FLAD: A Human-centered Video Content Flaw Detection System for Meeting Recordings

Haihan Duan
The Chinese University of Hong Kong, Shenzhen
Shenzhen, China
haihanduan@link.cuhk.edu.cn

Junhua Liao
Sichuan University
Chengdu, China
liaojunhua@stu.scu.edu.cn

Lehao Lin
The Chinese University of Hong Kong, Shenzhen
Shenzhen, China
lehaolin@link.cuhk.edu.cn

Wei Cai*
The Chinese University of Hong Kong, Shenzhen
Shenzhen, China
caiwei@cuhk.edu.cn

ABSTRACT

Widely adopted digital cameras and smartphones have generated a large number of videos, which have brought a tremendous workload to video editors. Recently, a variety of automatic/semi-automatic video editing methods have been proposed to tackle this issue in some specific areas. However, for the production of meeting recordings, the existing studies highly depend on additional conditions of conference venues, like infrared camera or special microphone, which are not practical. Moreover, current video quality assessment works mainly focus on the quality loss after compression or encoding rather than the human-centered video content flaws. In this paper, we design and implement FLAD, a human-centered video content flaw detection system for meeting recordings, which could build a bridge between subjective sense and objective measures from a human-centered perspective. The experimental results illustrate the proposed algorithms could achieve the state-of-the-art video content flaw detection performance for meeting recordings.

CCS CONCEPTS

• **Human-centered computing** → *Visual analytics*; • **Computing methodologies** → **Visual content-based indexing and retrieval**.

KEYWORDS

Video Editing; Meeting Recordings; Human-centered Computing; Video Quality Assessment

ACM Reference Format:

Haihan Duan, Junhua Liao, Lehao Lin, and Wei Cai. 2022. FLAD: A Human-centered Video Content Flaw Detection System for Meeting Recordings. In *32nd edition of the Workshop on Network and Operating System Support for Digital Audio and Video (NOSSDAV '22), June 17, 2022, Athlone, Ireland*. ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3534088.3534349

1 INTRODUCTION

With the popularity of digital cameras and smartphones, the generation of videos becomes increasingly convenient with lower cost. At the same time, the tremendous amount of videos also brings a heavy workload for professional video editors because of the post-production of video materials. The development of multimedia and computer vision provides promising technologies to relieve the burden of professional video editors. In recent years, some fully automatic video editing methods have achieved surprising performance in specific areas, e.g. multiparty conversation [34], social gatherings [46], school concerts [15], instructional videos for physical demonstrations [4], dialogue-driven scenes [16], dance videos [39], and social cameras (cameras that are carried or worn by people in activities) [2]. Otherwise, the participation of video editors seems inevitable in other scenes, so semi-automatic or assistant approaches are proposed, such as semantic zooming [21], narrated videos [38], home video [6], and video montage from the text[40].

The aforementioned automatic/semi-automatic video editing systems can effectively finish their task in specific areas, except meeting recordings. In our daily life, various meetings are held around the world every day, including academic conferences, sports press conferences, enterprise annual conferences, and so on. The meeting recording is relatively special in the various categories of videos. From the perspective of quality requirement, an effective meeting video must satisfy three criteria [29]: (1) It must capture enough visual information to allow viewers to understand what took place; (2) It must be compelling to watch; (3) It must not require substantial human effort. According to the above principle, the final output film of meeting recording only needs to capture sufficient information for inferring the events while fewer after-effects are required, but the tediously long videos bring a huge workload for professional video editors. Specifically, our interview with a video editor mentioned that the most time-consuming procedure is flaw detection from video materials, while the editing principle for meeting recordings is relatively simple that novices could handle.

However, the existing approaches cannot provide sufficient support for novices in editing meeting recordings. On the one hand, the automatic/semi-automatic studies [17, 19, 29] aiming at the generation of meeting recordings have limited practical scenarios. On the other hand, most existing video quality assessment works mainly focus on the objective quality loss after compression, encoding or transmission [24, 30, 32, 44], while few pay attention to the subjective sense of viewers caused by the content of videos. Therefore, it is imperative to develop a system from a human-centered perspective to help novices in editing meeting recordings.

In this paper, we implement a human-centered video content flaw detection system to detect flaws in meeting recordings, named FLAD. As shown in Figure 1, the proposed FLAD system pays attention to the three most common video content flaws mentioned

*Wei Cai is the corresponding author. caiwei@cuhk.edu.cn

NOSSDAV '22, June 17, 2022, Athlone, Ireland

ACM ISBN 978-1-4503-9383-6/22/06…$15.00
https://doi.org/10.1145/3534088.3534349

Figure 1: Illustration of Three Common Video Content Flaws

by professional video editors: **(1) blurriness:** the focus point of the camera usually changes from one speaker to another, causing blurriness in the video during the procedure; **(2) jitter:** the camera usually needs to be moved by the photographer, which might remain jitters in the video; **(3) occlusion:** the cameras are inevitably occluded by some objects or persons that pass through the cameras. The major contributions of this paper can be concluded as follows,

- We propose three video flaw detection algorithms to detect blurriness, jitter, and occlusion respectively from the perspective of human-centered computing.
- We build a testbed to evaluate the proposed algorithms. The experimental results demonstrate that the three proposed algorithms could achieve state-of-the-art (SOTA) performance.
- We integrate the three proposed flaw detection algorithms to build a human-centered flaw detection system, named FLAD, which could effectively visualize the detected flaws.

2 RELATED WORK

Automatic/semi-automatic video editing systems. For automatic video editing of meeting recordings, Lefevre et al. [17] proposed an automatic video stream selection method based on the detection of the light state change of the microphones to track the speaker. Intuitively, this method cannot work if the used microphones do not have an indicator light. Ranjan et al. [29] implemented an automated meeting capture system to capture videos of small group meetings. This system introduced a lot of sensors such as infrared cameras and the Vicon motion tracking system[1], which is hard to be deployed in normal conference venues. Liu et al. [19] settled three cameras in the lecture room and proposed a virtual video director based on a finite state machine (FSM) to manage the cameras, which also depends on the pre-defined layout and principles. Therefore, the existing automatic/semi-automatic methods are not mature enough to deal with practical applications.

Video quality assessment approaches. In recent years, lots of researchers have contributed works about video quality assessment. For example, Staelens et al. [32] provided information on how to model and measure the perceived video quality of end-users by leveraging fundamental and pure network measurements. Rassool et al. [30] proposed Video Multi-method Assessment Fusion (VMAF) which estimates the perceived quality by computing scores from multiple quality assessment algorithms and fusing them using a support vector machine (SVM). Min et al. [24] conduct a subjective study of audio and video (A/V) quality and validated and tested objective A/V quality prediction models on their proposed database. However, the existing methods mainly focus on the video quality after compression, encoding, or transmission, while do not consider the quality of content from a human-centered perspective.

Video content flaw detection algorithms. (1) Blurriness detection: The existing methods pay attention to the blur degree estimation of videos after encoding, but they further focus on the global quality rather than extracting specific frames of the videos [3, 28]. (2) Jitter detection: Currently, there are some automatic/semi-automatic video editing systems that integrate the jitter detection algorithms [38, 40], while other jitter detection algorithms mainly fall into some specific areas, such as videos from the satellite [20, 35, 41]. (3) Occlusion detection: The occlusion detection algorithms play an important role in many computer vision applications, e.g. video tracking [8, 13, 14, 43], optical flow estimation [11, 12, 33, 42], and pedestrian detection [23, 25, 26, 45]. The most relevant study is proposed by Liao et al. [18], which builds a large-scale database for occlusion detection and proposes a benchmark that will be applied to evaluate the performance of our proposed model.

3 SYSTEM DESIGN

The workflow of the proposed FLAD system is shown in Figure 2, in which a sliding window would extract 8 frames and send them to the FLAD to three common flaws in real shooting scenes (blurriness, jitter, and occlusion) and then generate a visualization report for editors. The following subsections will describe each module in detail. Note that, we use $V = \{f_1, f_2, ..., f_n\}$ to denote an input video with totally n frames, and f_i represents the ith frame of the video.

3.1 Blurriness Detection

The representation of out-of-focus is blurriness in videos, and a typical feature is that there are little edges in these frames, which inspires many significant methods [1, 28]. However, the existing blurriness detection method only sets a constant value as the threshold, which cannot perform well for videos that are recorded by different shooting equipment. For example, all frames of the video

Algorithm 1: Blurriness Detection Algorithm

Input: Video $V = \{f_1, f_2, ..., f_n\}$, parameter θ, k
Output: Frames with blurriness B
Init: *Detected blurriness frame array B, Laplacian map array L, variance array of Laplacian map v*
for $i = 1$ *to* $i = n$ **do**
  L_i = Laplacian_map(f_i);
  v_i = Variance(L_i);
end
$SD(v)$ = Standard_deviation(v);
if $SD(v) > \theta$ **then**
  t = mean(v) - (mean(v) - min(v)) / k;
  for $i = 1$ *to* $i = n$ **do**
    if $v_i < t$ **then**
      B.append(f_i);
    end
  end
end
return B;

[1] https://www.vicon.com/

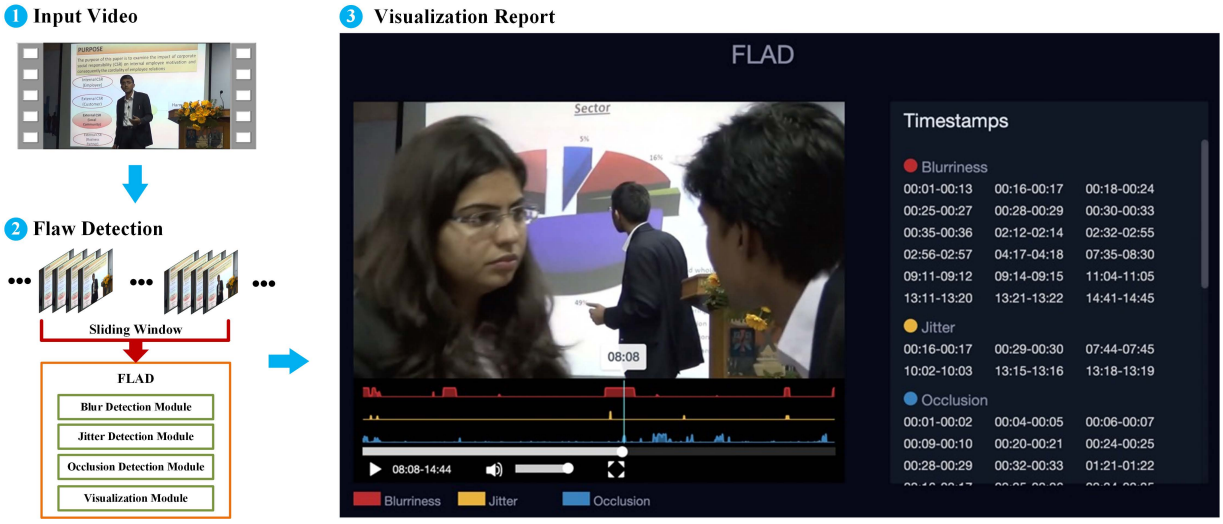

Figure 2: The Flowchart of FLAD System

with poor quality shooting equipment might be misrecognized by the method with an unsuitable constant threshold.

In the FLAD system, we modified a simple but sound blurriness detection method based on the Laplacian operator [28], as shown in Algorithm 1. We first calculate the second derivative of an image to represent the number of its edges, so the Laplacian operator is applied on each frame f_i and outputs their Laplacian map as $L = \{L_1, L_2, ..., L_n\}$. Then we calculate the variance of each Laplacian map as $v = \{v_1, v_2, ..., v_n\}$, where v_i could represent the number of edges in frame f_i. Intuitively, if the frame f_i is a blur frame, the v_i would get a lower value. Then we will calculate the standard deviation of the v, represented by $SD(v)$ to evaluate the average degree of blurriness. If the standard deviation $SD(v)$ is larger than $\theta = 1000$, we consider the video has an unbalanced clarity and then we will search out the blurriness. The frame would be regarded as a blur frame if its number of edges is lower than a dynamic threshold: $v_i < \overline{v} - (\overline{v} - v_{min})/k,\ i = 1, ..., n$, where $\overline{v}$ denotes the mean of variance array v, v_{min} is the minimum of v, and k is a coefficient and set as 3. The motivation of this algorithm is to utilize the global degree of blurriness for judgment instead of a fixed threshold.

3.2 Jitter Detection

In complicated shooting environment, the camera might be shaken and remain jitters in final recordings. However, the jitter is hard to be defined since there are many normal camera moves that might confuse the detection algorithms. Our preliminary experiments show that the existing jitter detection methods [38, 40] cannot well solve the jitter detection since they would falsely regard all camera moves as jitters. In this paper, we design a novel jitter detection algorithm based on an intuitive observation that normal consecutive frames should not move toward different directions in a short time slot (e.g., left then right). Algorithm 2 shows the pseudocode.

As shown in the flowchart of Figure 2, the sliding window would extract frames and send them for detection. Our algorithm first calculates the move direction of extracted frames. So the homography transformation matrix is calculated using SIFT features[22] and RANSAC regression[5] for consecutive frames as H. Each video frame is split into 4 equal parts from the middle, and the 4 pivots of which are selected as anchor points P. The 4 anchor points are used to measure whether the camera is moving. For example, if only 1/4 of anchor points have a shift, it might be a person who is walking while the camera does not move. Using the matrix H, we can calculate the mapping of anchor points as P' to estimate the Manhattan distance between P and P'. If more than 1 anchor points have a shift distance larger than a predefined distance dis (set as

Algorithm 2: Jitter Detection Algorithm

Input: Frames $f = \{f_k, f_{k+1}, ..., f_{k+7}\}$
Output: Whether the input frames f are jitters
Init: *Direction array* $D[7]$, *anchor points* $P[4]$,$P'[4]$
Resize all frames of f to (480×270);
for $i = 0$ *to* $i = 7$ **do**
  Calculate homography transformation matrix H using SIFT features of f_{k+i} and f_{k+i+1};
  Calculate the mapping P' of P using H;
  if *More than 1/4* $Manhattan\_distance(P, P') > dis$ **then**
    Calculate the moving direction as D_{k+i};
  end
  else
    $D_{k+i} = 0$;
  end
end
for $i = 0$ *to* $i = 7$ **do**
  if $D_{k+i} \neq D_{k+i+1}$ *and* $D_{k+i}, D_{k+i+1} \neq 0$ **then**
    return True;
  end
end
return False;

0.7), we believe the camera is moving. We define 4 directions (up, down, left, right) in axis directions, then we calculate the moving direction D_{k+i} if the 4 anchor points have the same moving direction, otherwise, we set $D_{k+i} = 0$. Then the algorithm will search for jitters from the array D. If 2 consecutive frames move toward different directions, we believe the input frames have jitters.

3.3 Occlusion Detection

The occlusion detection is relatively complex since the definition of occlusion does not have a consensus in different research areas. In video editing of meeting recordings, we regard the objects that appear at the improper time as occlusions, e.g., the objects that occlude the speakers. This task is highly suitable for the deep learning-based algorithm because it requires a semantic understanding of content.

In the FLAD system, we design a deep neural network model as shown in Table 1, which is a binary classification network (to classify whether the frame has occlusion). In this table, except for the special notes, all layers are convolutional layers with zero padding. For the input layer, 8 consecutive frames from the sliding window are resized as $8 \times 171 \times 128 \times 3$ as the input tensor. In each convolutional layer block, there are two branches of the network which divide the 3D convolution as a 1D+2D paradigm, and the features would be added before the max-pooling layer to combine the information. The loss function, which is demonstrated to be effective in occlusion detection [18], is applied in the training process.

Table 1: Occlusion Detection Neural Network Model

Stage	Parameters	
Conv1	$32, 3 \times 3 \times 3, (1, 1, 1)$	
	Max Pool 3D $1 \times 2 \times 2$ with Stride (1,2,2)	
Conv2	$32, 1 \times 1 \times 1, (0, 0, 0)$	$32, 1 \times 1 \times 1, (0, 0, 0)$
	$64, 1 \times 3 \times 3, (0, 1, 1)$	$64, 3 \times 1 \times 1, (1, 0, 0)$
	$64, 1 \times 1 \times 1, (0, 0, 0)$	$64, 1 \times 1 \times 1, (0, 0, 0)$
	Max Pool 3D $1 \times 2 \times 2$ with Stride (1,2,2)	
Conv3	$64, 1 \times 1 \times 1, (0, 0, 0)$	$64, 1 \times 1 \times 1, (0, 0, 0)$
	$128, 1 \times 3 \times 3, (0, 1, 1)$	$128, 3 \times 1 \times 1, (1, 0, 0)$
	$128, 1 \times 1 \times 1, (0, 0, 0)$	$128, 1 \times 1 \times 1, (0, 0, 0)$
	Max Pool 3D $1 \times 2 \times 2$ with Stride (1,2,2)	
Conv4	$128, 1 \times 1 \times 1, (0, 0, 0)$	$128, 1 \times 1 \times 1, (0, 0, 0)$
	$256, 1 \times 3 \times 3, (0, 1, 1)$	$256, 3 \times 1 \times 1, (1, 0, 0)$
	$256, 1 \times 1 \times 1, (0, 0, 0)$	$256, 1 \times 1 \times 1, (0, 0, 0)$
	Max Pool 3D $1 \times 2 \times 2$ with Stride (1,2,2)	
Conv5	$256, 1 \times 1 \times 1, (0, 0, 0)$	$256, 1 \times 1 \times 1, (0, 0, 0)$
	$512, 1 \times 3 \times 3, (0, 1, 1)$	$512, 3 \times 1 \times 1, (1, 0, 0)$
	$512, 1 \times 1 \times 1, (0, 0, 0)$	$512, 1 \times 1 \times 1, (0, 0, 0)$
	Max Pool 3D $1 \times 2 \times 2$ with Stride (1,2,2)	
Conv6	$512, 1 \times 1 \times 1, (0, 0, 0)$	$512, 1 \times 1 \times 1, (0, 0, 0)$
	$1024, 1 \times 3 \times 3, (0, 1, 1)$	$1024, 3 \times 1 \times 1, (1, 0, 0)$
	$1024, 1 \times 1 \times 1, (0, 0, 0)$	$1024, 1 \times 1 \times 1, (0, 0, 0)$
	Max Pool 3D $1 \times 2 \times 2$ with Stride (1,2,2)	
Conv7	$1024, 1 \times 1 \times 1, (0, 0, 0)$	$1024, 1 \times 1 \times 1, (0, 0, 0)$
	$2048, 1 \times 3 \times 3, (0, 1, 1)$	$2048, 3 \times 1 \times 1, (1, 0, 0)$
	$2048, 1 \times 1 \times 1, (0, 0, 0)$	$2048, 1 \times 1 \times 1, (0, 0, 0)$
	Global Max Pool 2D	
FC	Fully Connected 8×2048	
	Fully Connected 8×1024, Dropout 0.5	
	Fully Connected 8×512, Dropout 0.5	
	Fully Connected 8×2 and Softmax	

3.4 Detection Results Visualization

After the flaw detection, a Web-based report with an intuitive interface would be generated to visualize the detection results, as shown in Figure 2. The report provides three timelines with different colors to display detected flaws, and the fluctuations of each timeline straightly present the confidence of flaws in each second. For example, in the screenshot of Figure 2, the dialogue of the front two persons occluded the major speaker, so the series of frames is detected as occlusion and there is a ridge in the blue timeline. At the same time, since the camera is focusing on the major speaker, the front two persons show an obvious blurriness, which was also accurately detected as shown in the red timeline. On the other side, the panel "Timestamps" allows video editors to jump and check the specific duration. With this module, the FLAD could efficiently help video editors in flaw detection for meeting recordings.

4 EXPERIMENTS

4.1 Testbed

For the evaluation of the proposed algorithms, we totally collect 8 meeting recordings with different scenarios and quality (resolution of 1280×720) from YouTube. Some sample frames of the testbed are shown in Figure 3. Then we invited a professional video editor with 8 years of experience to carefully check the video materials. All the content flaws (blurriness, jitter, and occlusion) of the collected videos were annotated by the professional video editor second by second, which could be regarded as the baseline for

Figure 3: Sample Frames of Testbed

Table 2: Details of Video Content Flaw Testbed

Video No.	Scenario	Length	Blu.	Jit.	Occ.
Video 1	Academic Report	14:44	4	4	12
Video 2	Tutorial	08:54	2	6	4
Video 3	Project Meeting	09:01	0	1	7
Video 4	Award Ceremony	14:44	1	8	0
Video 5	Annual Meeting	13:36	0	2	3
Video 6	Academic Report	05:33	0	1	2
Video 7	Academic Conference	13:37	0	7	8
Video 8	Press Conference	05:13	0	6	0

further experiments. The detailed information about the dataset could be found in Table 2, in which we present the numbers of blurriness (Blu.), jitter (Jit.), and occlusion (Occ.) annotated by the professional video editor. For comparison, we also conduct a user study, in which totally 7 novices who are not familiar with video editing are recruited to point out the video content flaws second by second where they fill discomfort. This testbed will be open-sourced at https://www.kaggle.com/datasets/seaxiaod/flad-video-flaw-detection to support the related studies.

4.2 Implementation Details

The FLAD system is deployed on a server with 6 processors (Intel Core i7-7700 @ 3.60GHz) and 16GB RAM, which is equipped with an NVIDIA GTX 1080Ti GPU (11GB). The deep neural network model of occlusion detection is implemented using PyTorch [27]. A large dataset for occlusion detection [18] is utilized in the training of the neural network model, which contains 1,000 video segments where the appeared occlusions are annotated frame by frame. In the training process, the Stochastic Gradient Descent (SGD) is applied with 0.9 momentum and 0.0005 weight decay. The learning rate is initialized as 0.0001 with a learning rate decay of 0.5 every 10 epochs. And the degree of penalty λ in the loss function is set as 10. The whole training process contains 50 epochs.

4.3 Human-centered Evaluation Metrics

For the evaluation, we first evaluate how many annotated flaws are detected, denoted by *Sensitivity*. The classic action detection or recognition tasks apply frame-level evaluation, but, from a human-centered perspective, the event-level evaluation is more reasonable since the human sense cannot be accurate to frame-level detection. Therefore, we consider the flaw is detected if there is an overlap between the annotation from the professional video editor and the experimental detection result from algorithms or novices. On the other hand, the higher *Sensitivity* usually brings higher false positive results, so we use the duration of misrecognized flaws detected by each algorithm and novices to divide the total length of the video as the false positive ratio, represented as *FPR*. The *FPR* could be regarded as the additional workload that the users of the FLAD system need to check whether there is a flaw.

5 RESULTS

5.1 Comparison with Existing Algorithms

5.1.1 Blurriness Detection. Existing blurriness detection methods [1, 28] would set a fixed threshold, while the proposed algorithm can dynamically adjust the threshold using the global information, so the evaluation mainly focuses on its effectiveness. We implemented the most classic blurriness detection method based on the Laplacian operator as the baseline [28], in which we change its predefined threshold from 500 to 3500 with an interval of 500. Note that more clear images will show a higher variance of the Laplacian map, so the higher threshold is more sensitive to blurriness. The comparative results are illustrated in Figure 4. As shown in this figure, if the threshold is set as 500, the baseline method cannot detect any blurriness. With the growth of the threshold, the *Sensitivity* shows a significant increase, while the *FPR* also grows fast accordingly. Specifically, when the threshold is set as 2500, the baseline method and the FLAD system share the same *Sensitivity* which is larger than 0.9, but the *FPR* of the FLAD system is obviously lower than the baseline method. The experimental results demonstrate that the proposed blurriness detection algorithm can achieve better performance in detection and effectively maintain a low *FPR*.

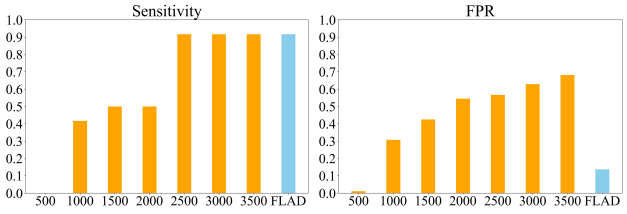

Figure 4: Comparison of Blurriness Detection Algorithms

5.1.2 Jitter Detection. To evaluate the performance of the proposed jitter detection algorithm, we introduce two SOTA jitter detection methods for comparison, including QuickCut [38] and Write-A-Video [40], which mainly evaluate the acceleration of video content to determine whether the camera is shaking. The comparison results of different jitter detection algorithms are shown in Figure 5. We can find that although the two comparative methods may detect more jitters in some specific videos, the proposed FLAD system can reach a higher average *Sensitivity*. It is worth noting that, for video 3, there is a jitter annotated by the professional video editor, but none of the three algorithms could search out it. On the other hand, the FLAD system also has better control over the *FPR*. Therefore, the proposed jitter detection method could achieve SOTA performance.

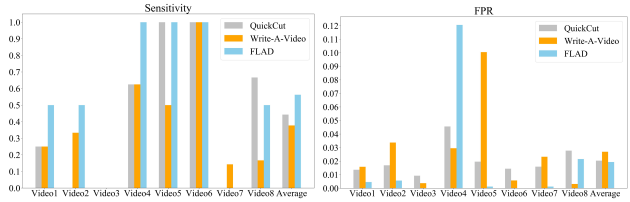

Figure 5: Comparison of Jitter Detection Algorithms

5.1.3 Occlusion Detection. For evaluating the occlusion detection model, we apply frame-level binary classification accuracy, receiver operating characteristic (ROC) curve with area under the curve

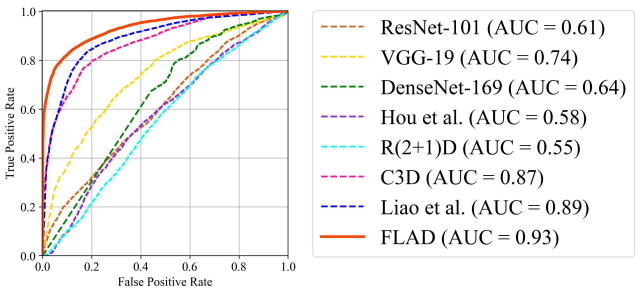

Figure 6: ROC Curves of Occlusion Detection

Table 3: Results of Flaw Detection Compared with Novices

Video No.	Blurriness				Jitter				Occlusion			
	Sensitivity		*FPR*		*Sensitivity*		*FPR*		*Sensitivity*		*FPR*	
	Novices	FLAD	Novices	FLAD	Novices	FLAD	Novices	FLAD	Novices	FLAD	Novices	FLAD
Video1	0.2143	0.7500	0.0008	0.1379	0.2500	0.5000	0.0034	0.0045	0.5714	0.5833	0.0111	0.0701
Video2	0.1429	1.0000	0.0027	0.4682	0.2381	0.5000	0.0067	0.0056	0.3571	1.0000	0.0088	0.4270
Video3	-	-	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.4694	1.0000	0.0783	0.1070
Video4	0.7143	1.0000	0.0034	0.0590	0.4286	1.0000	0.0352	0.1206	-	-	0.0000	0.1314
Video5	-	-	0.0019	0.4301	0.1429	1.0000	0.0004	0.0012	0.5714	0.6667	0.0033	0.1801
Video6	-	-	0.0000	0.0000	0.8571	1.0000	0.0016	0.0000	0.8571	1.0000	0.0053	0.9626
Video7	-	-	0.0005	0.0000	0.0816	0.0000	0.0014	0.0012	0.6429	1.0000	0.0051	0.0623
Video8	-	-	0.0009	0.0000	0.2857	0.5000	0.0409	0.0215	-	-	0.0000	0.0585
Average	0.3572	**0.9167**	**0.0013**	0.1369	0.2855	**0.5625**	**0.0112**	0.0193	0.5782	**0.8750**	**0.0140**	0.2499

Table 4: Experimental Results of Occlusion Detection

Method	Parameters	Accuracy	FPS
ResNet-101[7]	42.5M	0.6106	83
VGG-19[31]	139.59M	0.6885	70
DenseNet-169[10]	12.49M	0.6556	95
Hou et al.[9]	23.51M	0.4266	60
R(2+1)D[37]	33.18M	0.5910	99
C3D[36]	107.36M	0.7839	109
Liao et al.[18]	59.64M	0.8270	106
FLAD	50.17M	**0.8557**	**126**

(AUC), and frame per second (FPS) as evaluation metrics. The experiments are conducted in the occlusion detection dataset built by Liao et al. [18] with the same experimental settings as the benchmark. Since only minor studies focus on the video occlusion detection, regardless of four SOTA occlusion detection methods (Liao et al. [18], Hou et al. [9], R(2+1)D [37], C3D [36]), three most representative deep neural network models (ResNet-101 [7], VGG-19 [31], DenseNet-169 [10]) are also included as comparative methods.

The numbers of parameters and classification accuracy of different methods are shown in Table 4. As shown in this table, the model of the FLAD system could obtain the highest classification accuracy and FPS on this dataset, while the number of parameters is only 50.17M which is less than the model of the SOTA method (Liao et al. [18]). Moreover, the ROC curves with AUC values are illustrated in Figure 6. We can find that the model of the FLAD system has better AUC values (0.93) compared with other models. The experimental results illustrate the proposed FLAD system could achieve the SOTA performance in video occlusion detection.

5.2 Comparison with Novices

Compared with the existing methods, the FLAD system could achieve SOTA in the detection of the three video flaws, then we evaluate whether the FLAD could achieve higher detection ability compared with novices in the proposed testbed. The detailed experimental results are illustrated in Table 3, in which the notation '-' denotes the videos are not annotated with blurriness or occlusion by the professional video editor. Note that, for novices, the average values of their experimental results are utilized in comparison.

(1) Blurriness detection: The FLAD has a significantly higher average *Sensitivity* compared with novices, which is higher than 0.9. On the contrary, the novices have better control over the *FPR*, especially in video 2 and video 5. **(2) Jitter detection:** As shown in Table 3, the FLAD could achieve a *Sensitivity* of 0.5625, while the novices only have 0.2855, which is about half of the FLAD system. On the other hand, although the novices have better control over *FPR*, the *FPR* of the FLAD system is very close to the novices. In fact, the *Sensitivity* of 0.5625 also shows there are still possibilities for improvement. **(3) Occlusion detection:** The FLAD also shows a higher *Sensitivity* compared with novices, which is close to 0.9, but the *FPR* of FLAD is also higher than novices. In fact, the criteria or user acceptance are highly different for different people, but the experimental results could illustrate that, from the criteria of the professional editor, the FLAD is more sensitive to video flaws.

6 CONCLUSIONS

In this paper, we proposed a human-centered video content flaw detection system for meeting recordings, named FLAD, which builds a bridge between subjective sense and objective measures to assess the video quality. Specifically, we introduce three algorithms to detect the three most common flaws during the shooting of meeting recordings, containing blurriness, jitter, and occlusion. Afterward, the FLAD system would generate a visualization report for helping video editors fast locate the detected flaws. The experimental results illustrate that the proposed three algorithms can achieve SOTA performance compared with the existing approaches. And the proposed FLAD system also shows higher *Sensitivity* compared with novices, which means the FLAD system can effectively help video editors detect flaws in meeting recordings. In the future, we will keep improving the performance of proposed algorithms and enlarge the scalability of the FLAD system from the human-centered perspective, e.g. by extending more flaw detection modules.

ACKNOWLEDGMENTS

This work is supported by Project 61902333 by National Natural Science Foundation of China.

REFERENCES

[1] Usman Ali and Muhammad Tariq Mahmood. 2018. Analysis of blur measure operators for single image blur segmentation. *Applied Sciences* 8, 5 (2018), 807.

[2] Ido Arev, Hyun Soo Park, Yaser Sheikh, Jessica Hodgins, and Ariel Shamir. 2014. Automatic editing of footage from multiple social cameras. *ACM Transactions on Graphics* 33, 4 (2014), 1–11.

[3] Raghav Bansal, Gaurav Raj, and Tanupriya Choudhury. 2016. Blur image detection using Laplacian operator and Open-CV. In *2016 International Conference System Modeling & Advancement in Research Trends (SMART)*. IEEE, 63–67.

[4] Pei-Yu Chi, Joyce Liu, Jason Linder, Mira Dontcheva, Wilmot Li, and Bjoern Hartmann. 2013. Democut: generating concise instructional videos for physical demonstrations. In *Proceedings of the 26th annual ACM symposium on User interface software and technology*. 141–150.

[5] Martin A Fischler and Robert C Bolles. 1981. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Commun. ACM* 24, 6 (1981), 381–395.

[6] Andreas Girgensohn, John Boreczky, Patrick Chiu, John Doherty, Jonathan Foote, Gene Golovchinsky, Shingo Uchihashi, and Lynn Wilcox. 2000. A semi-automatic approach to home video editing. In *Proceedings of the 13th annual ACM symposium on User interface software and technology*. 81–89.

[7] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 770–778.

[8] Zhibin Hong, Chaohui Wang, Xue Mei, Danil Prokhorov, and Dacheng Tao. 2014. Tracking using multilevel quantizations. In *European Conference on Computer Vision*. Springer, 155–171.

[9] Ruibing Hou, Bingpeng Ma, Hong Chang, Xinqian Gu, Shiguang Shan, and Xilin Chen. 2019. Vrstc: Occlusion-free video person re-identification. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 7183–7192.

[10] Gao Huang, Zhuang Liu, Laurens Van Der Maaten, and Kilian Q Weinberger. 2017. Densely connected convolutional networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 4700–4708.

[11] Junhwa Hur and Stefan Roth. 2017. MirrorFlow: Exploiting symmetries in joint optical flow and occlusion estimation. In *Proceedings of the IEEE International Conference on Computer Vision*. 312–321.

[12] Eddy Ilg, Tonmoy Saikia, Margret Keuper, and Thomas Brox. 2018. Occlusions, motion and depth boundaries with a generic network for disparity, optical flow or scene flow estimation. In *Proceedings of the European Conference on Computer Vision (ECCV)*. 614–630.

[13] Saad M Khan and Mubarak Shah. 2008. Tracking multiple occluding people by localizing on multiple scene planes. *IEEE transactions on pattern analysis and machine intelligence* 31, 3 (2008), 505–519.

[14] Dieter Koller, Joseph Weber, and Jitendra Malik. 1994. Robust multiple car tracking with occlusion reasoning. In *European conference on computer vision*. Springer, 189–196.

[15] Rodrigo Laiola Guimaraes, Pablo Cesar, Dick CA Bulterman, Vilmos Zsombori, and Ian Kegel. 2011. Creating personalized memories from social events: community-based support for multi-camera recordings of school concerts. In *Proceedings of the 19th ACM international conference on Multimedia*. 303–312.

[16] Mackenzie Leake, Abe Davis, Anh Truong, and Maneesh Agrawala. 2017. Computational video editing for dialogue-driven scenes. *ACM Transactions on Graphics* 36, 4 (2017), 130–1.

[17] Florent Lefevre, Vincent Bombardier, Nicolas Krommenacker, Patrick Charpentier, and Bertrand Petat. 2018. Automatic video stream selection method by on-air microphone detection.

[18] Junhua Liao, Haihan Duan, Xin Li, Haoran Xu, Yanbing Yang, Wei Cai, Yanru Chen, and Liangyin Chen. 2020. Occlusion Detection for Automatic Video Editing. In *Proceedings of the 28th ACM International Conference on Multimedia*. 2255–2263.

[19] Qiong Liu, Yong Rui, Anoop Gupta, and Jonathan J Cadiz. 2001. Automating camera management for lecture room environments. In *Proceedings of the SIGCHI conference on Human factors in computing systems*. 442–449.

[20] Shijie Liu, Xiaohua Tong, Fengxiang Wang, Wenzheng Sun, Chengcheng Guo, Zhen Ye, Yanmin Jin, Huan Xie, and Peng Chen. 2016. Attitude jitter detection based on remotely sensed images and dense ground controls: A case study for Chinese ZY-3 satellite. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing* 9, 12 (2016), 5760–5766.

[21] A Chris Long, Brad Myers, Juan Casares, Scott Stevens, and Albert Corbett. 2004. Video Editing Using Lenses and Semantic Zooming. (2004).

[22] David G Lowe. 1999. Object recognition from local scale-invariant features. In *Proceedings of the seventh IEEE international conference on computer vision*, Vol. 2. IEEE, 1150–1157.

[23] Markus Mathias, Rodrigo Benenson, Radu Timofte, and Luc Van Gool. 2013. Handling occlusions with franken-classifiers. In *Proceedings of the IEEE International Conference on Computer Vision*. 1505–1512.

[24] Xiongkuo Min, Guangtao Zhai, Jiantao Zhou, Mylene CQ Farias, and Alan Conrad Bovik. 2020. Study of subjective and objective quality assessment of audio-visual signals. *IEEE Transactions on Image Processing* 29 (2020), 6054–6068.

[25] Wanli Ouyang and Xiaogang Wang. 2012. A discriminative deep model for pedestrian detection with occlusion handling. In *2012 IEEE Conference on Computer Vision and Pattern Recognition*. IEEE, 3258–3265.

[26] Wanli Ouyang and Xiaogang Wang. 2013. Joint deep learning for pedestrian detection. In *Proceedings of the IEEE international conference on computer vision*. 2056–2063.

[27] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. 2019. Pytorch: An imperative style, high-performance deep learning library. In *Advances in neural information processing systems*. 8026–8037.

[28] José Luis Pech-Pacheco, Gabriel Cristóbal, Jesús Chamorro-Martinez, and Joaquín Fernández-Valdivia. 2000. Diatom autofocusing in brightfield microscopy: a comparative study. In *Proceedings 15th International Conference on Pattern Recognition*, Vol. 3. IEEE, 314–317.

[29] Abhishek Ranjan, Jeremy Birnholtz, and Ravin Balakrishnan. 2008. Improving meeting capture by applying television production principles with audio and motion detection. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. 227–236.

[30] Reza Rassool. 2017. VMAF reproducibility: Validating a perceptual practical video quality metric. In *2017 IEEE international symposium on broadband multimedia systems and broadcasting*. IEEE, 1–2.

[31] Karen Simonyan and Andrew Zisserman. 2014. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556* (2014).

[32] Nicolas Staelens, Margaret H Pinson, Philip Corriveau, Filip De Turck, and Piet Demeester. 2015. Measuring video quality in the network: from quality of service to user experience. In *9th International Workshop on Video Processing and Consumer Electronics*. 5–6.

[33] Patrik Sundberg, Thomas Brox, Michael Maire, Pablo Arbeláez, and Jitendra Malik. 2011. Occlusion boundary detection and figure/ground assignment from optical flow. In *CVPR 2011*. IEEE, 2233–2240.

[34] Yoshinao Takemae, Kazuhiro Otsuka, and Naoki Mukawa. 2003. Video cut editing rule based on participants' gaze in multiparty conversation. In *Proceedings of the eleventh ACM international conference on Multimedia*. 303–306.

[35] Xiaohua Tong, Zhen Ye, Yusheng Xu, Xinming Tang, Shijie Liu, Lingyun Li, Huan Xie, Fengxiang Wang, Tianpeng Li, and Zhonghua Hong. 2014. Framework of jitter detection and compensation for high resolution satellites. *Remote Sensing* 6, 5 (2014), 3944–3964.

[36] Du Tran, Lubomir Bourdev, Rob Fergus, Lorenzo Torresani, and Manohar Paluri. 2015. Learning spatiotemporal features with 3d convolutional networks. In *Proceedings of the IEEE international conference on computer vision*. 4489–4497.

[37] Du Tran, Heng Wang, Lorenzo Torresani, Jamie Ray, Yann LeCun, and Manohar Paluri. 2018. A closer look at spatiotemporal convolutions for action recognition. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*. 6450–6459.

[38] Anh Truong, Floraine Berthouzoz, Wilmot Li, and Maneesh Agrawala. 2016. Quickcut: An interactive tool for editing narrated video. In *Proceedings of the 29th Annual Symposium on User Interface Software and Technology*. 497–507.

[39] Shuhei Tsuchida, Satoru Fukayama, and Masataka Goto. 2017. Automatic system for editing dance videos recorded using multiple cameras. In *International Conference on Advances in Computer Entertainment*. Springer, 671–688.

[40] Miao Wang, Guo-Wei Yang, Shi-Min Hu, Shing-Tung Yau, and Ariel Shamir. 2019. Write-a-video: computational video montage from themed text. *ACM Transactions on Graphics* 38, 6 (2019), 1–13.

[41] Mi Wang, Ying Zhu, Jun Pan, Bo Yang, and Quansheng Zhu. 2016. Satellite jitter detection and compensation using multispectral imagery. *Remote Sensing Letters* 7, 6 (2016), 513–522.

[42] Yang Wang, Yi Yang, Zhenheng Yang, Liang Zhao, Peng Wang, and Wei Xu. 2018. Occlusion aware unsupervised learning of optical flow. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 4884–4893.

[43] Alper Yilmaz, Xin Li, and Mubarak Shah. 2004. Contour-based object tracking with occlusion handling in video acquired using mobile cameras. *IEEE Transactions on pattern analysis and machine intelligence* 26, 11 (2004), 1531–1536.

[44] Yingxue Zhang, Yingbin Wang, Feiyang Liu, Zizheng Liu, Yiming Li, Daiqin Yang, and Zhenzhong Chen. 2018. Subjective panoramic video quality assessment database for coding applications. *IEEE Transactions on Broadcasting* 64, 2 (2018), 461–473.

[45] Chunluan Zhou and Junsong Yuan. 2018. Bi-box regression for pedestrian detection and occlusion estimation. In *Proceedings of the European Conference on Computer Vision (ECCV)*. 135–151.

[46] Vilmos Zsombori, Michael Frantzis, Rodrigo Laiola Guimaraes, Marian Florin Ursu, Pablo Cesar, Ian Kegel, Roland Craigie, and Dick CA Bulterman. 2011. Automatic generation of video narratives from shared UGC. In *Proceedings of the 22nd ACM conference on Hypertext and hypermedia*. 325–334.