3D Skeletal Tracking on Azure Kinect

--Azure Kinect Body Tracking SDK

Zicheng Liu, Principal Research Manager
Microsoft
Build computer vision and speech models using a developer kit with advanced AI sensors

- Get started with a range of SDKs, including an open-source Sensor SDK.
- Experiment with multiple modes and mounting options.
- Add cognitive services and manage connected PCs with easy Azure integration.
(1) 1MP depth sensor
(2) 7-mic array
(3) 12 MP RGB video camera
(4) Accelerometer and gyroscope (IMU)
(5) External sync pins
(6) 120 degree FOV mode
Use Cases

Analyzed over 900 IWANTKINECT survey responses for body tracking applications. Thank you!

Three clear winners
- Kinematic analysis
- Human understanding
- Human interaction

Large focus on these use cases in training and validating model
Kinematic Analysis

Posture analysis
Rehabilitation
Fitness
Patient monitoring
Fall detection
Sports instruction
Hack for Good
Gigi’s Playhouse
AI-Based Physical Therapy for Down Syndrome
Human Understanding

Shopper behavior understanding
Person detection and counting
Person tracking
Smart spaces interaction
Human Interaction

Information signs and video walls
Interactive art and performance
Interactive (museum) exhibits
Customer sizing and fitting
Machine safety
Overview of Body Tracking SDK

- Designed from the ground up for Azure Kinect DK
  - Instance segmentation map
  - 3D joint positions per person
  - Unique IDs to track temporally

**Improved performance over Kinect for Windows v2**
- Anatomically (28+ landmarks / joints) more accurate skeleton
- Higher joint accuracy and precision
- Improved robustness e.g. side view, bending, lying

**Cross platform development**
- Windows with Linux in preview
- C/C++ and C# (coming later)

**ONNX runtime with support for NVIDIA 1070 (or better) hardware acceleration**
System architecture

Azure Kinect DK → Sensor SDK → AB IR → ONNX Runtime → 2D Joints → Model Fitting → 3D Joints

Depth → ONNX Runtime → CNN → 2D Joints

Segmentation

2D Joints
3DSkeletons
Why 3D

• Calculating joint angles is not possible to do correctly in 2D

• Understanding of whether a joint is coincident with another 3D object

• Accurate scale estimation for user size/height
CNN: 2D pose estimation from IR
Human Pose Estimation

- **Top-Down**
  - ✓ Person detector + Single-person pose estimation
  - ✓ Person detection errors

- **Bottom-Up**
  - ✓ Directly inferring the poses of multiple people in an image
  - ✓ Unknown number of people that can occur at any position or scale

- **2D => 3D**
  - ✓ Ongoing research
  - ✓ Single-person based 2D-to-3D conversion
  - ✓ Depth/scale is not deterministic

* https://github.com/facebookresearch/VideoPose3D
Challenge

- Accuracy vs. Speed
  ✓ Trade-off for low-end GPUs

- RGB vs. AB/Depth
  ✓ No available dataset like MSCOCO for AB/Depth

- Real vs. Synthetic
  ✓ The reality gap

- Additional output
  ✓ Instance segmentation
  ✓ Pose estimation for hands and feet

- [https://github.com/CMU-Perceptual-Computing-Lab/openpose](https://github.com/CMU-Perceptual-Computing-Lab/openpose)
- [http://cocodataset.org/#keypoints-leaderboard](http://cocodataset.org/#keypoints-leaderboard)
CNN Architecture

Data Preprocessing

Normalized image
RGB: 3-channel [0, 1]
AB: 1-channel [0,1]
Depth: 1-channel [0,1]

RGB image
or AB map
or Depth map

RGB: 3-channel [0, 1]
AB: 1-channel [0,1]
Depth: 1-channel [0,1]

CNN Network

Backbone

Keypoints

PAFs

Body Part Segmentation

Peak Finding w/ NMS

18 heatmaps, one for each keypoint

A set of peaks, w/ their locations, labels, and probabilities

A set of grouped skeletons w/ instance and parts segmentation

17x2 heatmaps, 2 for each PAF

Skeleton Linking + Instance Segmentation

15 heatmaps, 1 for each part segmentation

15 heatmaps, 1 for each part segmentation
CNN Architecture

Backbone Network (Concatenated Pyramid Network)

ResNet50

\[ C_5 \rightarrow C_4 \rightarrow C_3 \rightarrow C_2 \rightarrow C_1 \]

\[ \text{Concat} \rightarrow \text{Conv} 3 \times 3 \rightarrow \text{Conv} 3 \times 3 \rightarrow F \]

Network Heads

\[ \text{Conv} 3 \times 3 \rightarrow \text{Conv} 3 \times 3 \rightarrow \text{Conv} 3 \times 3 \rightarrow \text{Conv} 3 \times 3 \rightarrow \text{Conv} 3 \times 3 \rightarrow \text{Conv} 1 \times 1 \rightarrow \text{Conv} 1 \times 1 \rightarrow S \]

\[ \text{Conv} 3 \times 3 \rightarrow \text{Conv} 3 \times 3 \rightarrow \text{Conv} 3 \times 3 \rightarrow \text{Conv} 3 \times 3 \rightarrow \text{Conv} 1 \times 1 \rightarrow \text{Conv} 1 \times 1 \rightarrow L \]

\[ \text{Conv} 3 \times 3 \rightarrow \text{Conv} 3 \times 3 \rightarrow \text{Conv} 3 \times 3 \rightarrow \text{Conv} 3 \times 3 \rightarrow \text{Conv} 1 \times 1 \rightarrow \text{Conv} 1 \times 1 \rightarrow X \]
CNN Training

\[ L = L_{\text{pose}}(D^{\text{pose}}_r) + L_{\text{pose}}(D^{\text{pose}}_s) + L_{\text{part}}(D^{\text{part}}_s) \]
Synthetics Data Strategy

*Synthetic Data used for training*
Synthetic Data
Reality Gap between Real and Synthetics
Results on Real AB Input
Results on Real RGB Input
Live Skeleton Tracking on iPhone

Real-time Skeletal Tracking on iPhone Demo
3D Model Fitting Using Depth Map
Model Fitting

**Input**
- AB frame
  - Linked 2D DNN keypoints
- Depth frame

**Output**
- 3D Joint Locations
- Joint Orientation
- Temporal Identity

Side views:
Model Fitting - Challenges

• **Easy Case**
  • Frontal view, un-occluded

• **Challenging Cases**
  • Unreliable depth
    • Dark clothes (IR absorbing)
    • FOV cut-off
  • Partial view of the person
    • Self-occlusions (e.g. side view)
    • People occluding other people
Model Fitting – Skeleton Based Tracking

**Kinematic Model**
- Joint angles
- Scaling factor
- Global rigid transform

**Input**
- Depth image
- Linked DNN keypoints in 2D (from AB image)

**Energy Data Terms**
- 2D keypoint reprojection
- 3D surface depth displacement

**Energy Regularization Terms**
- Anatomical joint limits
- Pose prior regularization
- Scale prior regularization
- Temporal coherency
Model Fitting – Results
Demo
Model Fitting – Results
## Runtime Speed

<table>
<thead>
<tr>
<th>Hardware</th>
<th>CPU</th>
<th>GPU</th>
<th>Depth Speed (ms)</th>
<th>DNN Speed (ms)</th>
<th>Model Fitting Speed[1 person] (ms)</th>
<th>SDK Framerate (FPS)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Z440</td>
<td>Xeon(R) CPU E5-1660 v4 @ 3.20GHz 3.20 GHz</td>
<td>GTX 1080Ti</td>
<td>3.0</td>
<td>19.2</td>
<td>2.9</td>
<td>50</td>
</tr>
<tr>
<td>Z420</td>
<td>Xeon(R) CPU E5-1620 0 @ 3.60GHz 3.60 GHz</td>
<td>GTX 1070</td>
<td>4.0</td>
<td>30.2</td>
<td>3.3</td>
<td>30</td>
</tr>
<tr>
<td>Surface Book</td>
<td>I7-8650U CPU @ 1.90GHz 2.11 GHz</td>
<td>GTX 1060M</td>
<td>6.2</td>
<td>47.1</td>
<td>3.6</td>
<td>17</td>
</tr>
</tbody>
</table>
Summary

- Azure Kinect Body Tracking SDK
  - DNN based algorithm
  - Using synthetic data
  - Handling challenging poses and camera angles
- Beta release in Windows and Linux:
  https://docs.microsoft.com/en-us/azure/kinect-dk/sensor-sdk-download
THANKS!

Acknowledgement to the dev team of the AKBT SDK

Contact: Zicheng Liu
zliu@microsoft.com