

# Subtitle-free Movie to Script Alignment

Pramod Sankar K.<sup>1</sup>

pramod\_sankar@research.iit.ac.in

C. V. Jawahar<sup>1</sup>

jawahar@iit.ac.in

Andrew Zisserman<sup>2</sup>

az@robots.ox.ac.uk

<sup>1</sup> Centre for Visual Information Technology

IIT, Hyderabad, INDIA

<http://cvit.iit.ac.in>

<sup>2</sup> Visual Geometry Group

University of Oxford, Oxford, UK

<http://www.robots.ox.ac.uk/~vgg>

---

## Abstract

A standard solution for aligning scripts to movies is to use dynamic time warping with the subtitles (Everingham *et al.*, BMVC 2006). We investigate the problem of aligning scripts to TV video/movies in cases where subtitles are *not* available, e.g. in the case of silent films or for film passages which are non-verbal.

To this end we identify a number of “modes of alignment” and train classifiers for each of these. The modes include visual features, such as locations and face recognition, and audio features such as speech. In each case the feature gives some alignment information, but is too noisy when used independently. We show that combining the different features into a single cost function and optimizing this using dynamic programming, leads to a performance superior to each of the individual features.

The method is assessed on episodes from the situation comedy *Seinfeld*, and on Charlie Chaplin and Indian movies.

## 1 Introduction

Scripts aligned with videos offer a number of possibilities: they can provide supervisory information for identifying characters [7] or learning actions [13]; they enable a scene level organization of the video material [6]; and they enable text-based retrieval and search [16]. The typical method of aligning a script (or transcript) with TV or movie videos is dynamic time warping with the subtitles/closed-captions, as introduced by Everingham *et al.* [7].

In this paper our objective is the visual alignment between TV/movie videos and their scripts, *without* using subtitles. Achieving such an alignment increases the scope and applicability of script-based approaches to videos with little or no spoken dialogue, and to situations where subtitles are unavailable, as is common in many non-European language films or silent films. Therefore, it considerably increases the extent of video material that is available for training visual classifiers, and is suitable for text based search. The challenge, however, is the comparison and matching of visual and audio information with the script descriptions of the video. The description may be quite generic (a street, an office) or quite specific (Jerry’s apartment), and may involve objects and actions for which visual recognition is not yet mature.

Our approach is to combine several cues, both visual and audio, which by themselves are not quite reliable, but when combined provide sufficiently strong constraints for a full alignment. We pose the problem as one of multi-state labelling of a sequence of shots, where the states for each shot correspond to the sentences of the script. In order to reduce the



**Figure 1:** Examples of shots correctly annotated by their descriptions. The first two shots are from the episode *The Contest*, the other two from *The Pick*. The annotations from scripts of Seinfeld episodes are aligned with the video, without using the timing information from subtitles, but by using clues from recognition alone.

ambiguity we explore segmenting the video into scenes associated with locations. Figure 1 shows an example of the resulting alignment.

## 2 Data and Performance Measure

We use episodes from the popular TV situation comedy *Seinfeld*, from Charlie Chaplin and Indian movies. Our dataset consists of episodes from Season 4 of *Seinfeld*: *The Pitch* and *The Ticket* (training data), *The Contest* and *The Pick* (test data). The Charlie Chaplin movies are excerpts from *The Gold Rush* and *City Lights*, and the Indian movie in consideration is *Agantuk*. The videos are divided into shots by computing the difference between colour histograms of consecutive frames. Whenever this difference is greater than a certain threshold, a shot-cut is detected. A typical 20 minute episode of *Seinfeld* has around 310 shots. Scripts for the Charlie Chaplin movies were obtained from [1], and for the *Seinfeld* shows from [4]. An excerpt from a script looks like:

```
[Setting: Monk's Coffee shop]
(Jerry and Kramer are sitting opposite Elaine at a booth, eating lunch)
JERRY: (To Elaine) Let me ask you a question.
ELAINE: Mm-hm.
JERRY: You're a hostage, captured by terrorists-
ELAINE: (Smiling, chewing) Who, me?
```

A typical script specifies the location of the scene (given as “Setting”), along with a brief description of the scene. The rest of the script has two distinct elements. The first is the detail about who is speaking and what, the other is a description of one or more of action and expressions of the characters.

**Performance measure:** The problem of alignment is now defined as, *given a video and the script for the events occurring in the video, assign each segment of text to the appropriate segment of the video*. The segment of the text is chosen to be a sentence, and that of the video to be a shot. The reason for this choice is that, in general, each sentence is “atomic” to a shot. Meaning, a sentence is generally spoken within one shot. The descriptions of actors’ action/expression are localised to a shot as well. We shall call the spoken sentences  $S_s, s \in [1, N_S]$ , and localised descriptions as  $D_d, d \in [1, N_D]$ , where  $N_S, N_D$  are respectively the number of such sentences. Thus, for each sentence in  $S_s \cup D_d$ , the alignment tries to identify the right shot  $T_t, t \in [1, N_T]$ . The performance of the alignment is evaluated against manually ground truthed sentence-shot correspondences (Figure 2), the accuracy given as the number of sentences from  $S \cup D$  assigned to the right shot. We shall denote by  $S', D'$ , the sentences

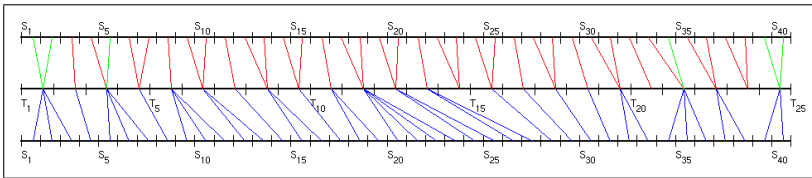


Figure 2: Example of a naive alignment scheme, shown over the first 40 sentences from the episode *The Contest*.  $S_s$  corresponds to sentences and  $T_t$  to shots. The blue lines correspond to the ground truth alignment. The green and red lines indicate the correct/incorrect correspondences from naive alignment. It can be seen that errors at the beginning drastically affect the rest of the alignment.

aligned with the shot  $t$ . The ground truth is represented as  $S_G^t, D_G^t$ . The performance measure is defined as

$$\rho = \frac{\sum_t |S^t \cap S_G^t| + \sum_t |D^t \cap D_G^t|}{N_S + N_D}$$

If every sentence is aligned correctly, the intersection between the alignment and the ground truth, summed over all shots yields a numerator equal to the total number of sentences. The value of  $\rho$  in this case would be one. Whenever the sentences are assigned to the wrong shot, the numerator decreases, and hence the  $\rho$ . We also define  $\rho_k$ , as the performance measure that allows errors in assignment upto  $k$  shots per sentence.

For a correct alignment, each sentence should compete for the right shot to fall into. The voting of a shot should depend on common features that can be extracted from both the sentence and the shot. For example, let us suppose that we know from the script that a sentence was spoken by Kramer while at the Monk’s Cafe. Such a sentence would more likely occur at a shot known to belong to that location, in which Kramer can be visually recognised. Additional clues from the speech domain, would provide further evidence in support of such an assignment. Towards this end, we extract three clues from each shot/sentence:  $\langle \text{Location}, \text{Face}, \text{Speech} \rangle$ . We apply state-of-the-art techniques for each of these modules, and the results on their performance are reported in the following section.

## 3 Recognizing Visual-Audio Aspects

### 3.1 Location Recognition

Television series are characterised by repetitive locations and recurring characters. In the case of *Seinfeld*, “Jerry’s Apartment” and “Monk’s Cafe” are the locations for a large number of the scenes. The setting of these locations remains relatively similar throughout the series, making them good candidates for scene recognition. In the case of sitcoms, each scene in the video is preceded by a *stock-shot* of the location from an exterior view. An example shown in Figure 3. The recognition of this stock-shot reliably identifies the starting shot for that location. The stock-shot varies in viewpoint, illumination and scale, across different occurrences. SIFT [14] features handle these variations well. We approach this problem as a near-duplicate image matching, given a set of stock-shot exemplars. Exemplars for stock-shot are identified from the training data. The SIFT features are vector-quantised into  $K$  visual words, and each shot is then represented by a bag of visual words (BoW) [17]. The BoW for each shot is compared by the  $L_1$ -Norm with the BoW representation of the exemplars for the stock-shots. If the closest exemplar is less than a small threshold, the shot is classified to the particular scene category. By this method, we are able to reliably identify the beginning of scenes whose location is either Jerry’s apartment or Monk’s Cafe.

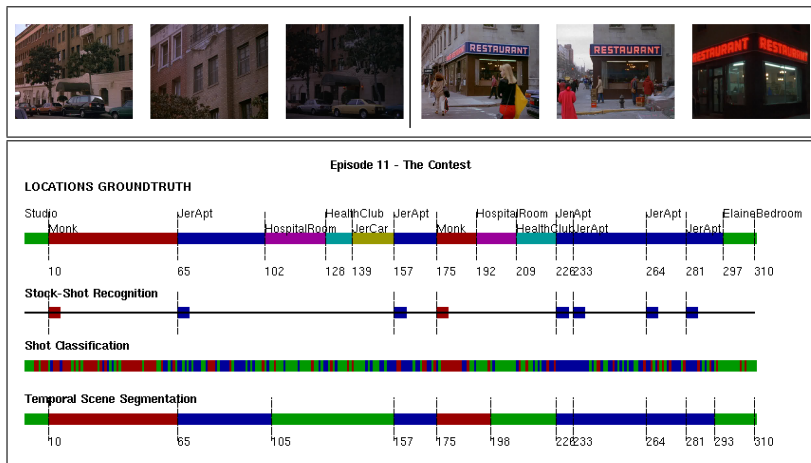


Figure 3: Location recognition pipeline. Stock shots (above) are recognised to identify the beginning of Jerry’s Apartment and Monk’s Cafe. Shots are then classified using an SVM into one of Jerry’s Apartment (blue), Monk’s Cafe (red) and Other (green). Temporal scene segmentation is performed by combining the stock-shot and classifier information.

Given the beginning of these scenes, the next step is to determine their extent. The goal here is to classify the subsequent shots as belonging to that location or not. This is a multi-class problem: the shot can belong to Jerry’s apartment or Monk’s cafe or ‘other’ (where ‘other’ covers all other locations). The classification proceeds in two steps: first individual shots are classified using a Kernel-SVM over BoW representation for shots; then a sliding window (on shots) is used to determine the scene boundary. The scene segmentation procedure is depicted in Figure 3.

In detail, I-frames are extracted from each shot, over which Hessian-Affine [15] and SIFT interest regions are obtained. These interest regions are represented using the SIFT descriptor, and a BoW representation obtained for each shot. A Kernel-SVM classifier is trained for each class, with the  $\chi^2$  kernel distance between two shots  $p, q$  given as

$$K(p, q) = e^{-\alpha \chi^2(p, q)} \quad \text{where,} \quad \chi^2(p, q) = \sum_{i=1}^N \frac{(p_i - q_i)^2}{p_i + q_i}$$

Here, the parameter  $\alpha$  is set to be the average  $\chi^2$  distance between training shots. The shot classification accuracy for the two classes with this method was about 67%.

To improve the location recognition accuracy, we experimented with excising humans before classifying the shots so that the matching can concentrate on the backgrounds. However, we found that this decreased the performance from 67% to 50%. To elaborate, we used the upper body detector provided by Ferrari *et al.* [10], to locate the humans and then masked them out using a matte constructed by averaging a number of upperbody training examples (i.e. the head and torso, not the entire ROI). All detected features within the mask were then filtered out, and the BoW computed from the remainder. The lack of success of this excising is possibly due to the camera focusing on the people, thereby defocusing the background; also, removing features over people still results in the loss of some features from the background.

**Temporal recognition refinement:** The location recognition accuracy is improved by using the property that the location does not change with every shot, but only across scenes. Errors in shot classifiers can be corrected with temporal filtering. The beginning of Jerry’s Apartment and Monk’s Cafe are obtained from the stock-shot recognition. We now need to identify where such scenes end and the ones from *Other* category begin. To identify the *scene-boundaries* between location  $L_l$  and  $L_{l+1}$ , a sliding window of  $W$  shots is moved across the video. At each shot  $s$ , an objective function is computed as

$$E_s = \alpha_1 \cdot \left\{ \sum_{i=s-W/2}^s (1 - P(L_l|i)) + \sum_{i=s}^{s+W/2} (1 - P(L_{l+1}|i)) \right\} + \alpha_2 \cdot \{Dist(s, s+1)\}, (\alpha_1 + \alpha_2 = 1)$$

The first term in  $E_s$  evaluates the cost of assigning the shots  $[s - W/2, s]$  to the location  $L_1$ , and those from  $[s, W]$  to  $L_2$ .  $P(L_l|i)$  is the score obtained from the shot classifier. The second term penalizes scene-boundaries at similar looking adjacent shots. The  $Dist(s, s+1)$  is obtained as the inverse of the L1-Norm difference between the shot’s BoWs. The scene-boundary  $B_l$  is obtained as the shot-boundary at which the objective function is minimum. From the training data, we estimate the best performing  $\alpha_1$  and  $\alpha_2$  to be 0.3 and 0.7 respectively, using a window size of eight shots.

The scene boundary between adjacent scenes, both belonging to *Other* is estimated from the number of sentences spoken in the given location. It is assumed that the scene duration is proportional to the length of the speech. This estimate is refined by assigning the scene-boundary to the closest shot with no faces or speech detected (see Sections 3.2 and 3.3). Such a shot would typically represent a change in the scene. The final location recognition accuracy is measured as the number of shots assigned to the correct location. The accuracy for the training and test data is 96%. An example result is shown in Figure 3.

## 3.2 Face Recognition

Seinfeld consists of four main characters, namely Jerry, George, Elaine and Kramer. By recognizing these characters, a large percentage of faces can be labeled in the video. We use the face detection/tracking/recognition pipeline of Everingham *et al.* [7]. We add to this a step of rejecting false face detections using skin pixel detection. The threshold for skin/non-skin classification is chosen such that 75% of the false detections are removed, while retaining about 95% of true detections. This removes typical false detections that occur over windows, TV channel logos, clothing etc.

In the pipeline facial feature points are extracted from the corners of the eyes, nose and mouth using the code provided by [7]. SIFT-like features are computed for each of 13 points, which are concatenated to form a single vector for each face image. Faces are classified against a set of hand picked exemplars for each character. We build a Kernel-SVM [9] for face recognition. Here, the kernel is defined as the min-min distance between a given face track and exemplar face tracks:

$$K(F_i, F_j) = \max_{p_i \in F_i, p_j \in F_j} s(p_i, p_j)$$

where  $s(p_i, p_j)$  is computed as a RBF Kernel on the distance between the facial feature vectors of  $p_i$  and  $p_j$ . We use a refuse-to-predict scheme of [7] where labels are given to face tracks only if we are confident about such an assignment. The precision-recall curve for the classifier is given in Figure 4 (left). Precision is the fraction of correctly labeled face tracks, and recall is fraction of the tracks whose label is predicted. Our classifier achieves a precision of 80% at 65% recall.

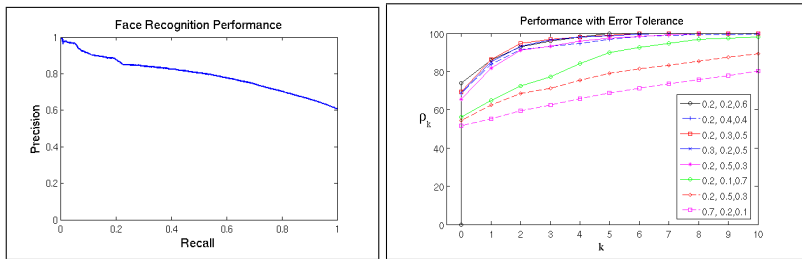


Figure 4: (left) Precision-Recall curve of face recognition. (right) Performance metric  $\rho_k$  across various values of  $k$ , as evaluated over the test episodes; the weights for the {location, face, speech} costs are specified in the legend.

### 3.3 Speech Recognition

The audio track provides useful clues for aligning the spoken sentences. We explore the use of speech recognition for our alignment procedure. The audio from each shot is isolated and provided as input to standard speech recognition packages, namely CMU-Sphinx [3] and Dragon Naturally Speaking (DNS) [2]. We do not perform either speaker or sentence segmentation of the audio speech. The in-built speech/speaker models were directly used, since training the speech models would require substantial training data for each speaker. Recognition output for an example shot is given below:

**Actual speech:** “No, ma, I’m not gonna see a psychiatrist. N- I don’t care if you do pay for it! No! Discussion over. Yeah, alright, I’ll see you later. Yes, of course I’m gonna come by. Alright. My mother wants me to see a psychiatrist now.”

**Recognised speech (CMU Sphinx):** “ooohh contest you psychiatrist now how difficult re horrible now shuttle door s elaine guess what sound that and i a hair and the walls visiting just now”

**Recognised speech (Naturally Speaking):** “home is an interest rate for no destruction of the IIRC it would – of course I’m going to combine for a little as we see a psychiatrist”

The word *psychiatrist* was correctly recognised once in both systems, even though it occurs twice in the conversation. Other recognised words were *see a, going* (DNS), *now* (Sphinx). Matches over stopwords such as {*a, the, and, I, it, ...*} are not considered. The recognition performance of speech recognition was understandably poor [5, 11], owing to the fact that we provide the software with “wild” data: the audio files contain multiple speakers in a single shot, laughter of the audience, varying speed of speech delivery, background music etc., which are not trained for in generic speech recognition systems. We get a word level recognition accuracy of 10%. The number of sentences in the training episodes with at least one word recognised correctly by DNS was 21%. The same for the test episodes was 23%.

## 4 Aligning Videos with Scripts

As was seen in the previous section, the visual-audio recognition modules are not accurate enough to align independently. However, additional characteristics of the problem can be exploited for aligning the script to the video. We formulate the problem as one of multi-state labelling of the shots, with the states of each shot corresponding to the sentences of the script. We will illustrate the formulation using the spoken sentences  $S$ , though a similar development can be given for the descriptions  $D$ . Let us denote by  $d(i, j)$ , the local cost of

the assignment  $S_i \Leftrightarrow T_j$ , and  $\mathcal{D}(i, j)$  the global cost of assignment. We have the following constraints in our formulation:

- **Uniqueness constraint:** Each sentence can be assigned to only one shot.
- **Ordering constraint:** The sentences and shots are ordered lists, hence implying a sequence constraint to the sentences and shots. Therefore, if  $S_i \Leftrightarrow T_j$ , indicates that the  $i^{\text{th}}$  sentence is assigned to the  $j^{\text{th}}$  shot, then  $\forall i' < i$  and  $S_{i'} \Leftrightarrow T_{j'} \Rightarrow j' \leq j$ .
- **Null-assignment:** It is possible that certain shots do not have any sentences associated with them. This could be because no character is speaking in the shot, or if it is a stock-shot. Hence, the predecessor of  $S_i \Leftrightarrow T_j$  in the alignment could be s.t.  $S_{i-1} \Leftrightarrow T_{j-k}, k \in [1, j-1]$ . A penalty term is associated with each jump over a shot. There are no null-assignments over sentences, i.e. every sentence is always assigned to a shot.
- **Multiple assignment:** Multiple (contiguous) sentences can be assigned to a single shot. However, their combined word count should fit during the shot duration. The local cost function is modified as  $d'(i, j) = d(i, j) + \gamma \cdot \text{dist}_{\text{Length}}(i, i-1, \dots, i-k, j)$ . We estimate the average number of words that could be spoken in a shot, based on its length. The  $\text{dist}_{\text{Length}}$  is the difference between the estimated word count and the number of words in the sentences  $[i-k, i]$  assigned to the shot  $j$ .

**Speech recognition costs.** Speech recognition results are filtered by word length, only words longer than four characters are considered. The similarity score is based upon the number of words overlapping between the speech recognition output and the given sentence. The maximum overlap was observed to be two. The speech based distance measure  $\text{Cost}_{\text{Speech}}$  is set to be 0 for two matching words, 0.5 for one match and 1 for no matches.

**Face recognition costs.** Three statistics over the training data are used in constructing the face recognition costs: (i) the probability of the speaker being visible in the shot is 0.94; (ii) the probability that a non-speaker is present in the shot is 0.36; and (iii) the probability that the speaker is visible, but not detected in the shot is 0.07. Accordingly, the  $\text{Cost}_{\text{Face}}$  is defined as  $(1 - 0.94) * (1 - \text{classifier\_score})$ , if the speaker of a sentence is visible in the given shot. In case the speaker is not visible,  $\text{Cost}_{\text{Face}} = (1 - 0.07) * \text{avg}_C(\text{classifier\_score})$ ,  $C$  is the character in consideration. For each non-speaker recognised in the shot, the face cost is *incremented* by 0.36 times the average classifier scores for those face tracks.

**Location recognition costs.** The location cost depends on the location for the sentence. Since the accuracy of scene segmentation of Jerry’s Apartment or Monk’s Cafe is 96%, the cost of a sentence being assigned to a shot recognised to be in these locations is set to be 0.04, and 0.96 otherwise.

The final local cost  $d(i, j)$  is computed from a weighted combination of the location, faces and speech distances:

$$d(i, j) = \alpha_1 \cdot \text{Cost}_{\text{Location}}(i, j) + \alpha_2 \cdot \text{Cost}_{\text{Face}}(i, j) + \alpha_3 \cdot \text{Cost}_{\text{Speech}}(i, j), \alpha_1 + \alpha_2 + \alpha_3 = 1$$

This formulation lends itself to be solved using dynamic programming. Apart from the global cost array  $\mathcal{D}$ , an indicator array  $I$  is maintained, where  $I(i, j)$  points to the table entry corresponding to the optimum subproblem solution of  $\mathcal{D}(i, j)$ . By backtracking  $I$ , we recover the alignment between the sentences and shots. The complexity of our algorithm is of the order  $O((N_S + N_D) \cdot N_T)$ .

## 5 Results

**Baseline alignment.** A blind alignment scheme would uniformly spread the sentences across the shots. Given  $N_S + N_D$  sentences and  $N_T$  shots, such a scheme would allot  $\lfloor (N_S + N_D)/N_T \rfloor$  sentences to each shot, sequentially. An example alignment using this scheme is shown in Figure 2. This alignment performs poorly, since errors once committed are hard to recover from, and such errors drastically effect the subsequent alignment. The performance of this method,  $\rho$ , is only about 4%. From the groundtruth sentence-shot correspondences (shown as blue lines), it can be seen that the uniform speech assumption is invalid. A stronger hypothesis is required to assign a particular sentence to a given shot.

**Subtitle alignment.** Subtitle based alignment uses the timing information in subtitles to assign the sentences to shots. However, the subtitles are designed such that they cover multiple sentences, displayed over multiple shots. There are many instances where a single shot would mark the end of a subtitle and the begin of the next. Such sentences are spread across multiple shots using the naive alignment scheme. The  $\rho$  of this scheme was 91%.

**Global alignment.** Aligning the script with video can be performed using different combinations of the modalities. In the first instance of using speech alone, the sparse sentence-shot correspondences obtained from speech are used to drive the dynamic programming. With ideal speech recognition, this method would replicate subtitle-based alignment. However, given the insufficient speech matches for a bulk of shots in some of the scenes, the  $\rho$  of this modality is about 47% on the training data. On the other hand, using the face recognition results alone gives a  $\rho$  of 33%. Since we only recognise the faces (we do not perform speaker detection), the sentences are matched across shots where the character is present but is not speaking. Further, the errors in face recognition deteriorate the alignment when the speaking character is not correctly recognised in the shot. These errors confuse the alignment by providing false matches between sentence and shots.

The weights  $\alpha$  for each modality are learnt using the training data. The weights for each modality guide the alignment in cases where the clues do not agree with each other. If for example, the location and speech clues conflict, the respective  $\alpha$  determines which modality takes precedence. Thus, with a higher weight for speech, errors due to location could be overcome and vice-versa. The best performing parameters were found to be  $\{0.2, 0.2, 0.6\}$ , at which the  $\rho$  over training data was 71%, and over the test episodes was 67%.

**Scene level alignment.** Using the location recognition of Section 3.1, we restrict assigning the sentences within a scene to the shots of a location video segment. The alignment within scenes is carried out using the face and speech information. This procedure essentially anchors the sentences and shots known to belong together, and identifies an alignment between such anchors. With this procedure, we are able to improve results and we obtain a  $\rho$  of 74%.

Example results from annotation are shown in Figure 1. We have analysed the errors in the episode *The Contest*. The two main reasons for the alignment errors are mistakes in the scene segmentation, and the lack of sufficient clues from speech to correct the location based errors. Of the 15 scenes in the episode, 5 scenes have poor temporal segmentation and sparse speech-based matches. These scenes account for about 71 erroneous sentence assignments. The remaining 68 errors are distributed across the other 10 scenes.

In most error cases, the sentence is assigned within a few shots of the actual correspondence. Over our test data, the maximum distance of an erroneous assignment was of five shots, hence  $\rho_5 = 100\%$ . This can be seen from the graph in Figure 4 (right). In a video retrieval scenario, for a given textual query, we could provide a video segment consisting of



multiple shots. By returning video segments consisting of about 11 shots, the precision of retrieval would be 1, even though the precise alignment might be less accurate.

## 5.1 Aligning Scripts of Silent Movies and Indian Films

We apply the various cues discussed above, to align the silent movies of Charlie Chaplin with their scripts. We apply our techniques to two scenes, one from *The Gold Rush* and the other from *City Lights*. In Seinfeld videos, we have used the stock-shot information for scene segmentation. In the case of Chaplin movies, scene changes are observed as a visual fading to a blank screen. By detecting the blank screen, which can be reliably performed, we find the scene boundaries. The scene segment of the text is identified from the mention of *Curtain*. For example, the scene changes whenever the script indicates similar to “Curtain lowered to denote lapse of time”

For the video clip from Gold Rush, the script only describes scenes occurring in the setting of *The Cabin*. To detect this scene, we use a Kernel-SVM classifier over BoW representation, similar to the one used in location recognition of Seinfeld. Shots are classified as belonging to the Cabin, the scene with the most non-cabin shots was classified as the un-scripted scene. Following this, the scenes in the text are correctly aligned with the corresponding video segment. Within a scene segment, the alignment of descriptions is carried out using face recognition, with the method of Section 3.2 to label the characters Lonely, Old Timer, Black Larsen. Sample annotations after alignment of the script with the video, are shown in Figure 5 (left). The clip from City Lights has *Intertitles*, where the dialogue is written on the video frame itself. These intertitles were detected and fed to a commercial OCR. The recognised text provides us with additional constraints for the alignment. Resulting annotations are shown in Figure 5 (right).







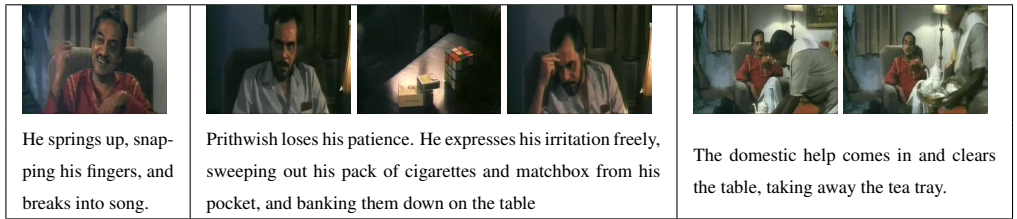
	Lonely, is seen at stove stirring boiling pot. Old Timer is leaning over table groaning.		Lonely pokes fork in pot as if testing tenderness of what he is cooking		Tramp stands up and accepts a glass from Millionaire
	Old Timer grimaces as he bites into piece of upper.		*Here's to our friendship —		He pours glasses full. They both hold glasses above heads in toast fashion.

Figure 5: Examples of annotated scenes from (left) *Gold Rush*, (right) *City Lights*. The second shot in the City Lights example is the Intertitle; the OCR output for this is shown alongside.

We further test the applicability of our approach in aligning scripts of films with no subtitles, a common situation for Indian films. For this example, we choose a segment from the movie *Agantuk*, the script for which was available in published form. We use the face and speech modalities for the matching sentences and shots. Though much of the dialogue is in the language of Bengali, it is interspersed with English words. By applying speech recognition using the DNS engine, we could obtain about nine strong matches between the recognised speech and the script. Using the speech and face clues, we achieved satisfactory alignment results for this data. Some examples are shown in Figure 6.



**Figure 6:** Examples of annotated scenes from the Bengali movie *Agantuk*. The movie DVD does not provide subtitles, in spite of which we were able to align the video with the dialogues and descriptions provided in the script.

## 6 Conclusions

We have presented an approach to enable visual alignment of movies with their scripts, without using subtitles. Different features were recognised for this purpose, to provide clues of the alignment. Though individual recognition modules performed poorly, their combination in a multi-modal framework yielded satisfactory alignment. The use of these alignment results could enable text-based search over movie video collections, and to create training data for various vision tasks.

We have presented a framework that can be extended as visual object recognition improves [8] or recognition of actions/interactions improves [12, 13], so that more correspondences can be found between nouns/verbs in the script and visual detections. Similarly, other audio cues such as silence, speaker recognition etc. could be included.

**Acknowledgements.** We are grateful for financial support from the UKIERI, EU Project CLASS and ERC grant VisRec no. 228180, and for discussions with James Philbin.

## References

- [1] Charlie chaplin scripts from: <http://www.charliechaplinarchive.org/>.
- [2] Dragon naturally speaking software from: <http://www.nuance.com/naturallyspeaking/>.
- [3] CMU sphinx from: <http://cmusphinx.sourceforge.net/html/cmusphinx.php>.
- [4] Seinfeld scripts from: <http://www.seinfeldscripts.com>.
- [5] M. Chen and A. G. Hauptmann. Multi-modal classification in digital news libraries. In *JCDL*, pages 212–213, 2004.
- [6] T. Cour, C. Jordan, E. Miltsakaki, and B. Taskar. Movie/script: Alignment and parsing of video and text transcription. In *Proc. ECCV*, 2008.
- [7] M. Everingham, J. Sivic, and A. Zisserman. “Hello! My name is... Buffy” – automatic naming of characters in TV video. In *Proc. BMVC*, 2006.
- [8] M. Everingham, Luc V. Gool, C. Williams, J. Winn, and A. Zisserman. The PASCAL Visual Object Classes Challenge 2007 (VOC2007) Results. 2007.

- 
- [9] M. Everingham, J. Sivic, and A. Zisserman. Taking the bite out of automatic naming of characters in TV video. *IVC*, 27(5), 2009.
  - [10] V. Ferrari, M. Marin-Jimenez, and A. Zisserman. Progressive search space reduction for human pose estimation. In *Proc. CVPR*, 2008.
  - [11] M. Huijbregts, R. Ordelman, and F. de Jong. Annotation of heterogeneous multimedia content using automatic speech recognition. In *Proc. SAMT*, 2007.
  - [12] I. Laptev and P. Perez. Retrieving actions in movies. In *Proc. ICCV*, 2007.
  - [13] I. Laptev, M. Marszalek, C. Schmid, and B. Rozenfeld. Learning realistic human actions from movies. In *Proc. CVPR*, 2008.
  - [14] D. G. Lowe. Distinctive image features from scale-invariant keypoints. *IJCV*, 60(2), 2004.
  - [15] K. Mikolajczyk, T. Tuytelaars, C. Schmid, A. Zisserman, J. Matas, F. Schaffalitzky, T. Kadir, and Luc V. Gool. A comparison of affine region detectors. *IJCV*, 65(1/2): 43–72, 2005.
  - [16] K. Pramod Sankar, S. Pandey, and C. V. Jawahar. Text driven temporal segmentation of cricket videos. In *Proc. ICVGIP*, 2006.