Depth, Human Pose, and Camera Pose

Jamie Shotton
Kinect Adventures

- Depth sensing camera
- Tracks 20 body joints in real time
- Recognises your face and voice
Trial of "touchless" gaming technology in surgery

By Adam Brotelow
Health Correspondent, BBC News

Doctors in London are using "touchless" technology, often used in TV games, to help them carry out delicate neurosurgery.

They allow surgeons to manipulate images with their voice and hand gestures rather than using a keyboard and mouse.

Surgeons say it gives them more control and avoids disruption.

The technology could be a valuable aid to surgery in the future.
What the Kinect Sees

depth image
(camera view)
Structured light

object at depth $d_1$

object at depth $d_2$

optic centre of IR laser

optic centre of camera

imaging plane

baseline

$x$

$y$

$z$
Depth Makes Vision That Little Bit Easier

**RGB**
- ✗ Only works well lit
- ✗ Background clutter
- ✗ Scale unknown
- ✗ Color and texture variation

**Depth**
- ✓ Works in low light
- ✓ Background removal easier
- ✓ Calibrated depth readings
- ✓ Uniform texture
Joint work with Shahram Izadi, Richard Newcombe, David Kim, Otmar Hilliges, David Molyneaux, Pushmeet Kohli, Steve Hodges, Andrew Davison, Andrew Fitzgibbon.

SIGGRAPH, UIST and ISMAR 2011.
ROADMAP

**The Vitruvian Manifold**

[CVPR 2012]

**Scene Coordinate Regression**

[CVPR 2013]
THE VITRUVIAN MANIFOLD

Jonathan Taylor  Jamie Shotton  Toby Sharp  Andrew Fitzgibbon

CVPR 2012
Human Pose Estimation

Given some image input, recover the 3D human pose:

In this work:

• Single frame at a time (no tracking)
• Kinect depth image as input (background removed)
Why is Pose Estimation Hard?
A Few Approaches

Regress directly to pose?
  e.g. [Gavrila ’00] [Agarwal & Triggs ’04]

Detect and assemble parts?
  e.g. [Felzenszwalb & Huttenlocher ’00] [Ramanan & Forsyth ’03] [Sigal et al. ’04]

Detect parts?
  e.g. [Bourdev & Malik ’09] [Plagemann et al. ‘10] [Kalogerakis et al. ‘10]

Per-Pixel Body Part Classification
  [Shotton et al. ‘11]

Per-Pixel Joint Offset Regression
  [Girshick et al. ‘11]
Background: Learning Body Parts for Kinect

input depth image → body parts → body joint hypotheses

[Shotton et al. CVPR 2011]
Synthetic Training Data

Record mocap
100,000s of poses

Retarget to varied body shapes

Render (depth, body parts) pairs

Train invariance to:
Depth Image Features

• Depth comparisons
  – very fast to compute

\[ f(x; \mathbf{v}) = d(x) - d(x + \Delta) \]

\[ \Delta = \frac{\mathbf{v}}{d(x)} \]

Background pixels
\( d = \text{large constant} \)
Decision tree classification

image window centred at $x$

$f(x; v_1) > \theta_1$

$f(x; v_2) > \theta_2$

$f(x; v_3) > \theta_3$
Training Decision Trees

\[ S_n = \{ x \} \text{ for all pixels} \]

\[ f(x; \mathbf{v}_n) > \theta_n \]

Goal: drive entropy at leaf nodes to zero

Take \((\mathbf{v}, \theta)\) that maximises information gain:

\[ \Delta E = -\frac{|S_l|}{|S_n|} E(S_l) - \frac{|S_r|}{|S_n|} E(S_r) \]
Decision Forests Book

• Theory – Tutorial & Reference
• Practice – Invited Chapters
• Software and Exercises
• Tricks of the Trade
input depth

inferred body parts

no tracking or smoothing
input depth

inferred body parts

front view

side view

inferred joint position hypotheses

top view

no tracking or smoothing
Body Part Recognition in Kinect

Single frame at a time $\rightarrow$ robust

Large training corpus $\rightarrow$ invariant

Fast, parallel implementation

Skeleton does not explain the depth data
Limited ability to cope with hard poses
A few approaches

Explain the data directly with a mesh model
[Ballan et al. ‘08] [Baak et al. ‘11]

- **GOOD:** Full skeleton
- **GOOD:** Kinematic constraints enforced from the outset
- **GOOD:** Able to cope with occlusion and cropping
- **BAD:** Many local minima
- **BAD:** Highly sensitive to initial guess
- **BAD:** Potentially slow
Human Skeleton Model

- Mesh is attached to a hierarchical skeleton
- Each limb $l$ has a transformation matrix $T_l(\theta)$ relating its local coordinate system to the world:

\[
T_{\text{root}}(\theta) = R_{\text{global}}(\theta) \\
T_l(\theta) = T_{\text{parent}(l)}(\theta)R_l(\theta)
\]

- $R_{\text{global}}(\theta)$ encodes a global scaling, translation and rotation
- $R_l(\theta)$ encodes a rotation and fixed translation relative to its parent
- 13 parameterized joints using quaternions to represent unconstrained rotations
- This gives $\theta$ a total of $1 + 3 + 4 + 4 \times 13 = 60$ degrees of freedom
Each vertex \( u \)
- has position \( p \) in base pose \( \theta_0 \)
- is attached to \( K \) limbs \( \{l_k\}_{k=1}^K \) with weights \( \{\alpha_k\}_{k=1}^K \)

In a new pose \( \theta \), the skinned position \( u \) of is:

\[
M(u; \theta) = \sum_{k=1}^{K} \alpha_k T_{l_k}(\theta)T_{l_k}^{-1}(\theta_0)p
\]

Mesh in base pose \( \theta_0 \)
Test Time Model Fitting

- Assume each observation $x_i$ is generated by a point on our model $u_i$

**Observed Points**: $x_1, ..., x_n$

**Corresponding Model Points**: $u_1, ..., u_n$

**Optimize**:

$$\min_{\theta} \min_{u_1, ..., u_n} \sum_i d(x_i, M(u_i; \theta))$$

Note: simplified energy - more details to come
Optimizing $\min_{\theta} \min_{u_1...u_n} \sum_i d(x_i, M(u_i; \theta))$

- Alternating between pose $\theta$ and correspondences $u_1, ... u_n$
  - Articulated Iterative Closest Point (ICP)
- Traditionally, start from initial $\theta$
  - manual initialization
  - track from previous frame

- Could we instead infer initial correspondences $u_1, ... u_n$ discriminatively?
- And, do we even need to iterate?
Can we achieve a good result without iterating between pose $\theta$ and correspondences $u_1, \ldots, u_n$?
From Body Parts to Dense Correspondences

Texture is mapped across body shapes and poses

Body Parts

increasing number of parts
classification regression

The “Vitruvian Manifold”
The “Vitruvian Manifold” Embedding in 3D

Geodesic surface distances approximated by Euclidean distance
Overview

Inferred dense correspondences

Regression forest

Energy function

Optimization of model parameters $\theta$

Final optimized poses

Training images

Test images
Discriminative Model: Predicting Correspondences

training images

input images

regression forest

inferred dense correspondences
Learning the Correspondences

• How to learn the mapping from depth pixels to correspondences?

• Render synthetic training set:

  - characters
  - mocap

• Train regression forest
Each pixel-correspondence pair descends to a leaf in the tree.

Learning a Regression Model at the Leaf Nodes
Inferring Correspondences
infer correspondences $U$

set up minimization

optimize parameters

$$\min_{\theta} E(\theta, U)$$
Full Energy

\[ E(\theta, U) = \lambda_{vis}E_{vis}(\theta, U) + \lambda_{prior}E_{prior}(\theta) + \lambda_{int}E_{int}(\theta) \]

- Term \( E_{vis} \) approximates hidden surface removal and uses robust error
- Gaussian prior term \( E_{prior} \)
- Self-intersection prior term \( E_{int} \) approximates interior volume

Energy is robust to noisy correspondences

- Correspondences far from their image points are “ignored”
- Correspondences facing away from the camera are “ignored”
  - avoids model getting stuck in front of the image pixels
“Easy” Metric: Average Joint Accuracy

Joints average accuracy (% joints within distance $D$)

$D$: max allowed distance to GT (m)

- Our algorithm
- Given GT $U$
- Optimal $\theta$

Results on 5000 synthetic images
Hard Metric: “Perfect” Frame Accuracy

Results on 5000 synthetic images

<table>
<thead>
<tr>
<th>$D$</th>
<th>0.09m</th>
<th>0.11m</th>
<th>0.17m</th>
<th>0.21m</th>
<th>0.45m</th>
</tr>
</thead>
<tbody>
<tr>
<td>$U$</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Comparison

Worst case accuracy (% frames with all joints within dist. $D$)

- **Vitruvian Manifold**
- [Shotton et al. '11] (top hypothesis)
- [Girshick et al. 11] (top hypothesis)
- [Shotton et al. '11] (best of top 5)
- [Girshick et al. '11] (best of top 5)

$D$: max allowed distance to GT (m)

Achievable algorithms

Require an oracle

Results on 5000 synthetic images
Sequence Result

Depth Image

Predicted Correspondences

Each frame fit independently: no temporal information used
Note that the algorithm fits the character with strongest signal in each frame.
Generalization to Multiple 3D/2D Views

- Easily extended to Q views where each view has
  - $n_q$ correspondences per view
  - viewing matrix $P_q$ to register the scene

- Can also extend to 2D silhouette views
  - let data points $x_{ik}$ be 2D image coordinates
  - let $P_q$ include a projection to 2D
  - minimize re-projection error

$$\min_{\theta} \sum_{q=1}^{Q} \sum_{i}^{n_q} d(x_{iq}, P_q M(u_{iq}; \theta))$$
Silhouette Experiment

Worst case accuracy (% frames with all joints within dist. $D$)

$D$: max allowed distance to GT (m)

- 2 silhouette views
- 3 silhouette views
- 5 silhouette views
- 1 depth view
- 2 depth views
- 5 depth views
Vitruvian Manifold: Summary

• Predict per-pixel image-to-model correspondences
  – train invariance to body shape, size, and pose

• “One-shot” pose estimation
  – fast, accurate
  – auto-initializes using correspondences
SCENE COORDINATE REGRESSION FORESTS
FOR CAMERA RELocalIZATION IN RGB-D IMAGES

JAMIE SHOTTON  BEN GLOCKER  CHRISTOPHER ZACH  SHAHRAM IZADI  ANTONIO CRIMINISI  ANDREW FITZGIBBON

[CVPR 2013]
Know this

A world scene

Observe this

Input RGB

Input depth

Single RGB-D frame

Where is this?

6D camera pose, $H$
(camera to scene transformation)
APPLICATIONS

- Lost or kidnapped robots
- Improving KinectFusion
- Augmented reality
TYPICAL APPROACHES TO CAMERA LOCALIZATION

- **Tracking** – alignment relative to previous frame  
  e.g. [Besl & MacKay ‘92]

- **Key point detection → local descriptors → matching → geometric verification**  
  e.g. [Holzer et al. ‘12], [Winder & Brown ‘07], [Lepetit & Fua ‘06], [Irschara et al. ‘09]

- **Whole key-frame matching**  
  e.g. [Klein & Murray 2008] [Gee & Mayol-Cuevas 2012]

- **Epitomic location recognition**  
  [Ni et al. 2009]
PROBLEMS IN REAL WORLD CAMERA LOCALIZATION

- The real world is less exciting than vision researchers might like
  - sparse interest points can fail

- The real world is big
KEY IDEA: SCENE COORDINATE REGRESSION

Scene coordinate XYZ
↔
RGB color space
KEY IDEA: SCENE COORDINATE REGRESSION

- Let each pixel predict direct correspondence to 3D point in scene coordinates:

Input RGB | Input Depth | Desired Correspondences
---|---|---
A | | 
B | | 
C | | 

Scene coordinate XYZ $\leftrightarrow$ RGB color space

3D model from KinectFusion (only used for visualization)
SCENE COORDINATE REGRESSION

- Offline approach to relocalization
  - observe a scene
  - train a regression forest
  - revisit the scene

- Aim for really precise localization
  - e.g. suitable for AR overlays
  - from a single frame
  - without an explicit 3D model
SCENE COORDINATE REGRESSION (SCoRe) FORESTS

RGB Depth & RGB features

Depth & RGB features

SCoRe Forest

tree $l$

$\mathcal{M}_{l_1}(p)$

$\cdots$

$\mathcal{M}_{l_T}(p)$

tree $T$

$\delta_1 \frac{D(p)}{D(p)}$

$\delta_2 \frac{D(p)}{D(p)}$

$f^{\text{depth}}(p) = D \left( p + \frac{\delta_1}{D(p)} \right) - D \left( p + \frac{\delta_2}{D(p)} \right)$

$f^{\text{da-rgb}}(p) = I \left( p + \frac{\delta_1}{D(p)}, c_1 \right) - I \left( p + \frac{\delta_2}{D(p)}, c_2 \right)$

Leaf Predictions $\mathcal{M}_l \subset \mathbb{R}^3$

Forest Predictions $\mathcal{M}(p) = \bigcup_t \mathcal{M}_{l_t}(p)$
TRAINING A SCoRe FOREST

Training Data
- RGB-D frames with known camera poses $H$
- Generate 3D pixel labels automatically:
  \[
  \mathbf{m} = H\mathbf{x}
  \]

Learning (standard)
- Greedily train tree
- Reduction in spatial variance objective:
  \[
  Q(S_n, \theta) = V(S_n) - \sum_{d \in \{L,R\}} \frac{|S_n^d(\theta)|}{|S_n|} V(S_n^d(\theta))
  \]
  with \[
  V(S) = \frac{1}{|S|} \sum_{(p, \mathbf{m}) \in S} \|\mathbf{m} - \bar{\mathbf{m}}\|_2^2
  \]
- Regression, not classification
- Mean shift to summarize distribution at leaf $l$ into small set $\mathcal{M}_l \subset \mathbb{R}^3
ROBUST CAMERA POSE OPTIMIZATION

**Energy Function**

\[
E(H) = \sum_{i \in I} \rho \left( \min_{m \in M_i} \|m - Hx_i\|_2 \right)
\]

- **camera pose**
- **pixel index**
- **robust error function**
- **correspondences predicted by forest at pixel** \(i\)

**Optimization**

- Preemptive RANSAC
  [Nistér ICCV 2003]
- With pose refinement
  [Chum et al. DAGM 2003]
  - efficient updates to means & covariances used by Kabsch SVD
- Only a small subset of pixels used
INLYING FOREST PREDICTIONS

Test images

Inliers for six hypotheses

Camera pose

Ground truth

Inferred
PREEMPTIVE RANSAC OPTIMIZATION
## The 7Scenes Dataset

<table>
<thead>
<tr>
<th>Scene</th>
<th>Spatial Extent</th>
<th># Frames</th>
</tr>
</thead>
<tbody>
<tr>
<td>Chess</td>
<td>$3m^3$</td>
<td>4k</td>
</tr>
<tr>
<td>Fire</td>
<td>$4m^3$</td>
<td>2k</td>
</tr>
<tr>
<td>Heads</td>
<td>$2m^3$</td>
<td>1k</td>
</tr>
<tr>
<td>Office</td>
<td>$5.5m^3$</td>
<td>6k</td>
</tr>
<tr>
<td>Pumpkin</td>
<td>$6m^3$</td>
<td>4k</td>
</tr>
<tr>
<td>RedKitchen</td>
<td>$6m^3$</td>
<td>7k</td>
</tr>
<tr>
<td>Stairs</td>
<td>$5m^3$</td>
<td>2k</td>
</tr>
</tbody>
</table>

Dataset available from authors
<table>
<thead>
<tr>
<th>Sparse Key-Points (RGB only)</th>
</tr>
</thead>
<tbody>
<tr>
<td>- ORB matching</td>
</tr>
<tr>
<td>[Rublee et al. ICCV 2011]</td>
</tr>
<tr>
<td>- FAST detector</td>
</tr>
<tr>
<td>- Rotation aware BRIEF descriptor</td>
</tr>
<tr>
<td>- Hashing for matching</td>
</tr>
<tr>
<td>- Geometric verification</td>
</tr>
<tr>
<td>- RANSAC &amp; perspective 3 point</td>
</tr>
<tr>
<td>- Final refinement given inliers</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Tiny-Image Key-Frames (RGB &amp; Depth)</th>
</tr>
</thead>
<tbody>
<tr>
<td>- Downsampling to 40x30 pixels</td>
</tr>
<tr>
<td>- Blur</td>
</tr>
<tr>
<td>- Normalized Euclidean distance</td>
</tr>
<tr>
<td>- Brute-force search</td>
</tr>
<tr>
<td>- Interpolation of 100 closest poses</td>
</tr>
</tbody>
</table>

[Klein & Murray ECCV 2008]
[Gee & Mayol-Cuevas BMVC 2012]
### Quantitative Comparison

**Metric:**
Proportion of test frames with < 0.05m translational error and < 5° angular error

<table>
<thead>
<tr>
<th>Scene</th>
<th>Baselines</th>
<th>Our Results</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Tiny-image RGB-D</td>
<td>Sparse RGB</td>
</tr>
<tr>
<td>Chess</td>
<td>0.0%</td>
<td>70.7%</td>
</tr>
<tr>
<td>Fire</td>
<td>0.5%</td>
<td>49.9%</td>
</tr>
<tr>
<td>Heads</td>
<td>0.0%</td>
<td>67.6%</td>
</tr>
<tr>
<td>Office</td>
<td>0.0%</td>
<td>36.6%</td>
</tr>
<tr>
<td>Pumpkin</td>
<td>0.0%</td>
<td>21.3%</td>
</tr>
<tr>
<td>RedKitchen</td>
<td>0.0%</td>
<td>29.8%</td>
</tr>
<tr>
<td>Stairs</td>
<td>0.0%</td>
<td>9.2%</td>
</tr>
</tbody>
</table>

Choice of different image features
QUALITATIVE COMPARISON

ground truth  DA-RGB SCoRe forest  sparse baseline  closest training pose
QUALITATIVE COMPARISON

ground truth  DA-RGB SCoRe forest  sparse baseline  closest training pose
TRACK VISUALIZATION VIDEOS

ground truth

DA-RGB SCoRe forest

RGB sparse baseline

single frame at a time – no tracking
AR VISUALIZATION

RGB input + AR overlay

depth input + AR overlay

rendering of model from inferred pose

single frame at a time – no tracking
Add a single extra hypothesis to optimization: the result from previous frame

<table>
<thead>
<tr>
<th>Scene</th>
<th>Depth</th>
<th>Our Results</th>
<th>Frame-to-Frame Tracking</th>
</tr>
</thead>
<tbody>
<tr>
<td>Chess</td>
<td>82.7%</td>
<td>92.6%</td>
<td>95.5%</td>
</tr>
<tr>
<td>Fire</td>
<td>44.7%</td>
<td>82.9%</td>
<td>86.2%</td>
</tr>
<tr>
<td>Heads</td>
<td>27.0%</td>
<td>49.4%</td>
<td>50.7%</td>
</tr>
<tr>
<td>Office</td>
<td>65.5%</td>
<td>74.9%</td>
<td>86.8%</td>
</tr>
<tr>
<td>Pumpkin</td>
<td>58.6%</td>
<td>73.7%</td>
<td>76.1%</td>
</tr>
<tr>
<td>RedKitchen</td>
<td>61.3%</td>
<td>71.8%</td>
<td>82.4%</td>
</tr>
<tr>
<td>Stairs</td>
<td>12.2%</td>
<td>27.8%</td>
<td>39.2%</td>
</tr>
</tbody>
</table>
AR VISUALIZATION WITH TRACKING

RGB input + AR overlay

depth input + AR overlay

rendering of model from inferred pose

simple robust frame-to-frame tracking enabled

[Bunny: Stanford]
MODEL-BASED REFINEMENT

- Model-based refinement
  - requires 3D model of scene
  - run rigid ICP from our inferred pose between observed image and model

![Chart showing the proportion of frames correct for different scenes and methods.

Baseline: Tiny-Image Depth
Baseline: Tiny-Image RGB
Baseline: Tiny-Image RGB-D
Baseline: Sparse RGB
Ours: Depth
Ours: DA-RGB
Ours: DA-RGB + D

References:
Besl & McKay PAMI 1992]
AR VISUALIZATION WITH TRACKING AND REFINEMENT

- RGB input + AR overlay
- depth input + AR overlay
- rendering of model from inferred pose

simple robust frame-to-frame tracking and ICP-based model refinement enabled
Fire Scene

SCoRe Forest (single frame at a time)

SCoRe Forest + simple robust frame-to-frame tracking

SCoRe Forest + simple robust frame-to-frame tracking + ICP refinement to 3D model

RGB input + AR overlay  depth input + AR overlay  rendering of model from inferred pose
**Pumpkin Scene**

SCoRe Forest (single frame at a time)

SCoRe Forest + simple robust frame-to-frame tracking

SCoRe Forest + simple robust frame-to-frame tracking + ICP refinement to 3D model
- Train one SCoRe Forest per scene
- Test frame against all scenes
- Scene with lowest energy wins
- Single frame only

<table>
<thead>
<tr>
<th>Scene</th>
<th>Chess</th>
<th>Fire</th>
<th>Heads</th>
<th>Office</th>
<th>Pumpkin</th>
<th>RedKitchen</th>
<th>Stairs</th>
</tr>
</thead>
<tbody>
<tr>
<td>Chess</td>
<td>100.0%</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Fire</td>
<td>2.0%</td>
<td>98.0%</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Heads</td>
<td></td>
<td></td>
<td>100.0%</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Office</td>
<td>0.5%</td>
<td>4.0%</td>
<td>95.5%</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Pumpkin</td>
<td>2.8%</td>
<td>1.2%</td>
<td>3.6%</td>
<td>92.4%</td>
<td>100.0%</td>
<td></td>
<td></td>
</tr>
<tr>
<td>RedKitchen</td>
<td>2.8%</td>
<td>1.2%</td>
<td>3.6%</td>
<td>92.4%</td>
<td>100.0%</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Stairs</td>
<td></td>
<td></td>
<td>10.0%</td>
<td>90.0%</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Scene coordinate regression forests
- provide a single-step alternative to detection/description/matching pipeline
- can be applied at any valid pixel, not just at interest points
- allow accurate relocalization without explicit 3D model

Tracking-by-detection is approaching temporal tracking accuracy
Depth cameras are having huge impact

Decision forests + big data

Unifying principal:
Per-pixel regression and per-image model fitting
Thank you!

With thanks to:

Andrew Fitzgibbon, Jon Taylor, Ross Girshick, Mat Cook, Andrew Blake, Toby Sharp, Pushmeet Kohli, Ollie Williams, Sebastian Nowozin, Antonio Criminisi, Mihai Budiu, Duncan Robertson, John Winn, Shahram Izadi

The whole Kinect team, especially: Alex Kipman, Mark Finocchio, Ryan Geiss, Richard Moore, Robert Craig, Momin Al-Ghosien, Matt Bronder, Craig Peeper