Experiments With Hand-tracking Algorithms in Video Conversations

Pihel Saatmann
Institute of Computer Science
University of Tartu
Estonia
pihels@gmail.com

Kristiina Jokinen
Institute of Computer Science
University of Tartu
Estonia
kristiina.jokinen@ut.ee

Abstract

This paper describes a simple colour-based object tracking plugin for the video annotation tool ANVIL. The tracker can be used to automatically annotate hand gestures or the movements of any object that is distinguishable from its background. The tracker records velocity duration and total travel distance of hand gestures and can be configured to display gesture direction. Results of the tracker are compared to manually created annotations for hand gestures. Data recorded by the tracker is not accurate enough to provide a complete alternative to manual annotation, but could rather be used as a basis for determining where hand gestures can be detected. Thus using the tracker in combination with a human annotator could significantly speed up the annotation process.

1 Introduction

Annotation of video data is an important prerequisite for human communication studies, but doing this manually is time and resource consuming. Analysis requires annotation by trained annotators, and although annotation tools such as Anvil (Kipp 2001) are available, the annotation process is slow and prone to inconsistencies and misconceptions. For instance, considering multimodal video annotation, one problematic issue is the segmentation of gestures and their temporal length: hand gestures can be considered to be the movement of hand(s) only, or they can also include the period after movement when the position of hands can be identified as static. Such differences can be eliminated by clear and unambiguous instructions to the annotators in advance, but also by automatic hand tracking algorithms where the detected hand movements provide a common set of elements to be analysed further by the annotators. Automatic gesture recognition also contributes to objective viewing of gesture elements and to more systematic treatment of data in general. It is thus useful to implement object tracking and gesture recognition algorithms in order to automatically annotate hand gestures, and given the large amounts of video data being collected in various projects, there is an urgent need for such advanced tools. The tracking tool described in this paper is intended to speed up the process of manual annotation, and help in the analysis by performing automatic annotations.

In this paper we describe a hand tracker plugin for Anvil. The goal is to create a technical solution that visually identifies hand movement on video files and tags this with descriptive and quantitative information. The plugin works in a similar manner as the plugin for face tracking (Jongejan 2012), but the hand tracker interface has controls for minimum saturation threshold and how many frames to skip on each iteration. Moreover, hand tracking is a more difficult task than face tracking, since the shape of the hand is not constant or as easily recognizable as that of the face, and hand movements are also rapid and irregular.

The paper is structured as follows. Section 2 gives an overview of the object detection and tracking algorithm CAMShift which is used in the paper. Section 3 describes the plugin for the Anvil while Section 4 gives an overview of the evaluation of the implementation. Section 5 provides discussion on the results, and Section 6 outlines possible solutions and future improvements, as well as draws conclusions and describes future work.

K. Jokinen and M. Vels. 2015. Proceedings of The 2nd European and the 5th Nordic Symposium on Multimodal Communication.

2 CAMShift Algorithm

Hand gesture recognition is a complex task which consists of several subtasks. As in object tracking in general, hand tracking algorithm must first detect the hand, then track the hand in each consecutive frame of a video, and finally estimate the trajectory and identify the complex movements as a gesture. Yilmaz et al. (2006) provide an overview of object tracking algorithms, and point out that an efficient algorithm also has to run in real-time and overcome problems such as image noise, poor or changing lighting, complex object shapes, irregular object motion, occlusion, and distractors present in the video. The objects of interest are represented by their shapes and appearances, and appropriate features are selected to distinguish them in the feature space. The four common features are colour, edges, optical flow and texture, and additional information about the object's orientation and shape can also be provided. Tracking algorithms often use a combination of various features (Han et al. 2009).

In this work we use the CAMShift (Continuously Adaptive Mean-Shift) algorithm (Bradski, 1998). CAMShift is a simple colour-based object tracking algorithm and it uses the HSV (Hue Saturation Value) colour system, which separates the colour (hue) of a pixel from the concentration of the colour (saturation) and brightness (value). For tracking an object only the hue data is used. Brightness and saturation values can be used to filter out noise caused by grey (low saturation) or white (high brightness) pixels which hue values cannot be accurately extracted.

CAMShift is based on the mean-shift algorithm, which works by finding the local maximum of a probability density function and iteratively moving a predefined search window until its centre is located over the maximum. In object tracking this means that if given an initial search window the algorithm will find the point in an image where the pixels matching the tracked object's hue value have the highest density and move the search window so it is centered over the point of maximum density. The process is repeated for each consecutive frame in a video recording and can be used to find the location of the tracked object in each frame. CAMShift was chosen for the implementation because it is fast, computationally efficient, and does not require prior training or a feature database.

3 Hand-tracking Annotation Plugin

The CAMShift algorithm is implemented as a plugin for the ANVIL annotation software (Kipp, 2001). ANVIL is a freeware video annotation tool that allows the user to create multi-layered colour-coded annotations (http://www.anvil-software.org/). The hand-tracker plugin produces a new track, camshift, in the Anvil specification file, where the detected gesture elements are marked.

The plugin is able to track hands and other coloured objects in a video and detect movements. The initial detection of the hand is left to the user by having them select a rectangular area that includes at least a part of the tracked object. The colour data for the selected area is used as a template for finding the tracked object in each consecutive video frame.

The tracker is able to track a single target at a time. If there is more than one object that matches the selected colour template in the video, it is possible that the tracker may switch targets at some point when another object of similar hue happens to be near the tracked object in the video. It is currently not possible to change the colour template for the tracked object after the tracker is started. In order to track a different coloured object the plugin must be restarted first so that a new area for tracking can be selected.

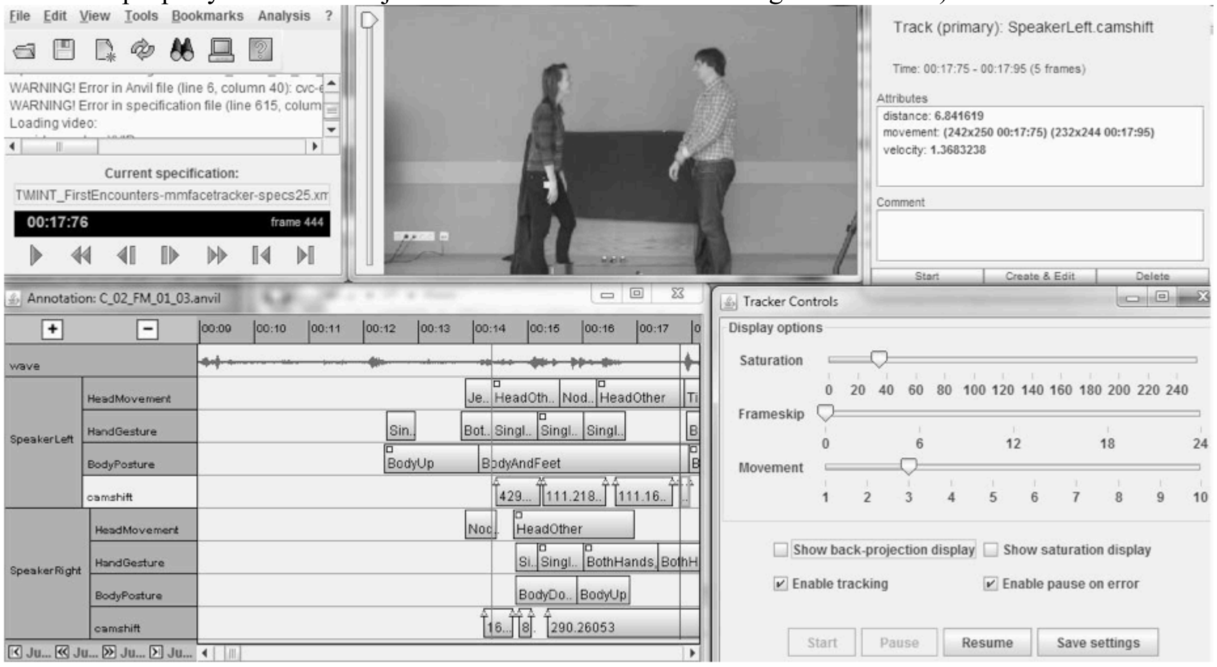
3.1 User Interface

The user interface to the Anvil and the tracker is given in Figure 1 (next page). Besides the Anvil-related windows of the annotation board, the video file, and the control and element information windows, the interface contains a specific control board for the plugin (Figure 1 bottom right). This includes controls for settings that can improve tracking accuracy and efficiency:

- **Saturation** – threshold for ignoring pixels with low saturation values.
- **Frameskip** – number of frames skipped after each processed frame.
- **Movement** – threshold for detecting movement in pixels.

Saturation can be used to filter out video noise. The possible range of saturation values is 0-255, usually filtering out pixels below the default value of 35 should be enough to improve tracking precision. With *Frameskip* the user can control the trade-off between computational cost and accuracy: higher values will lower computational costs, but may also lower tracking accuracy. *Movement* is used to control the length of the gesture. In each frame movement is measured by comparing the current location of the object to its location in the previous frame. If the difference between the two points is above the threshold in three consecutive frames, then a new annotation element is created and the movement is considered to be continued. If the difference is below the threshold in three consecutive frames, then no new annotation element is created, the previous annotation element will be completed and the movement is considered to have ended. This means that with higher values small or very slow movements will be ignored.

The best values for saturation and movement thresholds depend on the video quality and content. The user can change the values at any time during or before tracking. In order to achieve better accuracy different these values can be tried out depending on the user's needs. For example if the priority is to detect all movements in the video then the movement threshold should be lower, however if the user wishes to ignore small movements then the value can be set higher.
The user can also enable pausing the tracker when errors occur (for example when the tracked object becomes occluded and the search window is lost). The user can re-select the tracked object in the main video screen at any time, including when the tracker is paused. It is also possible to select a new object for tracking, however the colour data from the initial template will still be used (meaning the tracker will work properly if the new object is of similar colour to the original selection).



WARNING! Error in Anvil file (line 6, column 40): cvc-
WARNING! Error in specification file (line 615, colum
Loading video:

Current specification:
TWINT_FirstEncounters-mmfacetracker-specs25.xm

00:17:76 frame 444

Track (primary): SpeakerLeft.camshift

Time: 00:17:75 - 00:17:95 (5 frames)

Attributes
distance: 6.841619
movement: (242x250 00:17:75) (232x244 00:17:95)
velocity: 1.3683238

Comment

00:09 00:10 00:11 00:12 00:13 00:14 00:15 00:16 00:17

SpeakerLeft HeadMovement: Je.. HeadOth.. Nod.. HeadOther Ti
SpeakerLeft HandGesture: Sin.. Bot.. Singl.. Singl.. Singl.. B
SpeakerLeft BodyPosture: BodyUp BodyAndFeet B
SpeakerLeft camshift: 429... 111.218.. 111.16..
SpeakerRight HeadMovement: Noc.. HeadOther
SpeakerRight HandGesture: Si.. Singl.. BothHands BothH
SpeakerRight BodyPosture: BodyDo.. BodyUp
SpeakerRight camshift: 16.. 8. 290.26053

Saturation: 0 20 40 60 80 100 120 140 160 180 200 220 240
Frameskip: 0 6 12 18 24
Movement: 1 2 3 4 5 6 7 8 9 10

☐ Show back-projection display ☐ Show saturation display
☑ Enable tracking ☑ Enable pause on error

Figure 1. ANVIL and the tracker's user interface.

3.2 Anvil annotation elements

The tracked object's movements are automatically annotated by writing a new element containing data about the movement to a specified annotation track camshift in ANVIL (Figure 1 bottom left). In our experiments, the specification file contains multimodal behaviour tracks as specified in the NOMCO annotation scheme (Navarretta et al. 2012). In particular, it contains manually annotated gestures with which the detected gesture elements can be compared.

The recorded detailed information for each element can be viewed by clicking on a specific element on the annotation track. The details window (Figure 1 top right) shows information about the start and end point of the movement in the video, total movement distance and average velocity. The start and end point of the movement are also used to display a vector in the main video window between the two points.

3.3 Initial element selection

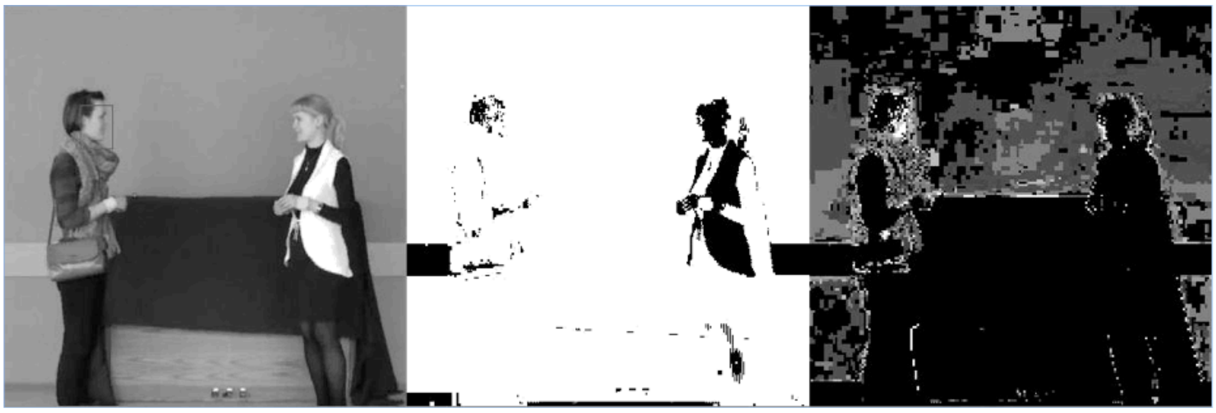


Figure 2. Normal video frame, saturation mask and probability map.

In order to select the most suitable area of an object for tracking, the user can enable the back projection display which shows the colour probability map of the current video frame. The colour probability map shows the most probable locations for the tracked object in the video frame, higher probability is represented by lighter pixel values. It is especially important to initially select as large an area of the object for tracking as possible but not include any parts of the background in the selection.

The saturation display shows pixels in the current video frame that have a value below the saturation threshold and are thus ignored when calculating the probability map. The saturation and probability representation of a video frame can be seen in Figure 2.

4 Evaluation

The plugin has been tested on real dialogue data recorded as part of the ETF (Estonian Science Foundation) project MINT (Multimodal INTeraction, Jokinen and Tenjes, 2012). The dialogue recordings feature two participants who are facing each other and are filmed from a side view as can be seen in Figure 2. The plugin was tested by tracking both bare hands and the yellow bracelets worn by some of the participants. The tracker has been evaluated by comparing annotations created by the tracker and manual annotations (see Section 4.2).

4.1 Tracking Bare Hands and Coloured Objects

The simple colour-based object tracking plugin proved to be problematic when tracking bare hands. The tracker would jump to other hands or faces in the frame if the regions overlapped (for example if the participants shook hands or crossed their arms) or get lost if the hand was completely occluded (for example if the person put their hands in their pockets). Movement was not accurately detected and very often small movements were detected when there actually was none. Detection accuracy was also affected when the participants had bare forearms as the tracker would then try to track the entire length of the bare arm instead of just the hand area.

However, when tracking the yellow bracelets, the tracker was working as expected. The tracking window didn't move to the wrong object and was lost only if the bracelet was completely occluded. Movement was detected relatively accurately. Tracking the bracelets was more accurate because the bracelets have a hue that is easily distinguishable from the background. In most cases it was the only object in the video of that particular colour. Even if more than one bracelet was present in the video it did not come into direct contact with other bracelets and thus the tracker would always stay on the right target.

4.2 Comparison to Manual Annotations

The tracker was applied to four randomly seclected video files that had been manually annotated for hand gestures following the NOMCO annotation scheme. The tracker was set to track the participants' yellow bracelets. In cases where one manually annotated movement was detected by the tracker as multiple movements, it was counted as one true positive detection. All single false positive annotation elements created by the tracker were counted as separate detections even if they were consecutive. In a more detailed comparison, the separated elements which have a distance below a threshold could be regarded as belonging to the same gesture, thus mimicking separation of the detected true positives. When the tracked object was lost then the tracker was manually reset to the correct position.

The average precision, recall and F-measure values for the tracker was calculated using the formulas below. True positive values are marked as TP, false positive as FP and false negative as FN. Table 1 shows a comparison of the tracker's movement detection ability to manual annotations.

Precision or positive predictive value represents the fraction of correct detections over all movements that were detected:

$$P = TP \div (TP + FP) \approx 0{,}306$$

Recall represents the fraction of correct detections over the number of movements that should have been detected:

$$R = TP \div (TP + FN) \approx 0{,}982$$

F-score is a weighted average of precision and recall and calculated as their harmonic mean:

$$F1 = 2 * P * R \div (P + R) \approx 0{,}455$$

Speaker	**TP**	**FP**	**FN**	**Precision**	**Recall**	**F-score**
1	22	68	0	0,244	1	0,393
2	9	43	0	0,173	1	0,295
3	17	24	0	0,415	1	0,586
4	60	84	4	0,417	0,938	0,577
5	63	96	3	0,396	0,955	0,560
6	25	107	0	0,189	1	0,318
Average				**0,306**	**0,982**	**0,455**

Table 1. Summary of the tracker's performance.

5 Discussion

The results show that the tracker is able to detect hand movements in video files, however the accuracy of gesture detection is low due to a large amount of false positive detections. As can be seen from the evaluation data the tracker has high recall values, but low precision due to a large number of false positive detections. The tracker also often detects multiple smaller movements where there is actually one long movement as seen in Figure 3. This is due to the tracked object's velocity dropping to very low values during some parts of the movement.

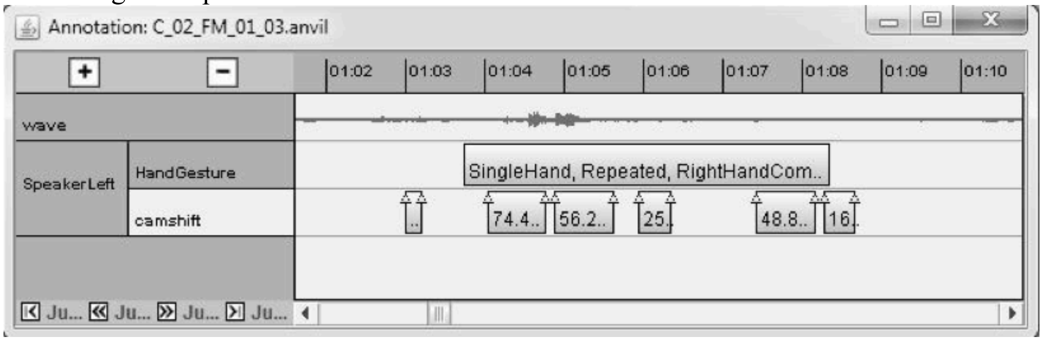


Figure 3. Comparision of manual (top track) and tracker (bottom track) annotation.

One of the main problems encountered during testing was that the tracker detects movement when there is none. Increasing the movement detection threshold would solve this problem, but then actual movements that are below the threshold (for example very slow movements) would also not be detected. However, the problem is alleviated by the fact that the manual annotations contained only data about hand movements that were deemed communicatively significant. Very small movements and those caused by the whole body moving were not annotated. Therefore in reality the number of false positives can be lower than represented by the evaluation results. In fact, this can be interpreted as a novel means to support more objective manual annotation as the tool detects gestures which the human annotators may not have even noticed. The annotators are thus provided with the same set of objectively recognised movements which they have to annotate and agree on their interpretation.

6 Conclusion

This paper discusses automatic hand tracking in video files and compares the automatic recognition results with human annotations. The tracker is good at detecting movements, but sometimes finds movement when there was none. Changing the detection threshold can improve the results, but is a trade-off since it would also prevent small movements being detected. In order to minimise false positive detections, the movement detection algorithm would have to be improved so that it would ignore very small movements yet be able to recognise long and slow movements as a single gesture. On the other hand, this can also be used as a basis for objective gesture segmentation and a starting point for the interpretation of communicative function of the detected hand movements.

The tracking and movement detection precision depend on the quality of the video being used and on the user specified settings. The tracker works better on objects that are easily distinguishable from their background by colour. Problems were identified when the hue of the hands was similar to the background colour. For fully automated annotations, possibilities to ignore distractors such as other hands present in the video and static background objects would have to be researched. Pre-processing the video material in order to improve quality could also be considered.

At current state the plugin cannot be used as a fully independent annotating tool as the number of false positive results is too high, and the tracker requires some human interference when the tracked object is lost. However, although the tool does not currently provide a complete alternative to manual annotation it can be used for creating a rough basis for annotations. Since evaluation showed that the tracker was able to detect all movements, the basis can be used to determine where in a video hand gestures can be detected. The user would be required to manually remove false positive detections.

We will continue to test the algorithm with different video files, and to evaluate the robustness of the algorithm. We will also test the quality of the results when the video quality goes down. Future work will also see detailed comparison of the linguistic-pragmatic annotations with the recognized gesture signals. The algorithm can be applied to human-robot interaction in order to study the robot's understanding of human gestures. Experiments on these lines are described in Han et al. (2012).

References

Gary R. Bradski. 1998. Computer video face tracking for use in a perceptual user interface. Intel Techno-logy Journal. Q2, pp.705-740.

Bing Han, Christopher Paulson, Taoran Lu, Dapeng Wu and Jian Li. 2009. Tracking of Multiple Objects under Partial Occlusion. Automatic target Recognition XIX, 7335. Available at: http://www.wu.ece.ufl.edu/mypapers/trackingSPIE09.pdf

Jing Guang Han, Nick Campbell, Kristiina Jokinen and Graham Wilcock. 2012. Investigating the use of non-verbal cues in human-robot interaction with a Nao robot. Proceedings of 3rd IEEE International Conference on Cognitive Infocommunications (CogInfoCom 2012), Kosice, 679-683.

Bart Jongejan. 2012. Automatic annotation of face velocity and acceleration in Anvil. Proceedings of the Language Resources and Evaluation Conference (LREC-2012). Istanbul, Turkey.

Kristiina Jokinen and Silvi Tenjes. 2012. Investigating Engagement - intercultural and technological aspects of the collection, analysis, and use of the Estonian Multiparty Conversational video data. In Proceedings of the Eight International Conference on Language Resources and Evaluation (LREC'12), 23-25 May 2012, Istanbul, Turkey

Kristiina Jokinen and Graham Wilcock. 2014. Automatic and manual annotations in first encounter dialogues. Proceedings of the 6th International Conference: Human Language Technologies – The Baltic Perspective. Kaunas, Lithuania.

Costanza Navarretta, Elisabet Ahlsén, Jens Allwood, Kristiina Jokinen and Patrizia Paggio. 2012. Feedback in Nordic First-Encounters: a Comparative Study. Proceedings of the Language Resources and Evaluation Conference (LREC-2012). Istanbul, Turkey.

Michael Kipp. 2001. Anvil - A Generic Annotation Tool for Multimodal Dialogue. Proceedings of the 7th European Conference on Speech Communication and Technology (Eurospeech), pp. 1367-1370.

Alper Yilmaz, Omar Javed, and Mubarak Shah. 2006. Object tracking: A survey. ACM Computing Surveys. 38 (4), Article 13. Available at: http://crcv.ucf.edu/papers/Object%20Tracking.pdf