Human Body Recognition and Tracking: How the Kinect RGB-D Camera Works

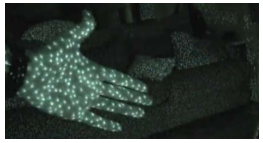
Many slides by D. Hoiem

Kinect RGB-D Camera

Microsoft "Kinect for Xbox 360"

- aka "Kinect 1" (2010)
- Color video camera + laser-projected IR dot pattern + IR camera

640 x 480, 30 fps

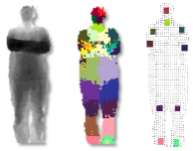


"2016 will be the year that we see interesting new applications of depth camera technology on mobile phones."

-- Chris Bishop, Director of Microsoft Research, Cambridge (2015)

What the Kinect Does

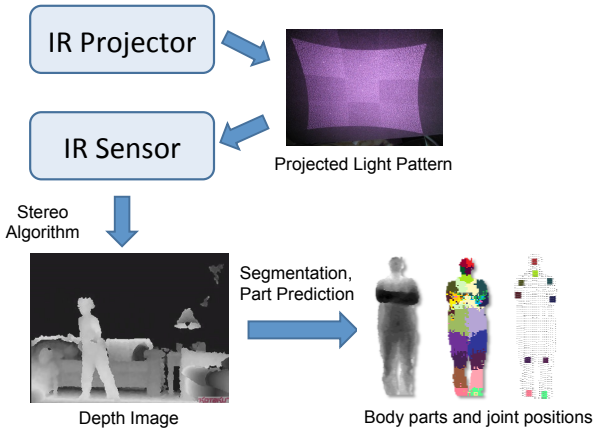
Compute Depth Image



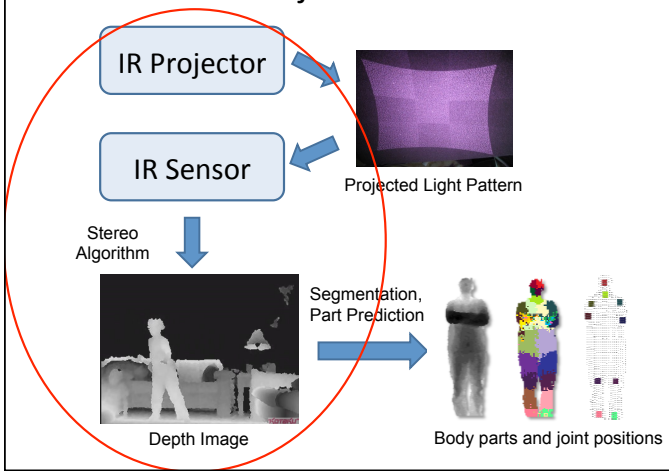
Estimate body parts and joint poses

Application (e.g., game)

How Kinect Works: Overview



Stereo from Projected Dots



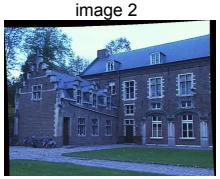
Stereo from Projected Dots

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

Depth from Stereo Images

image 1

image 2

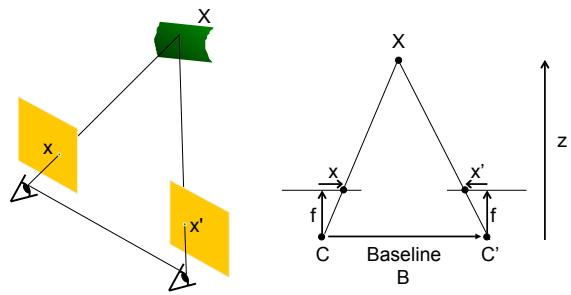


Dense depth map

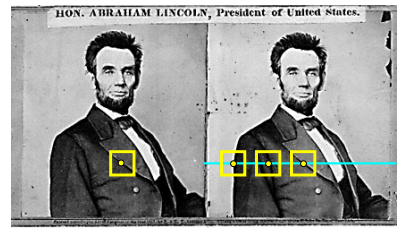
Some of following slides adapted from Steve Seitz and Lana Lazebnik

Depth from Stereo Images

- Goal: recover depth by finding image coordinate *x'* in Image 2 that corresponds to *x* in Image 1



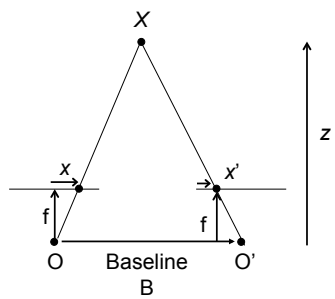
Basic Stereo Matching Algorithm



- 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

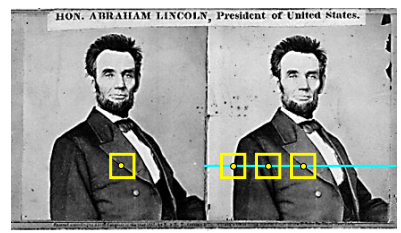
$$\frac{x-x'}{O-O'}=\frac{f}{z}$$



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

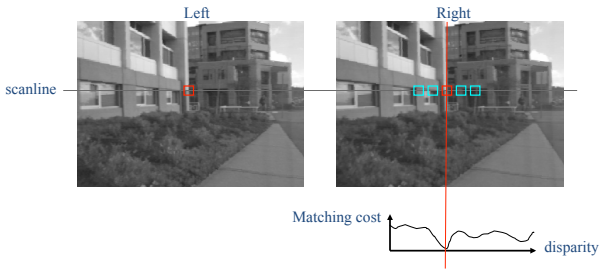
Disparity is inversely proportional to depth, *z*

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

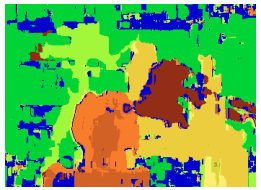


- Slide a window along the right scanline and compare contents of that window with the reference window in the left image
- Matching cost: SSD or normalized cross-correlation

Results of Window Search

Data

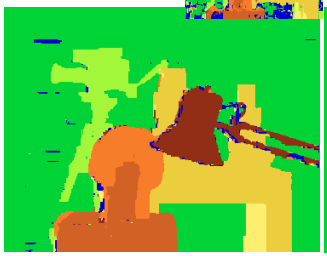
Window-based matching



Ground truth

Improve by Adding Constraints and Solve with Graph Cuts

Before



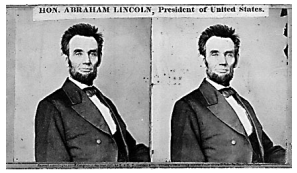
Graph cuts

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



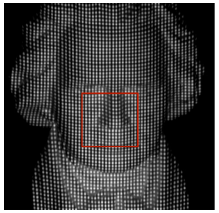
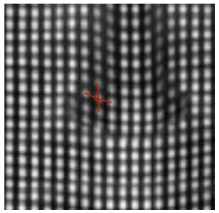
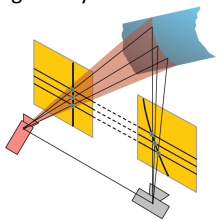
Textureless surfaces

Occlusions, repeated structures

Non-Lambertian surfaces, specularities

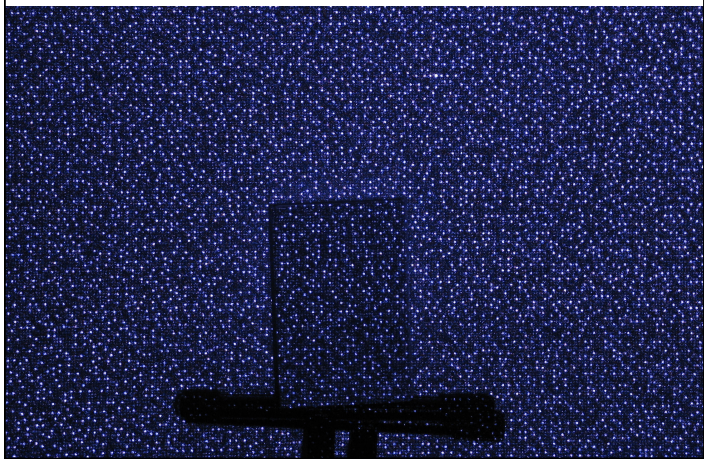
Structured Light

- Basic Principle
  - Use a **projector** to create *known features* in the 3D scene (e.g., points, lines)
- Light projection
  - If we project distinctive points, matching is easy

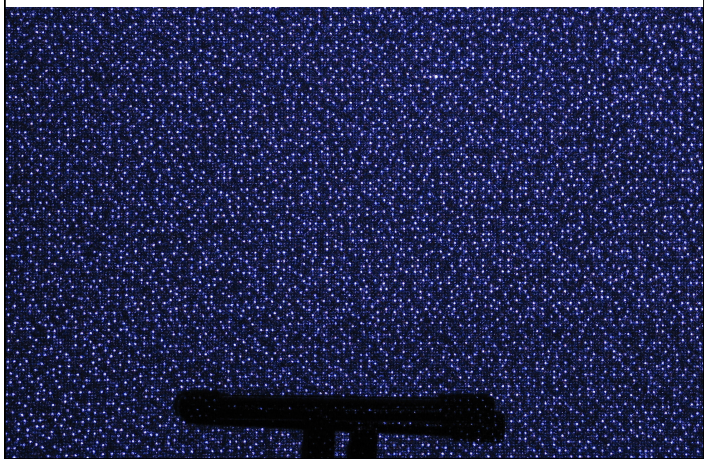
Source: http://www.futurepicture.org/?p=97

Example: Book vs. No Book



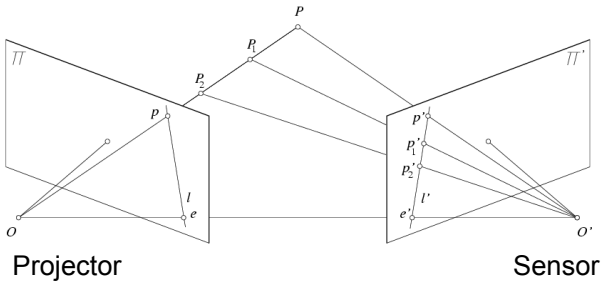
Source: http://www.futurepicture.org/?p=97

Example: Book vs. No Book

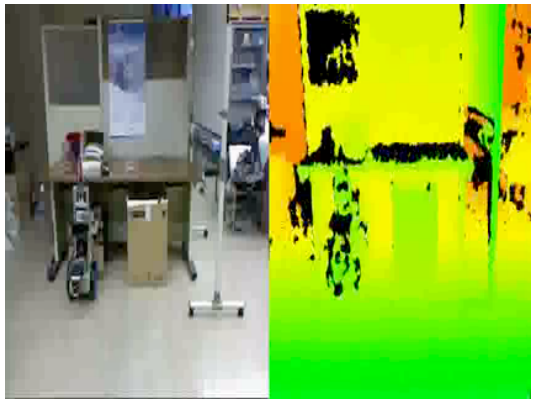


Kinect's Projected Dot Pattern

Same Stereo Algorithms Apply



Kinect RGB-D Camera

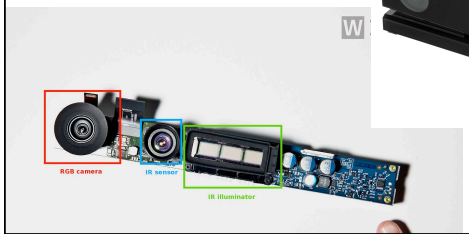


Implementation

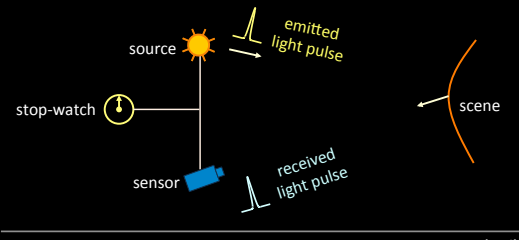
- In-camera ASIC computes 11-bit 640 x 480 depth map at 30 Hz
- Range limit for tracking: 0.7 – 6 m (2.3' to 20')
- Practical range limit: 1.2 – 3.5 m

Kinect for Xbox One

- aka "Kinect 2" (2013)
- Replaced Structured-Light Camera by Time-of-Flight Camera
- Higher resolution (1080p), larger view of view , 30 fps camera
- Depth resolution 2.5cm at 4m



Time-of-Flight Depth Sensing



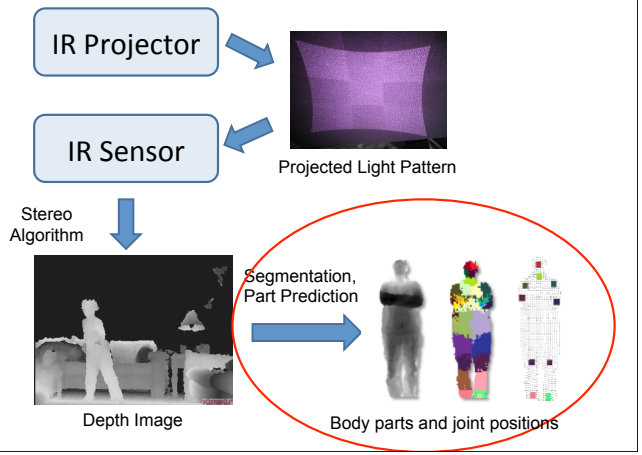
Impulse Time-of-Flight Imaging

[Koechner, 1968]

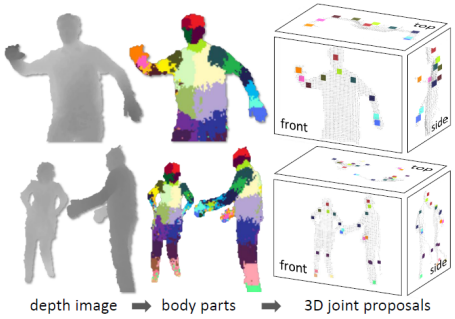
Kinect 2's Time of Flight Sensor

- Kinect 2 uses multiple measurements (3 pulse frequencies x 3 amplitudes) to compute at each pixel:
  - The amount of reflected light originating from the active light source (called the "active image")
  - The depth of the scene from the phase shifts for the multiple measurements (which disambiguate the depth)
  - The amount of ambient light

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

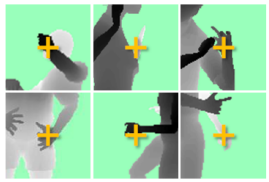
http://research.microsoft.com/apps/video/default.aspx?id=144455

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



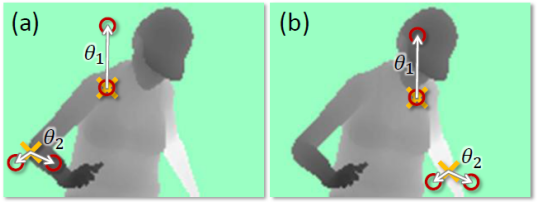
Finding Body Parts

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

Extract Body Pixels by Thresholding Depth

Features

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

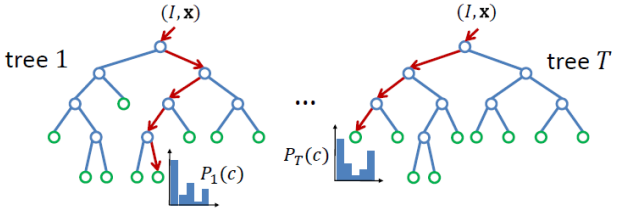


$$f_\theta(I, \mathbf{x}) = d_I\left(\mathbf{x} + \frac{\mathbf{u}}{d_I(\mathbf{x})}\right) - d_I\left(\mathbf{x} + \frac{\mathbf{v}}{d_I(\mathbf{x})}\right)$$

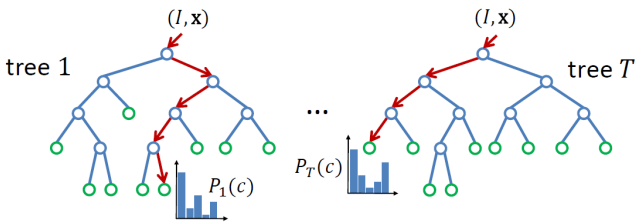
$d_I(\mathbf{x})$ is depth image, $\theta = (\mathbf{u}, \mathbf{v})$ is offset to second pixel

Part Classification with Random Forests

- **Random 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 $P(c|I, \mathbf{x})$



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:

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

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

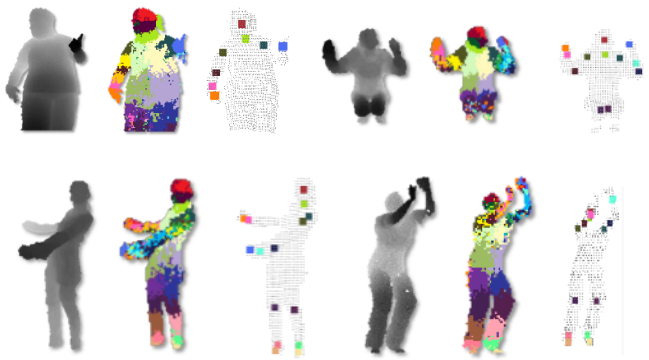
- Capture and sample 500K motion capture frames of people kicking, driving, dancing, etc.
- Get 3D models for 15 bodies with a variety of weights, heights, etc.
- Synthesize motion capture data for all 15 body types

Results

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)

Results



Real-Time Human Pose Recognition in Parts from Single Depth Images

CVPR 2011

Jamie Shotton, Andrew Fitzgibbon, Mat Cook, Toby Sharp, Mark Finocchio, Richard Moore, Alex Kipman, Andrew Blake
Microsoft Research Cambridge & Xbox Incubation

Cameras for Tracking

Leap Motion
- 2' x 2' x 2' volume
- 2015, $80