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

Tomas Pfister[1]
tp@robots.ox.ac.uk
James Charles[2]
j.charles@leeds.ac.uk
Mark Everingham[2]
m.everingham@leeds.ac.uk
Andrew Zisserman[1]
az@robots.ox.ac.uk

[1] Department of Engineering Science
University of Oxford
Oxford, UK

[2] School of Computing
University of Leeds
Leeds, UK

Abstract

We present a fully automatic arm and hand tracker that detects joint positions over continuous sign language video sequences of more than an hour in length. Our framework replicates the state-of-the-art long term tracker by Buehler *et al.* (IJCV 2011), but does not require the manual annotation and, after automatic initialisation, performs tracking in real-time. We cast the problem as a generic frame-by-frame random forest regressor without a strong spatial model.

Our contributions are (i) a co-segmentation algorithm that automatically separates the signer from any signed TV broadcast using a generative layered model; (ii) a method of predicting joint positions given only the segmentation and a colour model using a random forest regressor; and (iii) demonstrating that the random forest can be trained from an existing semi-automatic, but computationally expensive, tracker.

The method is applied to signing footage with changing background, challenging imaging conditions, and for different signers. We achieve superior joint localisation results to those obtained using the method of Buehler *et al.*

1 Introduction

There is growing evidence that large-scale weakly supervised learning can contribute to sign language recognition: using the correlations between subtitles and signers in signed TV broadcasts both Buehler *et al.* [3] and Cooper and Bowden [7] were able to automatically extract sign-video pairs from TV broadcasts; these automatically extracted sign-video pairs can then be used as supervisory material to train a sign language classifier [4]. However, current research in this area has been held back by the difficulty of obtaining a sufficient amount of training video with the arms and hands of the signer annotated. This is a great pity because there is a practically limitless supply of such signed TV broadcasts.

The standard approach of Buehler *et al.* [5] for tracking arms and hands requires manual labelling of 64 frames per video, which is around three hours of manual user input per one

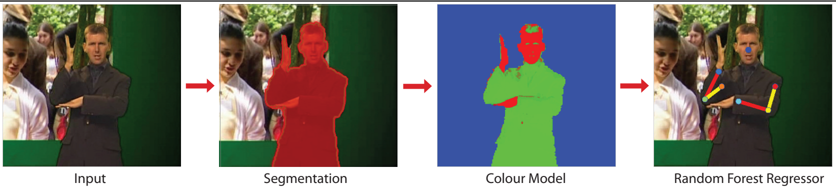


Figure 1: Arm and hand joint positions are predicted by first segmenting the signer using a layered foreground/background model, and then feeding the segmentation together with a colour model into a random forest regressor.

hour of TV footage. In addition, the tracker (by detection) is based on expensive computational models and requires hundreds of seconds computation time per frame. These two factors have hindered the large scale application of this method.

In this paper we describe a method for tracking joint positions (of arms and hands) without any manual annotation and, once initialised, the system runs in real-time. The three key ideas are (i) for signed video the signer can be segmented automatically using co-segmentation, (ii) given the segmentation, the joint positions can be predicted using a random forest, and (iii) the random forest can be trained using Buehler *et al.*'s tracking output, with no manual annotation. We show that the random forest trained in this manner generalizes to new signers. Figure 1 illustrates the processing steps.

In more detail, our source material consists of signed TV broadcasts, such as BBC news footage or educational programs. This is very challenging material to segment and determine human joint positions on for a number of reasons that include self-occlusion of the signer, self-shadowing, motion blur due to the speed of the signing, and in particular the changing background (since the signer is superimposed over a moving video that frequently even contains other people, *e.g.* see Figures 1 and 3).

Random forests for pose estimation. In recent years there has been increasing interest in random forest/fern-based methods for tasks such as image classification [2, 19], object detection [8, 11], segmentation [12, 25], head pose estimation [1] and feature extraction [18, 23]. In particular we are interested in the work on human pose estimation, where random forests have been used to obtain head pose [9], detect body parts [20] and infer full body pose [13, 24]. However, the success of these pose methods depends upon the use of depth imagery which is colour and texture invariant, while also making background subtraction much easier. Here we propose an upper body pose estimation method that exploits the efficiency and accuracy of random forests without the need for depth images, and instead use raw RGB images with only a partially known background (as described below).

Co-segmentation for signer extraction. Co-segmentation methods [6, 14, 16, 22] consider sets of images where the appearance of foreground and/or background share some similarities, and exploit these similarities to obtain accurate foreground-background segmentations. In our case we exploit the fact that sign language broadcasts consist of a layered model of the foreground and two separate backgrounds, one that is static throughout each video and another that changes with each frame. The signer stands partially against the static background and partially against the changing background (which they are describing).

To this end we propose a co-segmentation algorithm that automatically separates signers from any signed TV broadcast by building a generative layered model. We use this layered model of the signer in conjunction with a foreground colour model to provide a suitable input representation for the random forest regressor, superior to using the raw input image itself, and not requiring depth data.

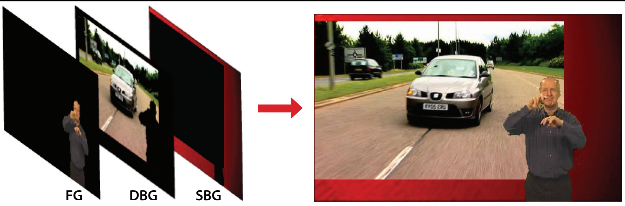


Figure 2: Generative layered model of each frame. The co-segmentation algorithm separates the signer from any signed TV broadcast by building a layered model consisting of a foreground (FG), dynamic background (DBG) and static background (SBG).

2 Co-segmentation Algorithm

The goal of the co-segmentation algorithm is to segment the overlaid signer from each frame of the broadcast. We exploit the fact that sign language broadcasts consists of an explicit layered model as illustrated in Figure 2. In the spirit of a generative model, we exploit these inherent layers to provide an accurate segmentation of the signer.

The static background layer (SBG) essentially consists of the framing (around the actual/original broadcast) that has been added by the studio. As can be seen in Figure 3, the static background is partially revealed and partially occluded in each frame depending on the position of the signer. In a similar manner to how a "clean plate" is constructed in film post-production, by looking through the whole video and combining the partially revealed static backgrounds one can nearly fully reconstruct the actual static background. This layer can then be exploited when segmenting the signer.

The dynamic background layer (DBG) consists of a fixed rectangle, where the original video is displayed, but is always partially covered by the signer and changes from one frame to another. Its colour information, for the region where it does not overlap a bounding box on the signer, is modelled separately and forms a background distribution for a subsequent segmentation of the signer.

Finally, the foreground layer (FG) consists of the moving signer. By assuming that the colour distribution of the signer remains constant we can build an accurate foreground colour model for the whole video.

Algorithm overview. The input to the co-segmentation algorithm is a signed TV broadcast video, and the output is a foreground segmentation, a quality score for the segmentation, the head position and a colour model for the skin and torso. These will be used in the random forest regressor. The algorithm consists of two main steps:

Automatic initialisation (per image sequence). To exploit the inherent layered model we initialise the algorithm by determining the "clean plate" (explained above), the dynamic rectangle and the foreground colour model. The details of how this "initialisation set" is obtained are given in Section 2.1.

Segmentation with a layered model and area constraints (per frame). The initialisation set is then used to derive an accurate hard segmentation of the signer. The clean plate and an area constraint are used to refine an initial rough segmentation. The details of this method are given in Section 2.2.

2.1 Co-segmentation initialisation

Our goal here is to obtain the layers and their layout that are common to the video sequence (in order to enable the subsequent per-frame segmentation). In detail, we wish to obtain the regions shown in Figure 3, as well as the foreground colour distribution. Our approach is to treat each frame as being generated from a number of layers, as depicted in Figure 2, and

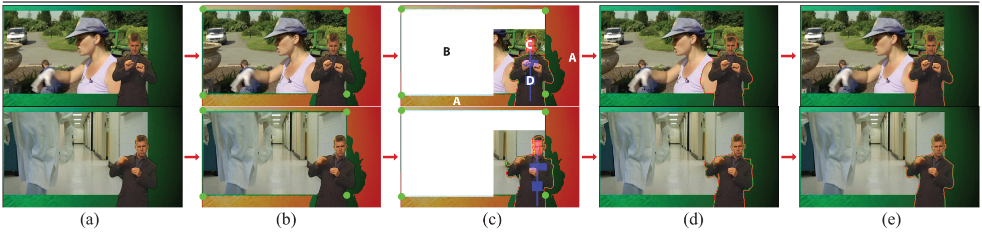


Figure 3: Co-segmentation. (a) the original frames; (b) the dynamic layer (rectangle spanned by the green dots) and the permanently fixed background (in red); (c) the rough segmentation with clamping regions for running graph cut. A is the permanently fixed background; B is the clamping region for the dynamic background; C is a soft foreground clamp and D is a hard foreground clamp. (d) the initial segmentation; (e) segmentation after clean plate and area size refinements.

to thereby solve for the layers and layout. This problem differs from typical applications of generative layered models for video, *e.g.* [15, 17] since part of the background in the video is always moving so we have a dynamic rather than fixed layer. The creation of the layered model can be broken down into a step per layer:

Dynamic background. We wish to find the rectangle that contains the DBG, and furthermore divide it into a region where the signer may overlap, and another where the signer never reaches (see Figure 3c). The latter region will be used to define per-frame background colour. To this end we find pixels that change values for the majority of frames and compute their rectangular bounding box, as shown in Figure 3b. Similarly, as a by-product, we obtain an area that is permanently static throughout the image sequence (region A in the same figure). This area will be used as a permanent BG clamping region.

Static background. Once the DBG has been found we find the SBG. As described above the SBG can be viewed as consisting of a "clean plate". The idea behind this is that by looking through the whole video and combining the partially revealed SBGs one can nearly fully reconstruct the actual SBG. We can then say with near-certainty whether a pixel is FG or BG. The clean plate is obtained by roughly segmenting a random set of frames into FG (signer) and BG using a graph cut algorithm. The regions used to obtain the FG and BG distributions are illustrated in Figure 3c. In particular, areas selected relative to the position of the signer's face (the face detection method is described below) is used to initialise the FG colour distribution. Given these segmentations, the clean plate (example shown in Figure 3d) is obtained as a median over the BG.

Foreground colour model. Here the aim is to obtain the signer colour distribution (which is assumed approximately constant throughout the sequence). The colour distribution (which is represented by a histogram) is obtained from the rough FG segmentations (see Figure 3c, computation described above) using frames where the colour histograms of the FG and DBG differ the most. The high colour difference increases the likelihood that there is a high contrast between the FG and BG and thus that the segmentation is correct.

Face detection. Face detection is used for initialisation and frame-by-frame segmentation. Detection of both frontal and profile view faces is done by choosing between the OpenCV face detector (high recall for frontal faces) and a face detector based on upper body detection [10] (lower recall but detects profile views) according to their confidence values.

2.2 Segmentation with a layered model and area constraints

Having finished the initialisation step we now have a layered model that can be employed for deriving a hard segmentation of the signer. The initialisation set is used to (i) compute a separate colour model for the dynamic background; (ii) improve the segmentation by comparing each pixel against the clean plate (yields near-certainty about whether a pixel is FG or BG as the BG is known); and (iii) provide a video-wide FG colour model (removes the need for finding accurate FG colour models for individual frames). Finally the FG segmentation is refined according to its area size (this corrects the segmentation where the FG catches part of the DBG because the colours of FG and DBG are similar). In detail there are three steps:

Initial segmentation. The aim of this step is to provide an initial segmentation that can be refined in the next steps. The segmentation uses GrabCut [21], with the FG colour model provided by the initialisation set and, as in Ferrari *et al.* [10], with the FG clamped in areas based on the face location (Figure 3c). The BG colour distribution is known from the DBG.

Refining the segmentation using the clean plate. In this step the initial segmentation is refined by using the clean plate of the SBG: the pixel is clamped to FG if the intensity of the segmentation and clean plate are dissimilar; otherwise clamped to BG. As illustrated in Figure 3e, by doing this we can effectively derive a perfect segmentation in the static layer for all pixels in the clean plate.

Refining the segmentation using the area size of the foreground. After the segmentation has been refined by the clean plate the FG is shrunk if it is much bigger than the average segmentations in the initialisation set. This is achieved by adding a constant to the graph cut unary potentials of the DBG (as this increases the likelihood that a larger part of the DBG is labelled as BG, hence also reducing the size of the FG). As illustrated in Figure 3e, this corrects the segmentation in frames where the FG catches part of the DBG because the FG colour model is similar to DBG colours.

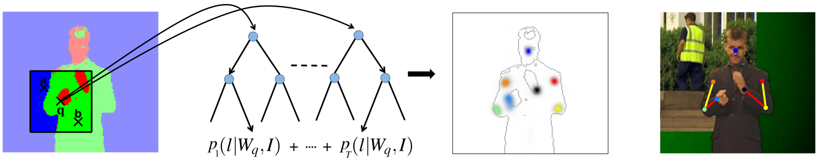
The segmentation still fails in certain difficult cases, *e.g.* when the colours of the FG and BG are very similar or when the face detector fails. To this end we compute a segmentation quality score based on the median FG area size.

Colour model initialisation At this stage we have a fully refined segmentation that is rated by a segmentation quality score. However, joint position prediction also benefits from information not explicit in the foreground segmentation, *e.g.* the colour of the hands and torso. The skin colour distribution comes from a patch of the face over several frames; the torso colour distribution comes from a set of FG segmentations from which the colours of the face/skin are automatically removed. The full segmentation, segmentation quality score, head position and colour model form the input to the random forest regressor for each frame.

3 Random Forest Regression

We cast the task of localising upper body arm joints and head position as a multi-class classification problem, classifying each image pixel into one of 8 categories $l \in$ {head centre, left/right wrist, left/right elbow, left/right shoulder, other} using a random forest classifier in a sliding-window fashion. From here on we also refer to "head centre" as a joint (see Figure 4d). As shown in Figure 4a, the input to the random forest comes from the colour model image after co-segmentation.

The random forest classifier uses simple features to make classification extremely computationally efficient. Classification to a discrete class label $l \in \{l_i\}$, for each pixel q across the image is performed in a sliding-window fashion. We classify the pixels by computing the conditional distribution $p(l|W_q, I)$ for each label, where I is the colour model image and W_q is the set of pixels in the window surrounding q. The random forest is an ensemble of T decision trees, as illustrated in Figure 4b. Each tree t consists of split nodes which perform a true or false test on incoming pixels. Pixels are recursively pushed down either the left



(a) Color model (b) Random forest (c) PDF of joints (d) Estimated joints

Figure 4: Estimating joint positions. (a) input colour model image; (b) random forest classifies each pixel using a sliding window and learnt test functions; (c) probability density function of each joint location, shown in different colours per joint (more intense colour implies higher probability); (d) joint estimates, shown as small circles linked by a skeleton.

or right branch depending upon the outcome of the test. When a pixel reaches a leaf at the bottom of the tree a learnt probability distribution $p_t(l|W_q,I)$ assigns the pixel a probability for class label l. The final conditional distribution $p(l|W_q,I)$ is obtained by taking an average across all trees in the forest as follows:

$$p(l|W_q,I) = \frac{1}{T}\sum_{t=1}^{T} p_t(l|W_q,I) \quad (1)$$

We use very efficient test functions $f(.)$ at the nodes of the trees which only compare pairs of pixel values [23]. A pixel q is represented by $\mathbf{x}_q = (x_q^1, x_q^2, x_q^3)$ where x_q^1, x_q^2, x_q^3 are the skin, torso and background colour posterior values at pixel q respectively [1]. The function f operates on a pair of pixels (a,b) from within the window W_q and produces a scalar value which is compared against a threshold value υ – see Figure 4a. These tests can take one of four forms: $f(a) = x_a^c$, or $f(a,b) = x_a^c - x_b^c$, or $f(a,b) = x_a^c + x_b^c$, or $f(a,b) = |x_a^c - x_b^c|$, where $c \in \{1,2,3\}$ indexes the type of colour posterior value to choose.

Training of the forest. In each frame of the video circular patches of radius 13 pixels centred on joint locations are labelled as that joint, with all other pixels labelled as 'other'. Each tree in the forest is trained by randomly sampling a diverse set of points S_n from the training frames. Each decision tree is trained recursively, with the split function and threshold at each node chosen to split the data reaching that node as "purely" as possible such that points belonging to the same class are sent to the same child node. The impurity of a split is measured using the Gini measure:

$$i(S_n) = 1 - \sum_l p(l|S_n)^2, \quad (2)$$

where $p(l|S_n)$ is represented by a histogram of the dataset S_n over possible labels l at node n. The Gini impurity is chosen for its efficient implementation compared to *e.g.* information gain. Because there are many more 'other' pixels than 'joint' pixels, we balance the dataset by normalising the number of elements in the bin labelled l by the total number of elements in the training set labelled l. The parameters of split nodes are learnt by trying all possible test functions $f(.)$ and colour posterior types c for a randomly sampled offset pixel (a,b). The offset pixel is uniformly sampled within W_q, where $q \in S_n$. The data entering the node is split into a left subset S_n^L if $f(.) < \upsilon$ or otherwise to a right subset S_n^R.

The drop in impurity is measured as $\triangle i(S_n) = i(S_n) - P_L i(S_n^L) - (1-P_L) i(S_n^R)$, where P_L is the fraction of data points that go to the left set. In each case the threshold value υ is chosen to maximise $\triangle i(S_n)$. The whole process is repeated k times (we use $k = 200$) and the set of parameters which maximise $\triangle i(S_n)$ overall is chosen as the winning decision function. This process is recursively repeated for all nodes. A node is declared a leaf node, and not split further, when (i) the maximum depth limit D of the tree has been reached or (ii) the node is pure *i.e.* all points reaching the node have the same class label. A per-leaf probability distribution $p_t(l|W_q)$ is stored at the leaf node, represented as a normalised histogram over the labels of all data points reaching the node.

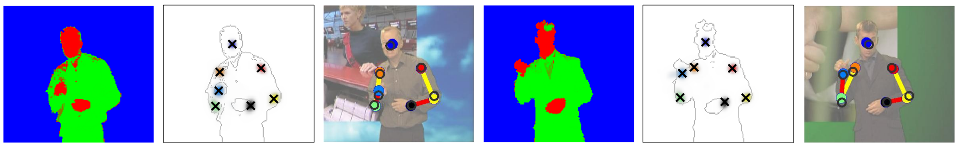

(a) (b) (c) (d) (e) (f)

Figure 5: Joint estimation. (a) and (d) show a colour model image from which we obtain probability densities of joint locations shown on top of colour model edge image in (b) and (e). Different colours are used per joint (higher intensity colour implies higher probability). Maximum probability per joint is shown as black crosses. The filled in circles linked by a skeleton in (c) and (f) are estimated joints overlaid on faded original frame which are compared with ground truth joint locations, shown as open circles.

Assigning joint locations. A location for the joint l is found by using the output of the random forest $p(l|W_q)$ and estimating the density of joint proposals using a parzen-window kernel density estimator with a Gaussian kernel. The position of maximum density is used as the joint estimate.

See Figure 5 for an illustration of this method and comparison against ground truth.

4 Experimental Results

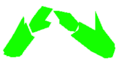
First the co-segmentation algorithm is evaluated (Section 4.1); then the performance of the joint position estimator is assessed (Section 4.2), and finally the computation time of the methods is discussed (Section 4.3). Sample videos are available online.[1]

4.1 Co-segmentation

The co-segmentation algorithm is evaluated on the arm and hand segmentation ground truth provided by Buehler *et al.* [5]. This ground truth consists of manually labelled left arm, right arm and hand segmentations for 342 frames for one signer. As in [5], an overlap measure is defined as $o = \frac{T \bigcap M}{T \bigcup M}$, where T is manual ground truth segmentations and M is the mask generated from an estimated configuration. This overlap score is evaluated separately for the left arm, the right arm and the hands. The overall overlap is defined as the mean over these three values. The mean results are given in Figure 6. As expected the left arm is the hardest to segment as it is always on top of a dynamic background.

Body part	Overlap score
Left arm	0.911
Right arm	0.957
Hands	0.975
Mean	0.951

(a)


(b)

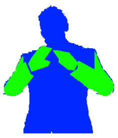

(c)

Figure 6: Co-segmentation evaluation. (a) the overlap scores for each body part; (b) example of the ground truth; (c) the segmentation (blue) evaluated against the ground truth (green).

4.2 Random forest regression

The joint estimation method is evaluated in two experiments. The first experiment demonstrates the effect of varying the most influential parameters of the random forest. The second experiment tests the hypothesis that using the segmented colour posterior image (Seg+CP)

[1] http://www.robots.ox.ac.uk/~vgg/research/sign_language

improves joint estimation results. This second experiment is conducted in two settings: (1) training and testing on the same signer (single-signer forest), as reported by Buehler *et al.* [5], and (2) training on multiple signers and testing on an *unseen* signer to demonstrate generalisation (multi-signer forest).

Dataset. We train and test on five videos, each video containing a different signer. One video typically contains over 40K frames. The training frames consist of the first 60% of each video and the testing frames the remaining 40%. Joint location output from a slow reliable tracker by Buehler *et al.* is used to assign labels to pixels in the training videos.

Sampling poses (training and testing). The forests are trained and tested on samples that cover a diverse range of poses given by Buehler *et al.*'s tracker. Poses are clustered using k-means into 100 clusters, and frames are uniformly sampled across clusters. Increasing the diversity of poses in the training set increases generalisation to testing data and improves accuracy on unusual poses. In the testing set it ensures that accuracy is not biased towards poses which occur more frequently, *e.g.* "resting" poses between signs.

Training data. When training and testing on the same signer, each tree is trained on a sample of 1,000 frames taken from a single video. Generalisation across signers is evaluated using 5-fold cross validation: the random forests are trained on 4 signer videos and evaluated on a 5th "held-out" video, and a sample of 1,000 frames from the training set (250 diverse frames from each of the 4 videos) is used for training.

Testing data. All experiments compare against manually annotated ground truth. 200 frames containing a diverse range of poses from the testing set of each of the 5 videos were annotated with joint locations (1000 frames in total).

Evaluation measure. An estimated joint is deemed correctly located if it is within a set distance of d pixels from the ground truth. Accuracy is measured as the percentage of correctly estimated joints. The experiments use a distance of $d = 5$ pixels from ground truth (signer shoulder width in Figure 4d $\approx$ 55 pixels).

Parameter optimisation experiment. Figure 7 shows the average joint estimation accuracy when varying the random forest parameters. It is noted that: (1) Regardless of input type, the largest window width in the experiment (91 pixels) produced best results. Window width is kept at this value in all experiments. (2) Maximum accuracy is obtained for single-signer forests at a depth of 128, however for only a small drop in accuracy ($< 1\%$) a large speed increase in computation time (30%) can be achieved by using trees at depth 64. In contrast, for silhouette input a larger tree depth increases accuracy significantly. Trees with depth 64 have on average $\approx$ 5,000 leaf nodes (0.1% of training pixels) indicating the trees are highly imbalanced. (3) An optimal lower depth of 64 is necessary to help the multi-signer forests generalise across unseen signers. (4) 8 trees in the forests produces best accuracy. (5) Figure 8 shows that forests trained and tested on the same signer are more accurate for smaller evaluation distance measures d than forests tested on unseen signers. However, the penalty for a looser fit to ground truth is small if one wants to generalise over unseen signers.

Random forest *vs* Buehler *et al.* experiment. Tables 1 and 2 show the results for single-signer and multi-signer forests respectively compared against manually annotated ground truth joint locations. A comparison against Buehler *et al.*'s tracker is also shown. Table 1 shows the single-signer per-joint accuracy averaged over all training videos. The results are shown for different inputs: (i) a raw colour pixel representation in LAB (LAB), (ii) colour posterior on the whole image (CP), (iii) signer silhouette (S), (iv) segmented colour posterior (Seg+CP). Using Seg+CP as input produces best results. Using LAB pixels performs similarly to using CP in the single-signer case. Occasionally the segmentation misses the

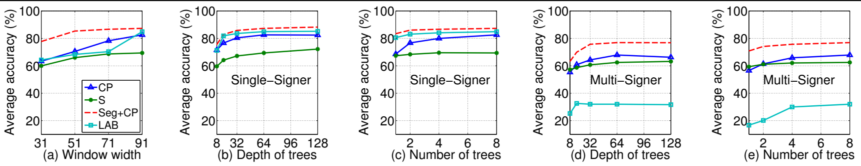


Figure 7: Effect of altering random forest parameters for different input types. (a) window width, (b) & (d) depth of trees for single-signer & multi-signer forests respectively, (c) & (e) number of trees for single-signer & multi-signer forests respectively.

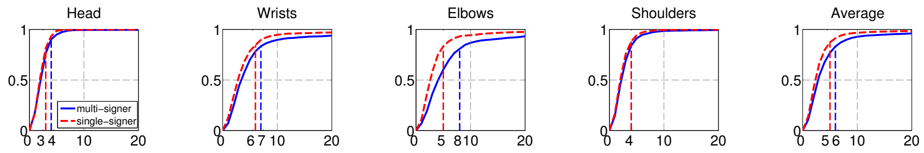


Figure 8: Per-joint average accuracy plotted against allowed distance from manual ground truth in pixels. 80% accuracy is shown as vertical dotted lines.

hand against the dynamic background, causing CP to outperform Seg+CP for the right wrist in the single-signer case. Table 2 shows that by using Seg+CP the multi-signer forests can generalise better to unseen signers.

For both single-signer and multi-signer forests we observe the results to be on average better than those of Buehler *et al.*'s. This suggests noisy data from Buehler *et al.*'s tracker is smoothed over by more consistent data at the leaf nodes in the trees. The accuracy scores of the random forests are promising even when measured tightly against ground truth joints (5 pixels leeway). This suggests very robust tracking can be achieved if a looser fit is required.

4.3 Computation time

The following computation times are on a 2.4GHz Intel Quad Core I7 CPU with a 320×202 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.

5 Conclusion

We have presented a fully automatic arm and hand tracker that detects joint positions over continuous sign language video sequences of more than an hour in length. Our framework attains superior performance to a state-of-the-art long term tracker [5], but does not require the manual annotation and, after automatic initialisation, performs tracking in real-time.

Acknowledgements: We are grateful to Lubor Ladicky for discussions, and to Patrick Buehler for his very generous help. Funding is provided by the EPSRC.

Method	Head	R Wrist	L Wrist	R Elbow	L Elbow	R shldr	L Shlder	Average
LAB	**98.0**	63.9	**85.8**	67.6	79.2	**87.4**	86.1	81.1
CP	97.7	**70.3**	82.9	67.9	70.0	84.3	72.6	78.0
S	91.9	22.2	30.8	67.8	78.8	82.2	89.0	66.1
Seg+CP	97.6	64.9	84.1	**72.5**	**80.2**	86.8	**92.0**	**82.6**
Buehler *et al.* [5]	96.4	58.8	66.0	67.6	71.5	83.1	83.7	75.3

Table 1: Average accuracy of per-joint estimates for single-signer forests measured as 5 pixels from manual ground truth. Using Seg+CP outperforms all other input types.

Method	Head	R Wrist	L Wrist	R Elbow	L Elbow	R shldr	L Shlder	Average
LAB	56.8	7.6	14.8	22.8	37.4	36.8	47.8	32.0
CP	93.8	52.9	**80.4**	30.8	62.1	75.7	79.4	67.9
S	88.4	15.6	18.4	59.8	**78.6**	85.0	91.4	62.5
Seg+CP	95.0	**60.3**	80.0	57.3	63.4	**88.0**	**94.5**	**76.9**
Buehler *et al.* [5]	**96.4**	58.8	66.0	**67.6**	71.5	83.1	83.7	75.3

Table 2: Average accuracy of per-joint estimates for multi-signer forests evaluated against manual ground truth.

References

[1] B. Benfold and I. Reid. Colour invariant head pose classification in low resolution video. In *Proc. BMVC*, 2008.

[2] A. Bosch, A. Zisserman, and X. Munoz. Image classification using random forests and ferns. In *Proc. ICCV*, 2007.

[3] P. Buehler, M. Everingham, and A. Zisserman. Learning sign language by watching TV (using weakly aligned subtitles). In *Proc. CVPR*, 2009.

[4] P. Buehler, M. Everingham, and A. Zisserman. Employing signed TV broadcasts for automated learning of British sign language. In *Workshop on Representation and Processing of Sign Languages*, 2010.

[5] P. Buehler, M. Everingham, D. P. Huttenlocher, and A. Zisserman. Upper body detection and tracking in extended signing sequences. *IJCV*, 95(2):180–197, 2011.

[6] Y. Chai, V. Lempitsky, and A. Zisserman. Bicos: A bi-level co-segmentation method for image classification. In *Proc. ICCV*, 2011.

[7] H. Cooper and R. Bowden. Learning signs from subtitles: A weakly supervised approach to sign language recognition. In *Proc. CVPR*, 2009.

[8] A. Criminisi, J. Shotton, D. Robertson, and E. Konukoglu. Regression forests for efficient anatomy detection and localization in CT studies. In *MICCAI workshop on Probabilistic Models for Medical Image Analysis*, 2011.

[9] G. Fanelli, J. Gall, and L. Van Gool. Real time head pose estimation with random regression forests. In *Proc. CVPR*, 2011.

[10] V. Ferrari, M. Marin-Jimenez, and A. Zisserman. Progressive search space reduction for human pose estimation. In *Proc. CVPR*, 2008.

[11] J. Gall and V. Lempitsky. Class-specific hough forests for object detection. In *Proc. CVPR*, 2009.

[12] E. Geremia, O. Clatz, B.H. Menze, E. Konukoglu, A. Criminisi, and N. Ayache. Spatial decision forests for MS lesion segmentation in multi-channel magnetic resonance images. *NeuroImage*, 2011.

[13] R. Girshick, J. Shotton, P. Kohli, A. Criminisi, and A. Fitzgibbon. Efficient regression of general-activity human poses from depth images. In *Proc. ICCV*, 2011.

[14] D.S. Hochbaum and V. Singh. An efficient algorithm for co-segmentation. In *Proc. ICCV*, 2009.

[15] N. Jojic and B. Frey. Learning flexible sprites in video layers. In *CVPR*, volume 1, pages 199–206, 2001.

[16] A. Joulin, F. Bach, and J. Ponce. Discriminative clustering for image co-segmentation. In *Proc. CVPR*, 2010.

[17] M. P. Kumar, P. H. S. Torr, and A. Zisserman. Learning layered motion segmentations of video. *IJCV*, 76:301–319, 2008.

[18] V. Lepetit and P. Fua. Keypoint recognition using randomized trees. *IEEE PAMI*, 28 (9):1465–1479, 2006.

[19] R. Marée, P. Geurts, J. Piater, and L Wehenkel. Random subwindows for robust image classification. In *Proc. CVPR*, 2005.

[20] S. Nowozin, C. Rother, S. Bagon, T. Sharp, B. Yao, and P. Kohli. Decision tree fields. In *Proc. ICCV*, 2011.

[21] C. Rother, V. Kolmogorov, and A. Blake. Grabcut: interactive foreground extraction using iterated graph cuts. In *Proc. ACM SIGGRAPH*, 2004.

[22] C. Rother, T. Minka, A. Blake, and V. Kolmogorov. Cosegmentation of image pairs by histogram matching-incorporating a global constraint into MRFs. In *Proc. CVPR*, 2006.

[23] J. Shotton, M. Johnson, and R. Cipolla. Semantic texton forests for image categorization and segmentation. In *Proc. CVPR*, 2008.

[24] J. Shotton, A. Fitzgibbon, M. Cook, T. Sharp, M. Finocchio, R. Moore, A. Kipman, and A. Blake. Real-time human pose recognition in parts from single depth images. In *Proc. CVPR*, 2011.

[25] P. Yin, A. Criminisi, J. Winn, and I. Essa. Tree-based classifiers for bilayer video Segmentation. In *Proc. CVPR*, 2007.