Capture, Analysis, and Applications of 3D Visual Signals

Zhengyou Zhang
Research Manager/Principal Researcher
Microsoft Research

zhang@microsoft.com
http://research.microsoft.com/~zhang/
Microsoft Kinect Sensor

- RGB camera
- infra-red camera
- infra-red projector
- Microphones
- Motor
- USB
Crazy Head Tracking Androids

Sittiphol Phanvilai @ Hua Lampong Co., Ltd has implemented head tracking using Kinect and creates crazy 3D effects with android dolls. This is a modification of their earlier Kinect VR project which had spheres instead of androids.
Every few hours new applications are emerging for the Kinect and creating a new phenomenon that is nothing short of revolutionary.

- Quote from KinectHacks.net
3D Video Capture

http://www.youtube.com/watch?v=7QrnwoO1-8A
Music Video

http://www.youtube.com/watch?v=VMWc2KPFFv4
Navigational Aids for the Visually Impaired

http://www.youtube.com/watch?v=l6QY-eb6NoQ
Kinect for Windows SDK

www.microsoft.com/en-us/kinectforwindows

- Access to deep Kinect system information
  - Depth data, near mode
  - Synchronized depth and RGB streams
  - Audio
  - Direct control of the Kinect sensor
  - System API
  - Skeletal tracking, sitting or standing up
  - Voice command
Sample App: Shape Game
HOW IT WORKS?

Human stereo vision
Computer stereo vision
Kinect sensing technology
Human Stereo Vision

Difference in your two eyes gives you the ability to perceive your surrounding environment in 3 Dimensions

http://www.vision3d.com/stereo.html
Key to Perceive in 3D

• See two different views
• Match similarity between the two views
• Fuse them to reconstruct the scene in 3D

To see 3D,
• Your two eyes must work *simultaneously*
• Your *brain* is able to fuse the two views

• At least 12% of people have some problem with their stereo vision

http://www.vision3d.com/whycant.html
How it works? **Kinect Sensor**

- Modified structured light 3D scanner
  - IR projector
  - IR camera
  - Random pattern
Matching & Depth Map

• Correlation
Overlay of Depth Map on IR Image
Kinect Calibration

- The Kinect calibration card is used to recalibrate your sensor in the event the sensor is not properly tracking your body. The card is included in the Kinect Adventures games.
RGB vs. Depth Sensors

**RGB**
- ✗ Only works well lit
- ✗ Background clutter
- ✗ Scale unknown

**Depth**
- ✓ Works in low light
- ✓ Person ‘pops’ out from bg
- ✓ Scale known
- ✗ Shadows, missing pixels

much easier with depth!
Challenges

• Noisy data
  – How to characterize the uncertainty?
  – How to deal with the sensor inefficiency (e.g., non-IR-reflective surface, environment with strong ambient IR)?

• Partial data
  – How to fuse multiple views?
  – How to deal with interference between multiple sensors?
  – How to leverage visual sensors?

• Raw data
  – How to infer high-level/semantic information?

• Multimodal data
  – How to collaborate with audio, tactile, inertial sensors to create compelling applications?
HUMAN BODY-LANGUAGE UNDERSTANDING
Human Body Language

• A form of non-verbal communication
  – Body posture
  – Gestures
  – Facial expressions
  – Eye movements (eye gaze)

• Humans send and interpret such signals almost entirely subconsciously
Body Language

**Communicator**
- Send out
- Receive

**Mode**
- Facial expression
- Body movement
- Tone of voice

**Control**
- Voluntary
- Involuntary

**Concordance**
- In concordance
- Discordance (lie)
Outline

• Skeletal tracking
• Human action recognition
• Hand gesture recognition
• Head pose and facial expression tracking
SKELETAL TRACKING

Jamie Shotton, Andrew Blake, Kinect Team
Human pose estimation

Kinect tracks 20 body joints in real time.
Depth cameras

• Technology
  – structured IR light

☑ cheap, fast, accurate
☒ missing pixels, shadows
Depth cameras

depth image (camera view)

top view

side view
The Kinect pose estimation pipeline

1. capture depth image
2. infer body parts
3. hypothesize body joints
4. track skeleton (3D side view)
Body part recognition

(left hand)

(neck)

(right shoulder)

(left elbow)

(“left” = player left with camera acting as mirror)
Classifying pixels

• Compute $P(c_i | w_i)$
  – pixels $i = (x, y)$
  – body part $c_i$
  – image window $w_i$

• Discriminative approach

• Learn classifier $P(c_i | w_i)$ from training data
Fast depth image features

- Depth comparisons:
  \[ f(i; \Delta) = d(i) - d(i') \]
  where \( i' = i + \Delta \)

- Background pixels
  \[ d = \text{large constant} \]
Decision tree classification

Toy example:
distinguish
left (L) and right (R)
sides of the body

\[ f(i; \Delta_1) > \theta_1 \]
\[ f(i; \Delta_2) > \theta_2 \]
Training decision trees [Breiman et al. 84]

Take \((\Delta, \theta)\) that maximises information gain:

\[
\Delta E = - \frac{|I_1|}{|I_n|} E(I_1) - \frac{|I_r|}{|I_n|} E(I_r)
\]

**Goal:** drive entropy at leaf nodes to zero
Depth of trees

input depth

Correct parts (ground truth)

inferred parts (soft)

depth 18
Depth of trees

Average per-class accuracy vs Tree depth for 5k and 100k training images.
Decision forests

• Single trees tend to over-fit
• Train forest – ensemble of trees:

- different random subset of images
- average tree posteriors

$$P(c|w) = \sum_{t=1}^{T} P_t(c|w)$$
Number of trees

Average per-class accuracy

Number of trees

input

ground truth

inferred body parts (most likely)

1 tree

2 trees

3 trees

40%

45%

50%

55%

60%
Body parts to joint hypotheses

• Depth image & probability mass

• Localize body parts in 3D
  – global centroid of prob. mass
  – local modes of density (mean shift)

• Map body parts to skeletal joints
  – many parts map directly to joints
3D joint hypotheses

NB No tracking or smoothing!

input depth image

inferred body parts & overlaid joint hypotheses

3D joint hypotheses

front view

top view

side view
... and then magic happens

• Exploit
  – 3D joint hypotheses
  – kinematic constraints
  – temporal coherence

• Predict
  – full skeleton
  – invisible joints
  – multi-player
HUMAN ACTION RECOGNITION
The Problem

• Recognize actions from sequences of depth maps

• Issues to address
  – large amount of data
  – Coarse and noisy depth measurement

Tennis Swing
Method - Action Graphs

- **Node:** Salient posture
- **Path:** Action

**Training Data Flow**

- **Extraction of Bags of 3D Points from Depth Maps**
- **Classification (Symbolization)**
  
  $X = \{x_1, x_2, \ldots, x_T\}$
  
  $S = \{s_1, s_2, \ldots, s_T\}$

**Posture Modeling**

**Action Graph**

$G = (\Omega, A_1, A_2, \ldots, A_M)$

**Action Recognition**

**Action Training**

**Recognition Data Flow**

Pair-wise clustering & GMM

Sequences of depth maps
Posture Modeling

• 3D representative points are sampled from each depth map → A Bag of Points (BoPs)
  – Projection based

• Distribution of the 3D points for each posture
  – GMM

• Distances between two depth maps
  – Hausdorff distance between the two BoPs
Experimental Results

• Data Collection
  – Depth camera using structured infrared light
  – Depth map resolution 640x480 pixels
  – 20 Actions
    • Movement of arms, legs, torso and coordination of them
  – 7 Subjects
    • Each subject performed each action 3 times
20 Actions

• 20 actions
  – 10 with one hand, 2 with two hands, 2 with one leg
  – 6 with whole body

<table>
<thead>
<tr>
<th>High-arm wave</th>
<th>Two hand wave</th>
</tr>
</thead>
<tbody>
<tr>
<td>Horizontal-arm wave</td>
<td>Side-boxing</td>
</tr>
<tr>
<td>Hammer</td>
<td>Bend</td>
</tr>
<tr>
<td>Hand catch</td>
<td>Forward-kick</td>
</tr>
<tr>
<td>Forward punch</td>
<td>Side-kick</td>
</tr>
<tr>
<td>High throw</td>
<td>Jogging</td>
</tr>
<tr>
<td>Draw x</td>
<td>Tennis swing</td>
</tr>
<tr>
<td>Draw tick</td>
<td>Tennis swing</td>
</tr>
<tr>
<td>Draw Circle (Clockwise)</td>
<td>Golf-swing</td>
</tr>
<tr>
<td>Hand clap</td>
<td>Pickup &amp; throw</td>
</tr>
</tbody>
</table>
Three Test Actions Sets

- Due to consideration of the computational cost, the 20 actions are divided into three subsets:

<table>
<thead>
<tr>
<th>Action Set One (AS1)</th>
<th>Action Set Two (AS2)</th>
<th>Action Set Three (AS3)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Horizontal-arm wave</td>
<td>High-arm wave</td>
<td>High throw</td>
</tr>
<tr>
<td>Hammer</td>
<td>Hand catch</td>
<td>Forward kick</td>
</tr>
<tr>
<td>Forward punch</td>
<td>Draw x</td>
<td>Side kick</td>
</tr>
<tr>
<td>High throw</td>
<td>Draw tick</td>
<td>Jogging</td>
</tr>
<tr>
<td>Hand clap</td>
<td>Draw circle</td>
<td>Tennis Serving</td>
</tr>
<tr>
<td>Bend</td>
<td>Two hand wave</td>
<td>Tennis swing</td>
</tr>
<tr>
<td>Tennis serve</td>
<td>Forward kick</td>
<td>Golf swing</td>
</tr>
<tr>
<td>Pickup &amp; throw</td>
<td>Side-boxing</td>
<td>Pickup &amp; throw</td>
</tr>
</tbody>
</table>
## Recognition Accuracy using 3D BoP

<table>
<thead>
<tr>
<th>Action Set</th>
<th>1/3 samples as training</th>
<th>2/3 samples as training</th>
<th>½ subjects’ samples as training</th>
</tr>
</thead>
<tbody>
<tr>
<td>AS1</td>
<td>89.5%</td>
<td>93.4%</td>
<td>72.9%</td>
</tr>
<tr>
<td>AS2</td>
<td>89.0%</td>
<td>92.9%</td>
<td>71.9%</td>
</tr>
<tr>
<td>AS3</td>
<td>96.3%</td>
<td>96.3%</td>
<td>79.2%</td>
</tr>
<tr>
<td>overall</td>
<td>91.6%</td>
<td>94.2%</td>
<td>74.7%</td>
</tr>
</tbody>
</table>
Comparison to 2D Silhouettes

• 2D silhouettes were obtained from the xy-projections
  – which is close to silhouettes from a 2D image
• 80 2D points were sampled from the contour of each 2D silhouette.
• Using
  – the same number of postures
  – the same number of Gaussian components and
  – the same number of training samples
Recognition Accuracy using 2D Silhouettes

<table>
<thead>
<tr>
<th>Action Set</th>
<th>1/3 samples as training</th>
<th>2/3 samples as training</th>
<th>½ subjects’ samples as training</th>
</tr>
</thead>
<tbody>
<tr>
<td>AS1</td>
<td>79.5%</td>
<td>81.3%</td>
<td>36.3%</td>
</tr>
<tr>
<td>AS2</td>
<td>82.2%</td>
<td>88.7%</td>
<td>48.9%</td>
</tr>
<tr>
<td>AS3</td>
<td>83.3%</td>
<td>89.5%</td>
<td>45.8%</td>
</tr>
<tr>
<td>overall</td>
<td>81.7%</td>
<td>86.5%</td>
<td>43.7%</td>
</tr>
</tbody>
</table>

vs. 3D Bag of Points

| overall    | 91.6%                   | 94.2%                   | 74.7%                           |

Recognition with 3D is much more accurate!
Zhou Ren, Junsong Yuan, Zhengyou Zhang

HAND GESTURE RECOGNITION
Challenges

Figure 1: Some challenging cases for hand gesture recognition with depth cameras: the first and the second hands have the same gesture while the third hand confuses the recognition.

The resolution of depth map is low
System of Kinect-based gesture recognition

Key Modules: Hand segmentation and representation, Dissimilarity Measure (Finger Detection and FEMD)
Hand Segmentation & Representation

Figure 3: Hand segmentation process. (a). The RGB color image captured by Kinect Sensor; (b). The depth map captured by Kinect Sensor; (c). The area segmented using depth information; (d). The hand shape segmented using RGB information.

Figure 4: Hand shape representation. (a). On the contour of the segmented hand, the green line is the detection of the black belt; the red point is the initial point; the cyan point is the center point detected by Distance Transform; (b). The time-series curve of the shape above.
Finger Detection via shape decomposition

Figure 6: Illustration of the two proposed finger detection methods: (a). Thresholding decomposition uses a height threshold $h_f$ in the time-series curve to detect fingers, which means to decompose the shape with a circle, thus information is inevitably lost; (b). Near-convex decomposition decomposes the hand into several near-convex parts that are fingers and the palm. The finger decomposition of (b) is more accurate and robust.

$$\min \quad \alpha \| x \|_0 + (1 - \alpha) w^T x,$$

$$s.t. \quad A x \geq 1, \quad x^T B x = 0, \quad x \in \{0, 1\}^n$$
Distance Metric: Finger-Earth Mover’s Distance

FEMD vs. EMD: 1. consider global feature (finger); 2. alleviate partial matching

Figure 5: The motivation of using Finger-Earth Mover’s Distance. (a) and (b) are two different hand shapes, whose time-series curves are shown in (e) and (f), respectively. Their major difference is the fingers. (c) and (d) are two signatures that partially match, their EMD cost is 0, however they are very different. Hence FEMD adds the penalty on empty holes. (e) and (f) are the time-series curves of the hand shapes in (a) and (b), each curve is represented as a signature with each finger as a cluster; the signature with bigger total weight serves as holes, the smaller one serves as earth piles.
Results

- New collected dataset with Kinect camera:

10 subjects * 10 gestures/subject * 10 cases/gesture = 1000 cases
Contain color image and depth map
Under uncontrolled environment
Accuracy and efficiency

<table>
<thead>
<tr>
<th></th>
<th>Thresholding Decomposition+FEMD</th>
<th>Near-convex Decomposition+FEMD</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mean Accuracy</td>
<td>90.6%</td>
<td>93.9%</td>
</tr>
<tr>
<td>Mean Running Time</td>
<td>0.5004s</td>
<td>4.0012s</td>
</tr>
</tbody>
</table>

Table 1: The mean accuracy and the mean running time of the two proposed methods.

Figure 15: The confusion matrix of Experiment II.
Robust Hand Gesture Recognition with Kinect Sensor

Zhou Ren, Jingjing Meng, Junsong Yuan, Zhengyou Zhang
School of EEE, NTU, Singapore & Microsoft Research Redmond, USA

Proc. of ACM Intl. Conf. on Multimedia 2011
Qin Cai, Cha Zhang, Zhengyou Zhang

HEAD POSE & FACIAL EXPRESSION TRACKING
Geminoid Summit
Deformable Face Tracking

• Many applications
  – Human computer interaction
  – Performance-driven facial animation
  – Face recognition

• Challenging
  – Limited number of features on the face
  – Dozens of parameters to estimate
Linear Deformable Model

\[
\begin{bmatrix}
q_1 \\
\vdots \\
q_K
\end{bmatrix} = \begin{bmatrix}
p_1 \\
\vdots \\
p_K
\end{bmatrix} + A \begin{bmatrix}
r_1 \\
\vdots \\
r_K
\end{bmatrix} + B \begin{bmatrix}
s_1 \\
\vdots \\
s_K
\end{bmatrix}, \text{ where } A = \begin{bmatrix}
A_1 \\
\vdots \\
A_K
\end{bmatrix}, \ B = \begin{bmatrix}
B_1 \\
\vdots \\
B_K
\end{bmatrix}
\]

(Artist rendered linear deformable model)
Maximum Likelihood DMF

• Formulation, \((q_k, g_k)\) correspondence pair:
  
  \[
  R(p_k + A_k r + B_k s) + t = g_k + x_k
  \]
  
  \[
  x_k \sim N(0, \Sigma_{x_k})
  \]

• Iterative closest point (ICP)
  
  – Assume closest points correspond
  – Compute transformation
  – Iterate until convergence
Model Initialization

Input → Face Detection → Face Alignment → Model Initialization

- Green dots: point-to-point distance
- Blue dots: point-to-plane 3D distance
- Red dots: point-to-plane 2D distance
- White dots: unused

Deformable model projected onto the texture image

Alignment points

$v_{lk}$
Face Tracking

• Tracking
  – Shape deformations fixed
  – Based on feature point correspondence
  – Solve for action deformation, rotation and translation

• Regularization
  • $l_2$ norm constraining the difference between neighboring frames’ action deformations
  • $l_1$ norm constraining the number of non-zero action deformation parameters
Tracking Results: Video

Top to bottom: Seq #1 (810 frames), Seq #2 (681 frames), Seq #3 (300 frames)
## Qualitative Results

### Median tracking error in pixels

<table>
<thead>
<tr>
<th></th>
<th>ID+$l_2$</th>
<th>ID+$l_1$</th>
<th>ID+$l_2+l_1$</th>
<th>NM+$l_2$</th>
<th>NM+$l_1$</th>
<th>NM+$l_2+l_1$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Seq #1</td>
<td>3.56</td>
<td>2.88</td>
<td>2.78</td>
<td>2.85</td>
<td>2.69</td>
<td>2.66</td>
</tr>
<tr>
<td>Seq #2</td>
<td>4.48</td>
<td>3.78</td>
<td>3.71</td>
<td>4.30</td>
<td>3.64</td>
<td>3.55</td>
</tr>
<tr>
<td>Seq #3</td>
<td>3.98&lt;sup&gt;L&lt;/sup&gt;</td>
<td>3.91</td>
<td>3.91</td>
<td>3.92&lt;sup&gt;L&lt;/sup&gt;</td>
<td>3.91</td>
<td>3.50</td>
</tr>
</tbody>
</table>

**ID**: use identity covariance matrix for sensor noise

**NM**: use the proposed noise modeling scheme

$l_2$: quadratic constraint between successive frames

$l_1$: sparse constraint on the action transforms

<sup>L</sup>: lost tracking in the middle and never recover
Avatar Kinect
Avatar Kinect
CHALLENGES
Challenges (1)

• Model human body language
  – Facial expression
  – Head gesture
  – Hand gesture
  – Body gesture
  – Motion dynamics
  – Behaviors
  – Human-human interaction
  – ...

Challenges (2)

- Improve sensor quality
  - Short range vs. Long range
  - Day vs. Night
  - Indoor vs. Outdoor
  - Different surface materials
- Model sensor imprecision
- Fuse multiple sensors
Challenges (3)

• Develop efficient and robust algorithms
  – Deal with various challenging situations
  – Process a large amount of data
  – Handle inter-/intra- person variations
  – Collect and label large-scale training/test datasets
  – ...

• Understand societal implications
  – E.g. Privacy
References

• Z. Ren, J. Yuan, and Z. Zhang, ``Robust Hand Gesture Recognition Based on Finger-Earth Mover’s Distance with a Commodity Depth Camera'', in Proc. ACM International Conference on Multimedia (ACM MM), Scottsdale, Arizona, USA, Nov. 28--Dec. 1, 2011. To Appear.
Acknowledgment

• Qin Cai
• Phil Chou
• Anoop Gupta
• Zicheng Liu
• Cha Zhang
• Niru Chandrasekaran

• Jamie Shotton
• Wanqing Li
• Zhou Ren
• Junsong Yuan