Automatic and Efficient Long Term Arm and Hand Tracking for Continuous Sign Language TV Broadcasts

Tomas Pfister\textsuperscript{1}, James Charles\textsuperscript{2}, Mark Everingham\textsuperscript{2}, Andrew Zisserman\textsuperscript{1}

\textsuperscript{1}Visual Geometry Group
University of Oxford

\textsuperscript{2}School of Computing
University of Leeds
Motivation

Automatic sign language recognition:

- We want a large set of training examples to learn a sign classifier.
  - We obtain them from signed TV broadcasts.
- Exploit correspondences between signs and **subtitles** to automatically learn signs.
- Use the resulting sign-video pairs to train a sign language classifier.
Objective

Find the position of the head, arms and hands

- Use arms to disambiguate where hands are
Difficulties

- Colour of signer similar to background
- Overlapping hands
- Hand motion blur
- Faces and hands in background
- Changing background
Overview

Our approach:
- **First**: Automatic signer segmentation
- **Second**: Joint detection
Hand detection for sign language recognition

**State-of-the-art**: Long Term Arm and Hand Tracking for Continuous Sign Language TV Broadcasts [Buehler et al., BMVC’08]

**Method**: generative model of foreground & background using a layered pictorial structure model

**Necessary user input**: 75 annotated frames per one hour of video (3 hours work)

**Performance**: accurate tracking of 1 hour long videos, but at a cost of 100s per frame
Hand detection for sign language recognition

**State-of-the-art:** Long Term Arm and Hand Tracking for Continuous Sign Language TV Broadcasts [Buehler et al., BMVC’08]

**Method:** generative model of foreground & background using a layered pictorial structure model

**Necessary user input:**
75 annotated frames per one hour of video (3 hours work)

- **Colour & shape model**
- **HOG templates**

**Out work – automatic and fast!**

Find pose with minimum cost

Input

- Find pose with minimum cost

No manual annotation

Runs in real-time

**Performance:** accurate tracking of 1 hour long videos, but at a cost of 100s per frame
Overview

Our approach:
- **First**: Automatic signer segmentation
- **Second**: Joint detection
The problem

- How do we segment the signer out of a TV broadcast?
One solution: depth data (e.g. Kinect)

- Using depth data, segmentation is easy

- But we only have 2D data from TV broadcasts…
Constancies

- How do we segment a signed TV broadcast?

Clearly there are many constancies in the video

- Box contains changing background
- Signer never crosses this line
- Same signer
- Part of the background is always static
Co-segmentation

- Exploit constancies to help find a **generative model** that describes all layers in the video
Co-segmentation – overview

Method: co-segmentation – consider all frames together

For a sample of frames obtain …

Background

Foreground colour model

… and use the background and the foreground colour model to obtain

Per-frame segmentations

hist(...)
Find a “clean plate” of the static background

- Roughly segment a **sample of frames** using GrabCut
- Combine background regions with a median filter

Use this to refine the final foreground segmentation
Foreground colour model

Find a colour model for the foreground in a **sample of frames**
- Find faces in a sub-region of the video
- Extract a colour model from a region based on the face position

Use this as a global colour model for the final GrabCut segmentation
Qualitative co-segmentation results
Overview

Our approach:
- **First**: Automatic signer segmentation
- **Second**: Joint detection
Segmentations are not always useful for finding the exact location of the hands

Skin regions give a strong clue about hand location

**Solution**: find a colour model of the skin/torso

**Method**:
- skin colour from a face detector
- torso colour from foreground segmentations (face colour removed)

Improves generalisation to unseen signers
Overview

Our approach:
- **First**: Automatic signer segmentation
- **Second**: Joint detection
Joint position estimation

- **Aim**: find joint positions of head, shoulders, elbows and wrists
- Train from Buehler et al.’s joint output
Random Forests

- **Method**: Random Forest multi-class classification
- **Input**: skin/torso colour posterior
- **Classify** each pixel into one of 8 categories describing the body joints
- **Efficient simple node tests**

\[
\begin{align*}
  f(a) &= x_a \\
  f(a, b) &= |x_a - x_b| \\
  f(a, b) &= x_a^+ - x_b^+ \\
  f(a, b) &= x_a^+ + x_b^-
\end{align*}
\]

- **Colour posterior**
- **Random forest**
- **PDF of joints**
- **Estimated joints**
Evaluation: comparison to Buehler et al.

- Joint estimations compared against joint tracking output by Buehler et al.
Evaluation: comparison to Buehler et al.
Evaluation: quantitative results

Our method vs. Buehler et al. compared against manual ground truth

- Buehler et al. (2008)
- Our Method

e.g. 80% of wrist predictions are within 5 pixels of ground truth

Manual ground truth
Evaluation: problem cases

- Left and right hands are occasionally mixed

- Occasional failures due to a person standing behind the signer
Evaluation: generalisation to new signers

Trained & tested on **same** signer

Trained & tested on **different** signers

Generalises to new signers
Conclusion:

- Presented method which finds the position of hands and arms automatically and in real-time
- Method achieves reliable results for hours of tracking and generalises to new signers

Future work:

- Adding spatial model to avoid mixup of hands

Web page:

- This presentation is online at: http://www.robots.ox.ac.uk/~vgg/research/sign_language
Dynamic background

- How do we find the rectangle spanning the dynamic background?
- Reverse the question: **what area is permanently not dynamic?**

- The rectangle in the result spans the dynamic background
Parameter optimisation experiments
The following computation times are on a 2.4GHz Intel Quad Core I7 CPU with a $320 \times 202$ pixel image. The computation time for one frame is 0.14s for the co-segmentation algorithm and 0.1s for the random forest regressor, totalling 0.2s (5fps). The per-frame initialisation timings of the co-segmentation algorithm are 6ms for finding the dynamic background layer and static background, 3ms for obtaining a clean plate and 5ms for finding the image sequence-wide foreground colour model, totalling 14ms (approx. 24min for a 100K frames). Each tree, as used in our single signer random forests, takes 4.5 hours to train.
Todo: decide exactly what else to show. Could do a video of the above and use it to explain?