) = d_I\left(\mathbf{x} + \frac{\mathbf{u}}{d_I(\mathbf{x})}\right) - d_I\left(\mathbf{x} + \frac{\mathbf{v}}{d_I(\mathbf{x})}\right)$$

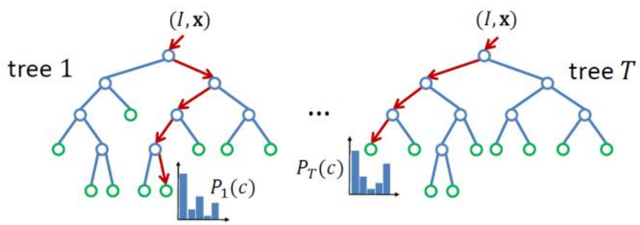
$d_I(\mathbf{x})$ is depth image, $\theta = (\mathbf{u}, \mathbf{v})$ is offset to second pixel

Part Classification with Random Forests

- **Randomized decision forest**: collection of independently-trained binary **decision trees**
- Each tree is a classifier that predicts the likelihood of a pixel **x** belonging to body part class *c*
  - Non-leaf node corresponds to a thresholded feature
  - Leaf node corresponds to a conjunction of several features
  - At leaf node store learned distribution $P(c \mid I, \mathbf{x})$



Classification



Learning Phase:

1. For each tree, pick a randomly sampled subset of training data
2. Randomly choose a set of features and thresholds at each node
3. Pick the feature and threshold that give the largest information gain
4. Recurse until a certain accuracy is reached or tree-depth is obtained

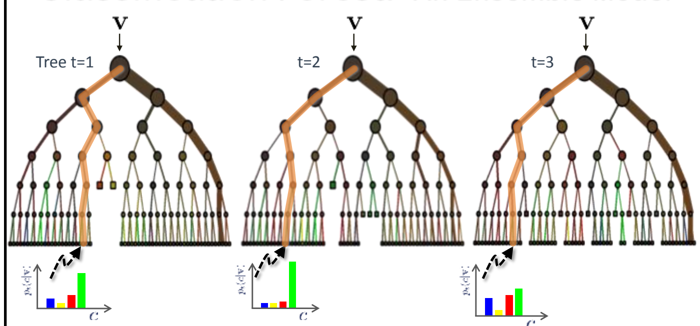
Classification

Testing Phase:

1. Classify each pixel **x** in image *I* using *all* decision trees and ***average*** the results at the leaves:

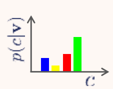
$$P(c|I, \mathbf{x}) = \frac{1}{T} \sum_{t=1}^{T} P_t(c|I, \mathbf{x})$$

Classification Forest: An Ensemble Model



The ensemble model

Forest output probability $p(c|\mathbf{v}) = \frac{1}{T} \sum_{t}^{T} p_t(c|\mathbf{v})$

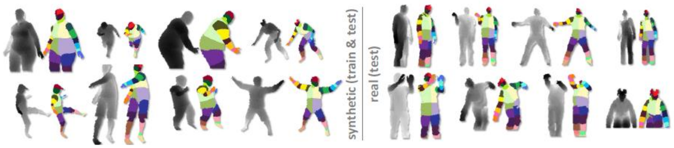


Implementation

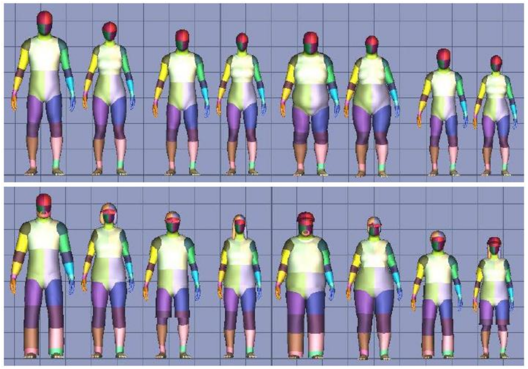
- 31 body parts
- 3 trees (depth 20)
- 300,000 training images per tree randomly selected from 1M training images
- 2,000 training example pixels per image
- 2,000 candidate features
- 50 candidate thresholds per feature
- Decision forest constructed in 1 day on 1,000 core cluster

Get Lots of Training Data

- Capture and sample 500K mocap frames of people kicking, driving, dancing, etc.
- Get 3D models for 15 bodies with a variety of weights, heights, etc.
- Synthesize mocap data for all 15 body types

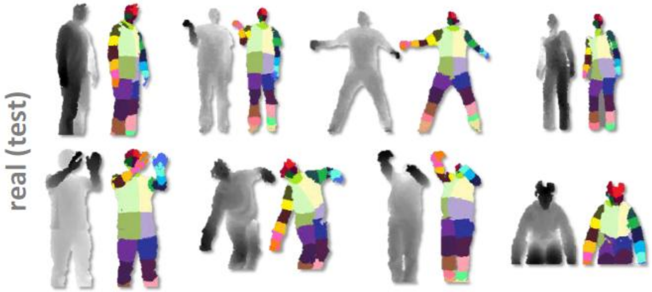


Synthetic Body Models



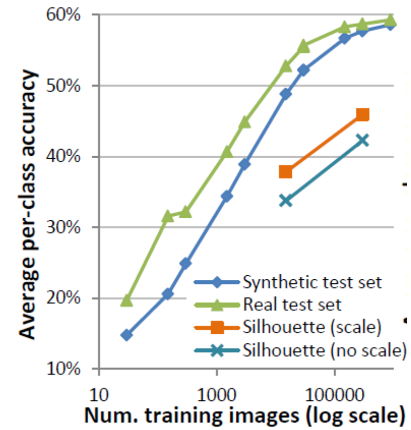
Synthetic Data for Training and Testing

Real Data for Testing

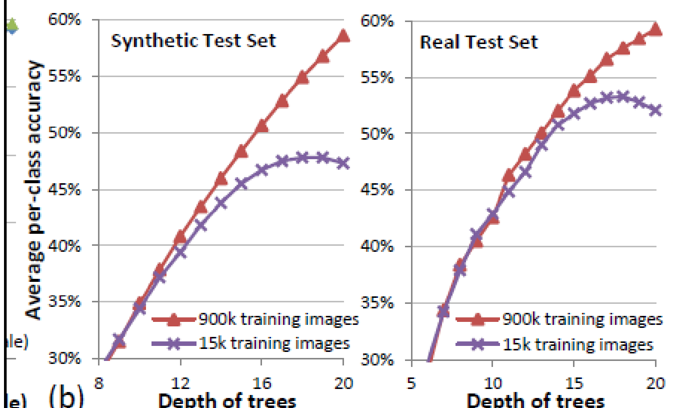


Results

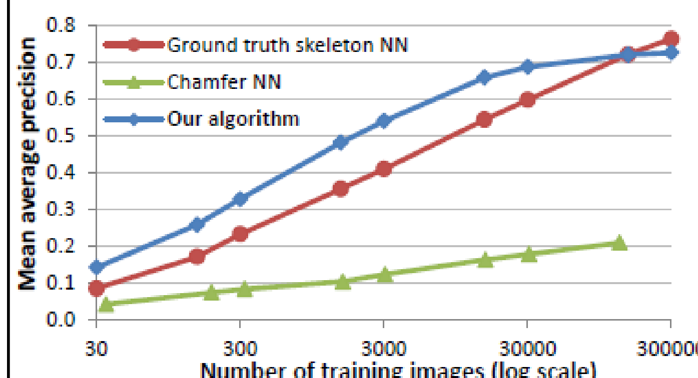
Classification Accuracy vs. # Training Examples



Classification Accuracy



Comparison with Nearest-Neighbor Matching of Whole Body



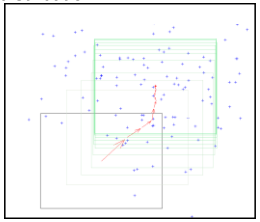
Step 2: Joint Position Estimation

- Joints are estimated using the **mean-shift clustering** algorithm applied to the labeled pixels
- Gaussian-weighted density estimator for each body part to find its mode 3D position
- "Push back in depth" each cluster mode to lie at approx. center of the body part
- 73% joint prediction accuracy (on head, shoulders, elbows, hands)

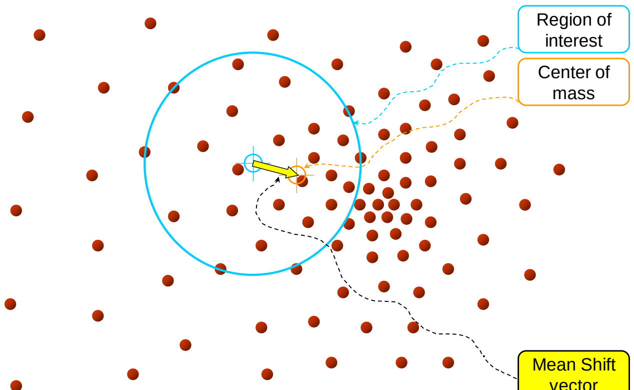
Mean Shift Clustering Algorithm

1. Choose a search window size
2. Choose the initial location of the search window
3. Compute the mean location (centroid of the data) in the search window
4. Center the search window at the mean location computed in Step 3
5. Repeat Steps 3 and 4 until convergence

The mean shift algorithm seeks the ***mode***, i.e., point of highest density of a data distribution:



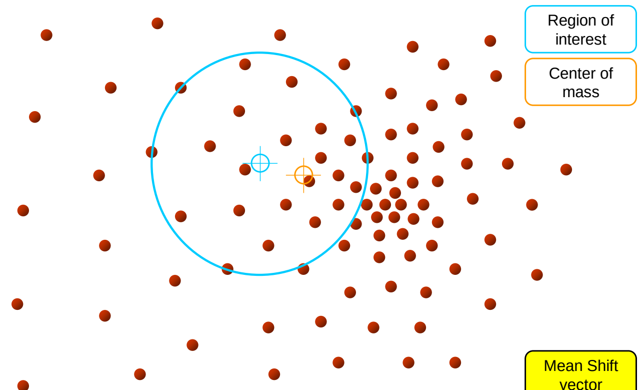
Intuitive Description



Objective : Find the densest region

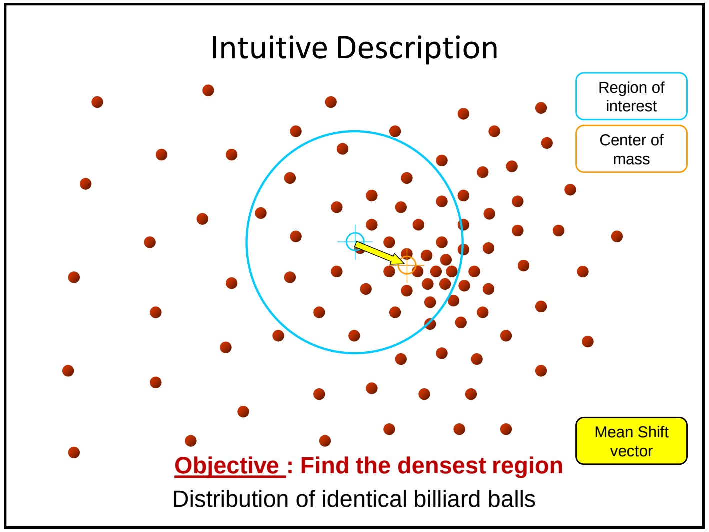
Distribution of identical billiard balls

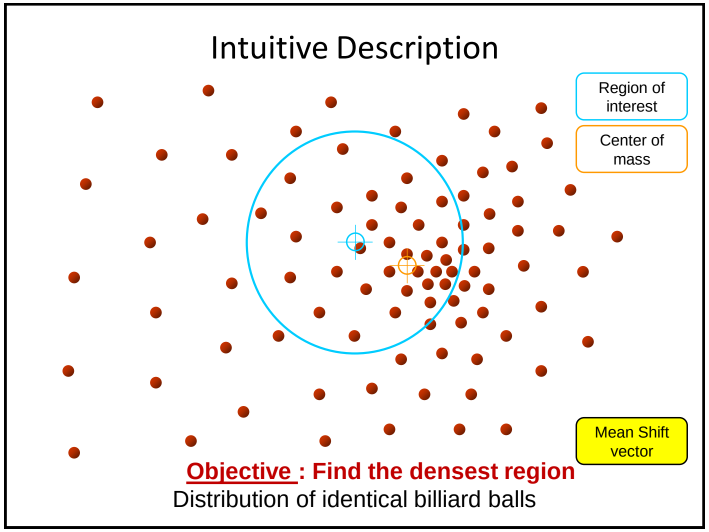
Intuitive Description

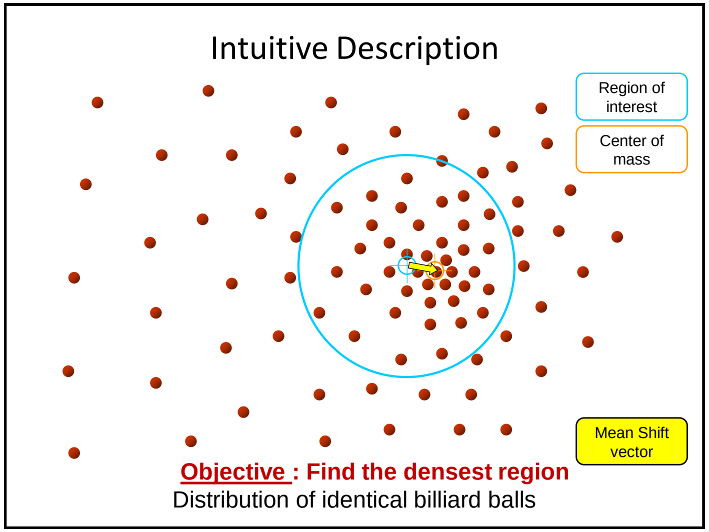


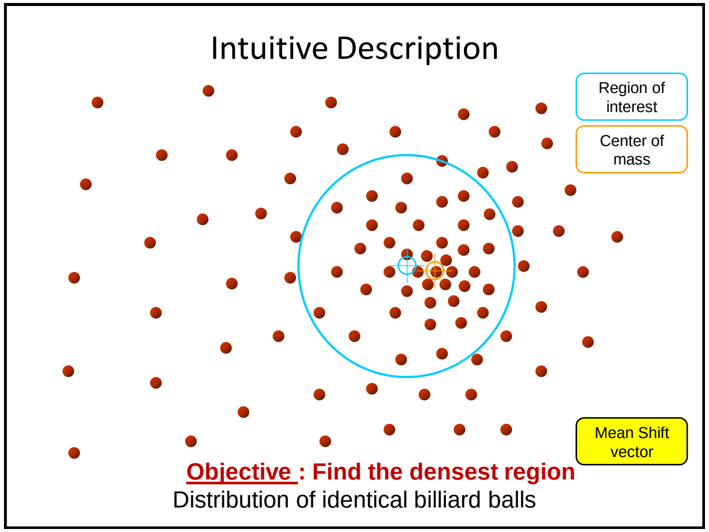
Objective : Find the densest region

Distribution of identical billiard balls

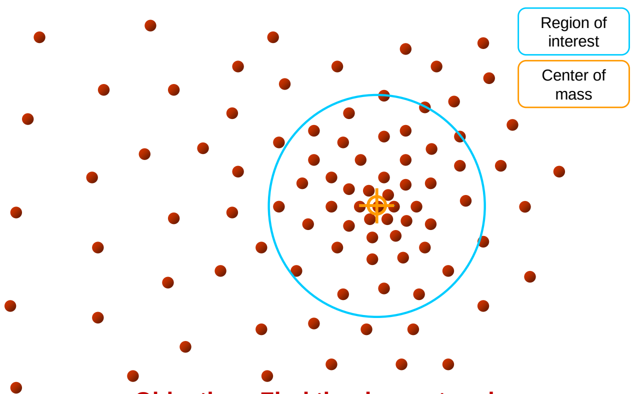

Intuitive Description
Region of interest
Center of mass
Mean Shift vector
Objective : Find the densest region
Distribution of identical billiard balls


Intuitive Description
Region of interest
Center of mass
Mean Shift vector
Objective : Find the densest region
Distribution of identical billiard balls


Intuitive Description
Region of interest
Center of mass
Mean Shift vector
Objective : Find the densest region
Distribution of identical billiard balls


Intuitive Description
Region of interest
Center of mass
Mean Shift vector
Objective : Find the densest region
Distribution of identical billiard balls

Intuitive Description



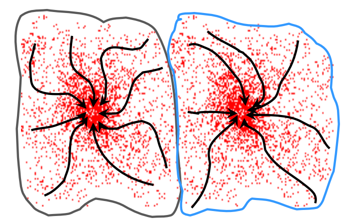
Objective : Find the densest region

Distribution of identical billiard balls

Clustering

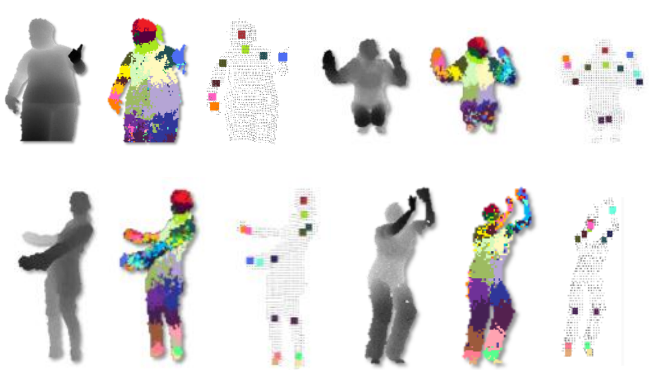
Cluster : All data points in the ***attraction basin*** of a mode

Attraction basin : the region for which all trajectories lead to the same mode

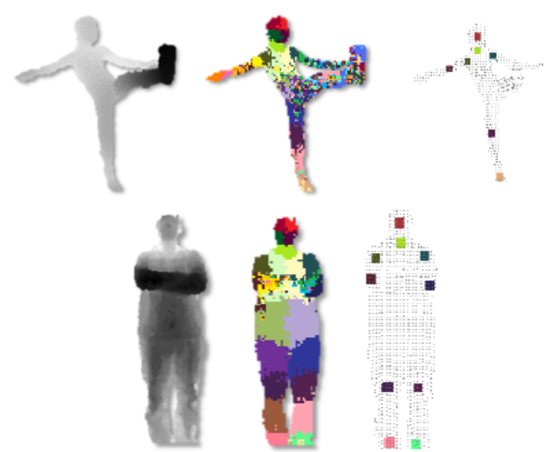


Mean Shift : A robust Approach Toward Feature Space Analysis, by Comaniciu, Meer

Results



Failures



Runtime Implementation

- Uses Xbox 360's Xenon CPU with 3 cores
- 500 MHz GPU with 10 MB DRAM

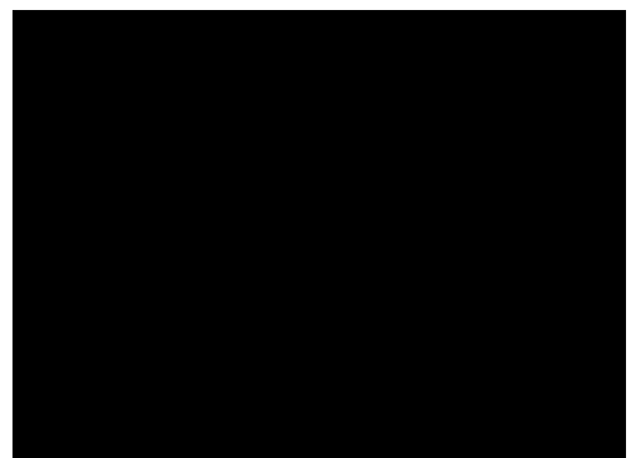
Applications

- Gesture recognition
  - http://www.youtube.com/watch?v=e0c2B3PBvRw
- Robot SLAM
  - http://www.youtube.com/watch?v=aiNX-vpDhMo
  - http://www.youtube.com/watch?v=58_xG8AkcaE
- Object recognition
  - http://www.youtube.com/watch?v=KAWLwzGdSwQ
- Nano helicoptors
  - http://www.youtube.com/watch?v=YQIMGV5vtd4

Applications

- Mario: http://www.youtube.com/watch?v=8CTJL5lUjHg
- Robot Control: http://www.youtube.com/watch?v=w8BmgtMKFbY
- Capture for holography: http://www.youtube.com/watch?v=4LW8wgmfpTE
- Virtual dressing room: http://www.youtube.com/watch?v=1jbvnk1T4vQ
- Fly wall: http://vimeo.com/user3445108/kiwibankinteractivewall
- 3D Scanner: http://www.youtube.com/watch?v=V7LthXRoESw

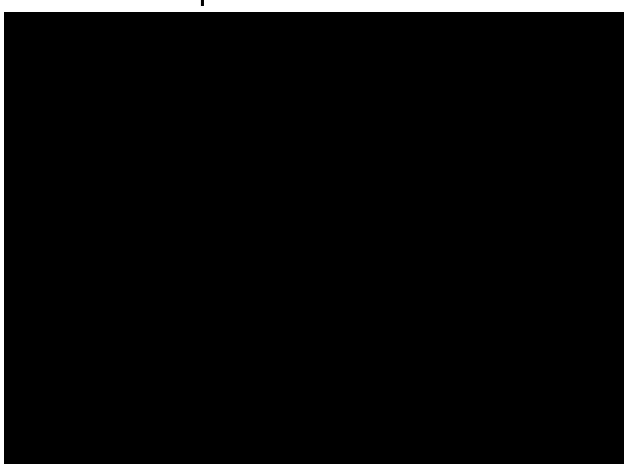
Uses of Kinect: Gesture Recognition



Uses of Kinect: Robot SLAM

Uses of Kinect: Robot SLAM (2)

Nano Helicopters



Uses of Kinect: Object Recognition