Occlusion Detection for Automatic Video Editing

Junhua Liao*
Sichuan University
Chengdu, China
junhua_liao@yeah.net

Haihan Duan*
The Chinese University of Hong Kong, Shenzhen
Shenzhen Institute of Artificial Intelligence and Robotics for Society
Shenzhen, China
haihanduan@link.cuhk.edu.cn

Xin Li
Haoran Xu
Sichuan University
Chengdu, China
lixinlee3@gmail.com
xuhaoran1126@126.com

Yanbing Yang
Sichuan University
The Institute for Industrial Internet Research, Sichuan University
Chengdu, China
yangyanbing@scu.edu.cn

Wei Cai
The Chinese University of Hong Kong, Shenzhen
Shenzhen Institute of Artificial Intelligence and Robotics for Society
Shenzhen, China
caiwei@cuhk.edu.cn

Yanru Chen
Liangyin Chen†
Sichuan University
The Institute for Industrial Internet Research, Sichuan University
Chengdu, China
yanruchen@stu.scu.edu.cn
chenliangyin@scu.edu.cn

ABSTRACT

Videos have become the new preference comparing with images in recent years. However, during the recording of videos, the cameras are inevitably occluded by some objects or persons that pass through the cameras, which would highly increase the workload of video editors for searching out such occlusions. In this paper, for releasing the burden of video editors, a frame-level video occlusion detection method is proposed, which is a fundamental component of automatic video editing. The proposed method enhances the extraction of spatial-temporal information based on C3D yet only using around half amount of parameters, with an occlusion correction algorithm for correcting the prediction results. In addition, a novel loss function is proposed to better extract the characterization of occlusion and improve the detection performance. For performance evaluation, this paper builds a new large scale dataset, containing 1,000 video segments from seven different real-world scenarios, which could be available at: https://junhua-liao.github.io/Occlusion-Detection/. All occlusions in video segments are annotated frame by frame with bounding-boxes so that the dataset could be utilized in both frame-level occlusion detection and precise occlusion location. The experimental results illustrate that the proposed method could achieve good performance on video occlusion detection compared with the state-of-the-art approaches. To the best of our knowledge, this is the first study which focuses on occlusion detection for automatic video editing.

CCS CONCEPTS

• **Information systems** → **Multimedia content creation**; • **Computing methodologies** → *Visual content-based indexing and retrieval*.

KEYWORDS

Occlusion Detection; Automatic Video Editing; Human–Computer Interaction; Deep Learning

ACM Reference Format:
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 (MM '20), October 12–16, 2020, Seattle, WA, USA.* ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/3394171.3413725

1 INTRODUCTION

With the rise of "We Media", more and more people will record some interesting videos and share them on the Internet after editing. However, video editing is a relatively professional and labor-intensive human-computer interaction (HCI) procedure. Thus, an automatic video editing system can, in a certain degree, relieve the video editors' burden or help people with less professional experience to edit video.

In recent years, automatic video editing has gradually known to the public. As early as 2016, Truong et al. [34] developed QuickCut, which utilizes dynamic programming algorithm to edit narrative videos. A year later, Leake et al. [22] implemented an automatic editing system for dialogue scenes. In the same year, Tsuchida et al. [35] automatically cropped multi-camera dance videos based on the expert's prior knowledge, performer behavior, and skeleton information. In 2019, Wang et al. [37] developed "Write-A-Video", which extracts keywords presented by users on text to match the video materials for automatic editing. In terms of video quality,

*Both authors contributed equally to this research.
†Liangyin Chen is the corresponding author. chenliangyin@scu.edu.cn

MM '20, October 12–16, 2020, Seattle, WA, USA.

ACM ISBN 978-1-4503-7988-5/20/10…$15.00
https://doi.org/10.1145/3394171.3413725

QuickCut [34] is only limited to recognize the blur and shake of shots. "Write-A-Video" [37] filters video segments by combining the stability of the shots, brightness and continuity of the image. Therefore, none of them have done specific study on the quality of video editing, leading to unsatisfactory results.

Automatic video editing is a very complex task, since the shooting scenes in reality are usually complicated and diverse, and the video quality is also affected by abnormal images such as blur, shake and occlusion. In fact, although the artistry of automatic video editing remains a quite tough issue to be solved since there are some subjective factors, the quality of video must be an indispensable evaluation criterion. Besides the man-made special effects, most videos with blur, shake or occlusion would be regarded as low quality videos. So far, there are a lot of works which achieve good results [2, 18, 42] for the detection of blur and shake in videos, while few researches focus on occlusion. In real circumstance of video shooting, it is inevitable that some objects or persons might pass through the front of the cameras within a short period. These shots would be regraded as occluded shots in post video editing, which should be searched out and cut off by video editors. For example, as shown in Figure 1, if a camera is occluded at a certain time, other unblocked cameras should be used to fill the content of the time period. However, the occurrences of occlusions are hard to be predicted only using the information from the video, so the searching of occlusions, as a HCI procedure, is very labor-intensive for video editors, which is a highly pressing problem needed to be solve by automatic video editing. Therefore, as the pre-condition of an automatic video editing system to better optimize the HCI procedure, a reliable algorithm for video shot occlusion detection is imperative to be proposed.

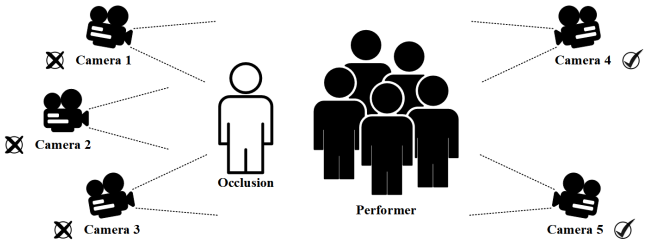

Figure 1: Schematic diagram of shot selection for multi-camera video editing.

There are four difficulties of shots occlusion detection we summarized as follows: (1) The occlusion that appears in the shot has no fixed shape. As a result, the occlusion is not like the objects in the traditional object detection task which have significant characteristics for reference, while depending on the relationship between each objects. (2) Even normal images can contain complex occlusion relationships. For example, in a news video, the host would be occluded by the table in front of him/her, but the table would not be regraded as an occlusion during editing. (3) The same object may play different roles when it appears in the camera at different times. For example, in an awards ceremony, when the winner receives the award on the stage, he belongs to the subject of the shots. When he reappears in the shots on his way back to his seat, he may become an occlusion. (4) There is not an existing dataset to evaluate the performance of occlusion detection. A high-quality dataset is a booster for further development. Therefore, how to effectively detect shots with occlusion is a very challenging task and will make an important contribution to the development of automatic video editing in the future.

In this paper, we built a large scale dataset for evaluating the performance of occlusion detection in videos. Because the video editing is performed at frame level, this paper proposes a frame-based occlusion detection method. By training on a dedicated dataset, the proposed deep learning model can predict occlusion scores for each frame in the video in order to detect occlusions. To the best of our knowledge, this is the first paper which focuses on the shots occlusion detection for automatic video editing, while current researchers mainly pay attention to the rule of video editing and assume the quality of the video materials could satisfy the standards. In summary, the contributions of this paper are as follows,

- We have built a large dataset for video occlusion detection, which contains video segments from a total of 1000 shots with occlusions of 7 scenarios in the real world. To the best of our knowledge, this is the first dataset about video shots occlusion.
- We propose a frame-based occlusion detection method with a specific occlusion loss function dedicated to video occlusion detection for better training. And an occlusion correction algorithm based on the occurrence pattern of occlusions is also implemented to correct the prediction results.
- The experimental results on the dedicated dataset illustrate that the proposed method could achieve an optimal detection accuracy of 84.0%.

2 RELATED WORK

Occlusion detection and handling play an important role in computer vision tasks, including pedestrian detection [24, 26, 27, 39, 47], video tracking [12, 20, 21, 25, 43], stereo matching [4, 19, 30, 36, 48]and optical flow estimation [16, 17, 31, 40, 44].

Under the hypothesis of Lambertian reflection and static illumination, Ayvaci et al. [1] considered the detection of occlusion area as a variational optimization problem and used convex minimization approximation to solve the problem. Instead of treating occlusion as another source of noise, Pepikj et al. [28] included the occluder itself into the modelling, by mining distinctive, reoccurring occlusion patterns from annotated training data. Different from modeling local coherency of occlusions or attempting to learn the structure of occlusions from data, Hsiao et al. [14] constructed an interactive model of a three-dimensional object to represent occlusion from arbitrary viewpoint. Liu et al. [23] used semantic phrases to establish 8 occlusion modes for target detection on VOC dataset [6]. In the video segmentation task, Wu et al. [41] believed that a basic idea of occlusion detection is to model the size of each segment over time, and to predict when the size will reach 0. They used the past 10 frames to build a linear model to predict the size of the segment in the next frame, and the segment would be considered as an occlusion if the ratio of a segment's predicted size to the average size of the last 10 frames falls below the threshold. Zhou et al. [46] observed that different status of occlusion would lead to different confidence of pedestrian detectors, while some detectors give high

scores with large visible area and others give low scores because of occlusion. Moreover, the detection score of the part detector is often low for the non-pedestrian region. Based on this, Zhou et al. used K part detectors instead of whole-body detectors in pedestrian detection, and averaged the highest S scores among the K scores of the detectors to determine whether an occlusion occurred. Zhang et al. [45] designed a part occlusion-aware RoI pooling unit. They divided the pedestrian region into five areas based on the empirical ratio [7], and estimated the occlusion status of each area using the micro neural network equipped in the proposed unit. Hou et al. [13] believed that occlusion usually occurred in successive frames, and the occluders have different semantic features from the original body parts. So Hou et al. calculated the cosine similarity between frame region feature and video region feature as a criteria score, and the area is considered as being occluded when the score is lower than the prescribed threshold.

There are different definitions of occlusion in different application scenarios. For example, in pedestrian detection and video tracking, the occlusion means the object for detecting or tracking is occluded. So the design of corresponding occlusion detection are also diverse. Therefore, an occlusion detection method that excels in specific tasks does not mean that it can perform well in other tasks, and even cannot be migrated to other tasks. By contrast, the concept of occlusion in automatic video editing is difficult to define accurately, since it is quite subjective and varies from person to person. In this article, we consider that a shot has occlusion when an object appears at the inappropriate time, e.g., the object occludes the leading role of the shot. Although there are many studies on occlusion detection, to the best of our knowledge, this is the first study to deal with occlusion detection for automatic video editing.

3 METHODOLOGY

3.1 Occlusion Detection Method

In our proposed method, we regard the occlusion detection task of video shots as a frame-level classification problem. The flowchart of this method is shown in Figure 2. Considering different resolutions of videos, we firstly resize the video frames to a specific resolution and divides them into segments with a fixed number of frames. The experimental parameters will be introduced in Section 5.1. Secondly, the improved C3D [32] extracts the temporal and spatial feature of the input video segments. There are two reasons for choosing C3D as the basic framework for occlusion detection. On the one hand, the occlusion detection task is similar to the action recognition, which firstly extract features from videos and then perform classification. On the other hand, C3D is a classic network model with relatively simple architecture, which could achieve good performance on different tasks, e.g., action recognition, action similarity labeling, scene and object recognition. Therefore, we believe that the C3D-based model can also perform well in occlusion detection task. When pooling, we adjust the stride and kernel size of the original pooling layer in temporal dimension of C3D to 1, so as to ensure that it becomes a model with multiple input and multiple output. This adjustment will affect the extraction of temporal information, so we added a Non-Local block [38] to compensate for the temporal information. Unlike convolutional neural networks and recurrent neural networks that operate on local regions, Non-Local block will consider the weights of all location features when calculating the features of a certain location (spatial or temporal). The introduction of global information through Non-Local block can deliver richer information to the subsequent network layer. In addition, we also modify the channels of some convolutional layers, the stride and kernel size of the pooling layer to reduce the model parameters. The detailed parameters of our model is shown in Table 1. Then, we send the temporal and spatial features extracted from each frame to fully connected layers, with dropout layers between each fully connected layers for prevention of overfitting. After that, a softmax function is utilized to calculate the probability of occlusion, named as occlusion score. Finally, an occlusion correction algorithm based on the occurrence pattern of occlusions is implemented to correct the occlusion score. In our experiments, a frame would be defined as an occlusion if its occlusion score greater than or equal to 0.5, otherwise it is considered without occlusion.

Table 1: Proposed model for feature extraction

Layer	Parameter	Output Size
Conv1	$3 \times 3 \times 3$, 32, stride 1, 1, 1	$32 \times 8 \times 128 \times 171$
Pool1	$1 \times 2 \times 2$, max, stride 1, 2, 2	$32 \times 8 \times 64 \times 85$
Conv2	$3 \times 3 \times 3$, 64, stride 1, 1, 1	$64 \times 8 \times 64 \times 85$
Pool2	$1 \times 2 \times 2$, max, stride 1, 2, 2	$64 \times 8 \times 32 \times 42$
Conv3a	$3 \times 3 \times 3$, 128, stride 1, 1, 1	$128 \times 8 \times 32 \times 42$
Conv3b	$3 \times 3 \times 3$, 256, stride 1, 1, 1	$256 \times 8 \times 32 \times 42$
Non-Local	channels 256	$256 \times 8 \times 32 \times 42$
Pool3	$1 \times 2 \times 2$, max, stride 1, 2, 2	$256 \times 8 \times 16 \times 21$
Conv4a	$3 \times 3 \times 3$, 512, stride 1, 1, 1	$512 \times 8 \times 16 \times 21$
Conv4b	$3 \times 3 \times 3$, 512, stride 1, 1, 1	$512 \times 8 \times 16 \times 21$
Pool4	$1 \times 2 \times 2$, max, stride 1, 2, 2	$512 \times 8 \times 8 \times 10$
Conv5a	$3 \times 3 \times 3$, 512, stride 1, 1, 1	$512 \times 8 \times 8 \times 10$
Conv5b	$3 \times 3 \times 3$, 512, stride 1, 1, 1	$512 \times 8 \times 8 \times 10$
Pool5	$1 \times 3 \times 3$, max, stride 1, 3, 3	$512 \times 8 \times 3 \times 4$

3.2 Occlusion Loss Function

In order to better adapt the characteristics of occlusion detection task, we propose a special loss function for model training, shown as follows:

$$L_{occ} = -e^{\frac{occ\_ratio}{\lambda}}(l_j^i \log(o_j^i) + (1 - l_j^i)\log(1 - o_j^i))$$

Where l_j^i and o_j^i represent the label and prediction result of the j-th frame in the i-th video respectively. In our experiments, there are only two labels, where the normal frame is represented by 0 and the occlusion frame is denoted by 1. occ_ratio represents the proportion of the frame occupied by occlusion, and its calculation formula is as follows:

$$occ\_ratio = \frac{occlusion\_area}{frame\_area}$$

Where $occlusion\_area$ is the area of occlusion and $frame\_area$ is the original area of the frame. When annotating the data, we enclose the occlusions with bounding-boxes, so the area of this box is regarded as the area of occlusion. In multi-camera video editing, the shot with missed detected occlusion would be left in the final video which may cause highly negative effects on the quality of

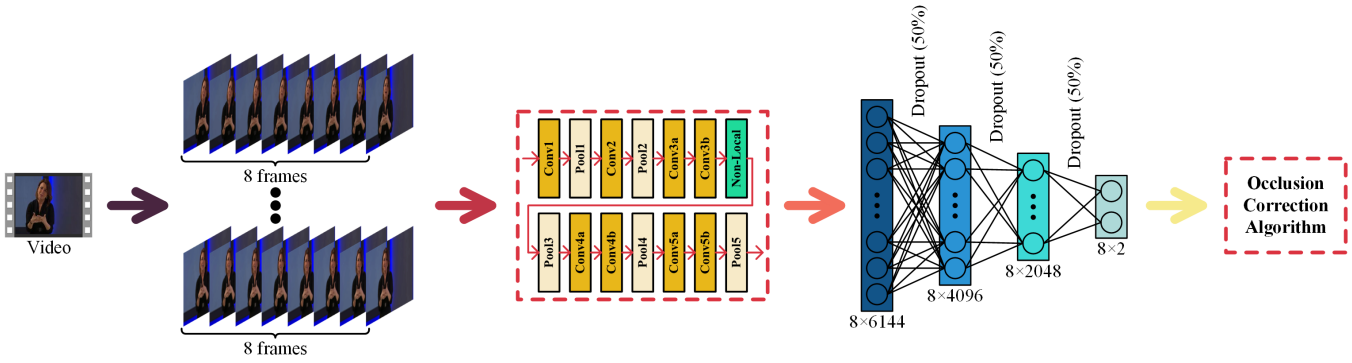

Figure 2: The flowchart of the proposed occlusion detection approach.

experience, while a shot with false detection of occlusion would be replaced by other normal shots with less influence. Therefore, we expect the model could be more sensitive to occlusion by giving it more weight to the occlusion frame. However, excessive penalty will make the model difficult to fit, so we reduce the *occ_ratio* by λ times. In this paper, $\lambda = 10$ can improve the frame weight while converge the model.

3.3 Occlusion Correction Algorithm

In most scenes, video shot occlusions will last for some time, e.g., a person who passes through the camera usually need several seconds. So it is rare for occlusions to occur several times in a short time period. Even if it happens, the whole segments during this period should be cut off by video editors. In addition, we notice two points by observing the prediction results of the test set. Firstly, 77.9% prediction results are in the two ranges of $[0, 0.05]$ and $[0.95, 1]$, and the accuracy of frames in these two ranges is as high as 89.0% (details would be discussed in Section 5.2). Therefore, we intuitively regard the prediction results of frames in these two ranges as right results, which could be utilized to correct the result of a sliding window based on the number of frames falling into the two ranges. Secondly, we found that there are lots of false predictions when the predicted occlusion state changes frequently within a short time period. Based on the occurrence pattern of occlusion and above-mentioned observations, we propose an occlusion correction algorithm to further improve accuracy, as shown in Algorithm 1. We use a window of w frame size to slide with a step of s, and calculate the number of frames in the window with occlusion scores in the range of $[0, 0.05]$ and $[0.95, 1]$, respectively. When the occlusion state in the window changes more than z times, the prediction result will be adjusted according to the principle of minority obeying the majority. In this article, we set $w = 36$, $s = 26$, $z = 3$.

Algorithm 1: Occlusion Correction Algorithm

Input: $X = \{x_i\}$: the original prediction result sequence
w: sliding window size
s: stride
θ: occlusion threshold
z: fluctuation threshold
Output: $\widehat{X} = \{\widehat{x_i}\}$: the corrected result sequence

Initialize $t = 0, \widehat{X} = X$;
while $t + w \leq |X|$ **do**
  $W = \{x_t, x_{t+1}, \cdots, x_{t+w-1}\}$;
  $M = \{x_k \in W | x_k \geq 0.95\}$;
  $m = \{x_k \in W | x_k \leq 0.05\}$;
  $Z = \{x_k \in W | x_k \geq \theta, x_{k+1} < \theta \text{ or } x_k < \theta, x_{k+1} \geq \theta\}$;
  if $|Z| \geq z$ **then**
    if $|M| \geq |m|$ **then**
      replace $\widehat{x_i} < 0.95, i \in [t, t + w)$ in $\widehat{X}$ with 0.95 ;
    else
      replace $\widehat{x_i} > 0.05, i \in [t, t + w)$ in $\widehat{X}$ with 0.05 ;
    end
  end
  $t = t + s$
end

4 DATASET

Previous datasets [3, 5, 6, 8, 9] used for occlusion detection were not designed for video shots occlusion detection tasks. Therefore, we have established a new large-scale dataset to evaluate the proposed method. The dataset contains 7 real world scenes, including academic conferences, activities, annual conferences, movies, panel speak, pre-game press conferences and summits, which are usually shotted by multiple cameras simultaneously. Therefore, they are well matched our original intention for occlusion detection in automatic multi-camera video editing.

Video collection. In order to ensure the quality of the dataset, we trained three annotators with basic computer vision knowledge to build the dataset. We searched videos from YouTube and cinematographic for footage that include pre-defined occlusions. Then we utilized Corel VideoStudio[1] to clip the videos to short segments. During clipping, we usually retained some normal scenes before and after occlusion, and make sure all video segments were without transition. To reduce storage space for facilitating the download of dataset, we re-encode the clipped segments using the video codec

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

Figure 3: Examples of different scenes in our dataset.

Table 2: Total number of videos of each scenario in our dataset

Scenario	Training Set	Testing Set
Academic Conference	115	29
Activity	95	24
Annual Conference	229	59
Movie	135	34
Panel Speak	89	18
Pre-Game Press Conference	56	15
Summit	81	21
Total	**800**	**200**

'XviD'. Totally 1,000 video segments were collected with occlusion. Five sample frames of occlusion in seven scenarios were illustrated in Figure 3.

Annotation. For automatic video editing, only frame-level labels are required, but we annotated the bounding-box of occlusion in each frame for the potential of further extending, which cost several months to complete the dataset. Noting that, some of the videos have multiple occlusion. The annotated occlusions are shown by the green bounding-boxes in Figure 3.

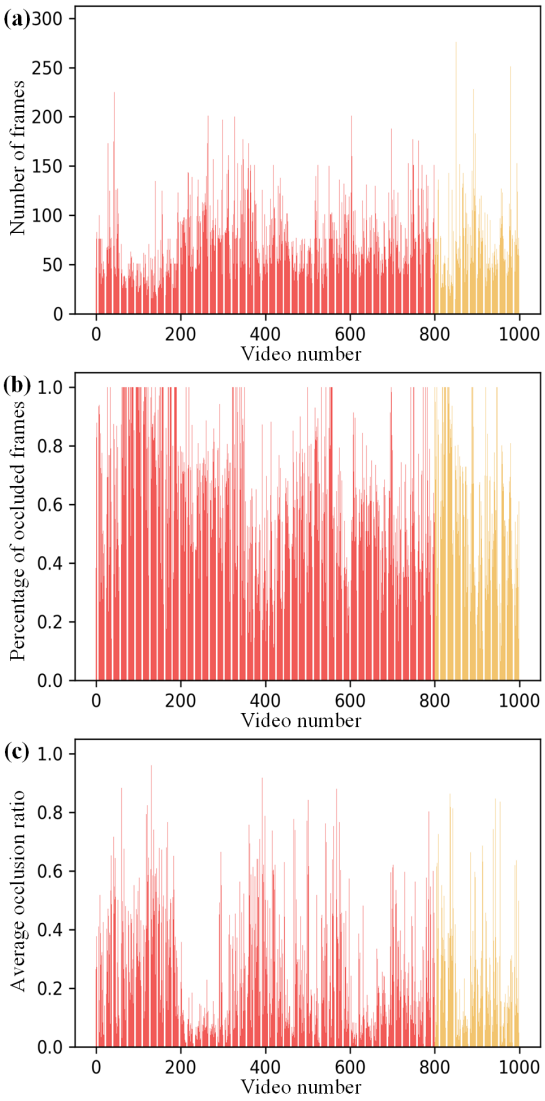

Figure 4: The number of video frames, the percentage of occluded frames and the average occlusion ratio in the dataset. On the horizontal axis, the number of 0 to 799 are training sets and 800 to 999 are testing sets.

Training and testing sets. We randomly divided the dataset into two parts, including 800 videos for the training set and 200 videos for the testing set (details are shown in Table 2). And it can be seen that videos of the 7 scenarios in the training set and the testing set have the roughly same proportion. The number of frames, the percentage of occluded frames, and the average occlusion ratio in each video are shown in Figure 4(a), (b), and (c) respectively.

5 EXPERIMENTS

5.1 Implementation Details

Before feature extraction, we resize each video frame to 128 × 171 pixels. Limited to the 4GB RAM of NVIDIA GTX 1050Ti GPU, the

Table 3: The results of occlusion correction algorithm in different ranges

Ranges	Proportion	AFSR	AEVAC
$[0, 0.05] \cup [0.95, 1]$	77.9%	89.0%	84.0%
$[0, 0.10] \cup [0.90, 1]$	83.8%	87.1%	83.2%
$[0, 0.15] \cup [0.85, 1]$	87.5%	85.9%	83.0%
$[0, 0.20] \cup [0.80, 1]$	90.1%	85.2%	82.9%
$[0, 0.25] \cup [0.75, 1]$	92.3%	84.5%	82.9%
$[0, 0.30] \cup [0.70, 1]$	94.0%	83.9%	82.5%

Table 4: Accuracy of various approaches on our dataset

Method	Parameters	Accuracy
ResNet-50[11]	23.51M	59.3%
VGG-19[29]	139.59M	63.3%
DenseNet-121[15]	6.95M	64.9%
Zhang et al.[45]	0.04M	44.5%
Hou et al.[13]	23.51M	38.0%
R(2+1)D[33]	33.18M	57.4%
C3D(single output)[32]	107.36M	74.8%
C3D(multiple output)[32]	107.36M	77.7%
Our Method(without correction)	59.64M	82.1%
Our Method(with correction)	59.64M	84.0%

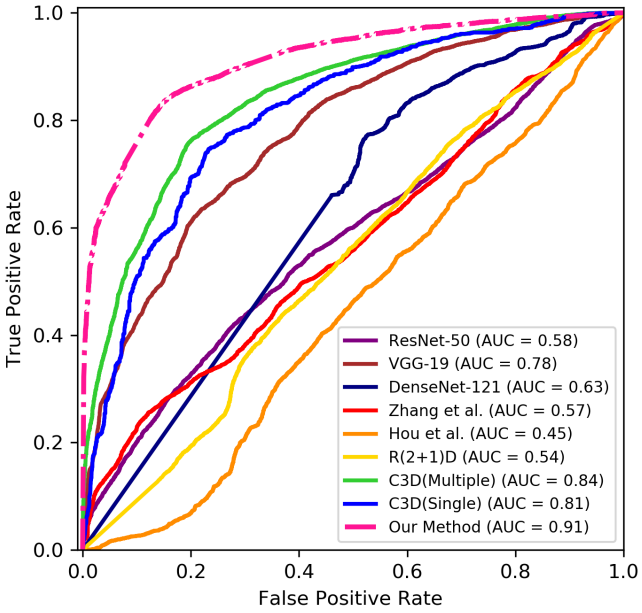

Figure 5: ROC comparison of ResNet-50 [11] (purple), VGG-19 [29] (brown), DenseNet-121 [15] (navy), Zhang et al. [45] (red), Hou et al. [13] (darkorange), R(2+1)D [33] (gold), C3D(Multiple) [32] (limegreen), C3D(Single) [32] (blue) and our method (deeppink).

input video is divided as multiple segments every 8 consecutive frames for feature extraction. When the number of remained frames of the video is less than 8, the last frame of the video would be used for padding. The network is trained utilizing Stochastic Gradient Descent (SGD), with 0.9 momentum and 0.0005 weight decay. The learning rate is initialized as 0.0001, and the training process contains 50 epochs with learning rate decay of 0.5 every 10 epochs.

Evaluation Metric. For performance evaluation, we utilize frame-level accuracy, receiver operating characteristic (ROC) curve and its corresponding area under the ROC curve (AUC) to evaluate the performance of our method.

5.2 Comparison with the State-of-the-art

Due to the lack of related work on occlusion detection of video shots, we compared the proposed method with three classic CNN models, two analogous state-of-the-art occlusion detection methods and two classic action recognition methods. Zhang et al. [45] divides the pedestrian region into five fixed-scale regions, but fails to cover the entire pedestrian region. In the experiment, we regard the entire frame as a pedestrian region, and widen three regions that are 0.7 times of the width of the frame to 1.0 times so that the divided five regions could completely cover the frame. Then, following with the original paper, we apply the RoI pooling layer [10] to extract features for each of the five regions and send them to the occlusion processing unit designed for calculating the visibility score of each region. If the visibility score of any region is less than the 0.65 set in the original paper, we consider that the frame is occluded. For the task of video person re-identification, Hou et al. [13] used the similarity score to determine whether the occlusion occurred. According to their method, each frame are divided into three regions (upper, middle, and lower) and the ResNet-50 [11] trained on our occlusion detection dataset was utilized to extract the features of each region, called region feature. For each video, three mean values of the three region features of the whole video are calculated, named video region feature. After that, three cosine similarities are calculated from the region features of every frame and the video region feature as the score. If one of three scores of a frame is lower than the threshold of 0.89 provided in the original paper, the frame is regarded as occlusion. In action recognition, we choose two classic methods, C3D [32] and R(2+1)D [33]. They apply different motivations for action recognition and achieve good performance, where the C3D [32] utilizes 3D convolution and the R(2+1)D [33] combines 2D convolution and 1D convolution to extract spatial-temporal features. We adjust the temporal stride of R(2+1)D to 1, and the stride and kernel size of the C3D pooling layer in the temporal dimension to 1, so as to ensure the temporal dimension does not change.

Table 3 shows the results of the occlusion correction algorithm when selecting different ranges. Among them, *Ranges* denotes the range of prediction scores and *Proportion* means the proportion of the number of corresponding frames to total frames. The accuracy of frames in the selected range (*AFSR*) and the accuracy of the entire video after correction (*AEVAC*) are illustrated in Table 3. It can be seen that when the algorithm selects the $[0, 0.05]$ and $[0.95, 1]$ ranges, the best accuracy can be achieved.

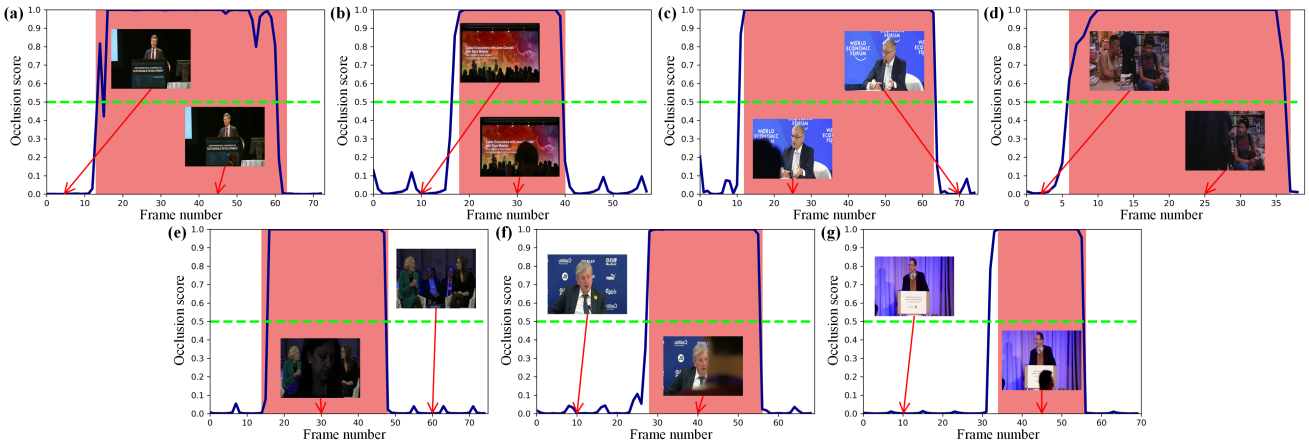

Figure 6: Positive results of our method on testing videos. Colored window shows ground truth of occlusion region. The solid blue line represents the predicted occlusion score of our method, and the green dotted line is the occlusion threshold.

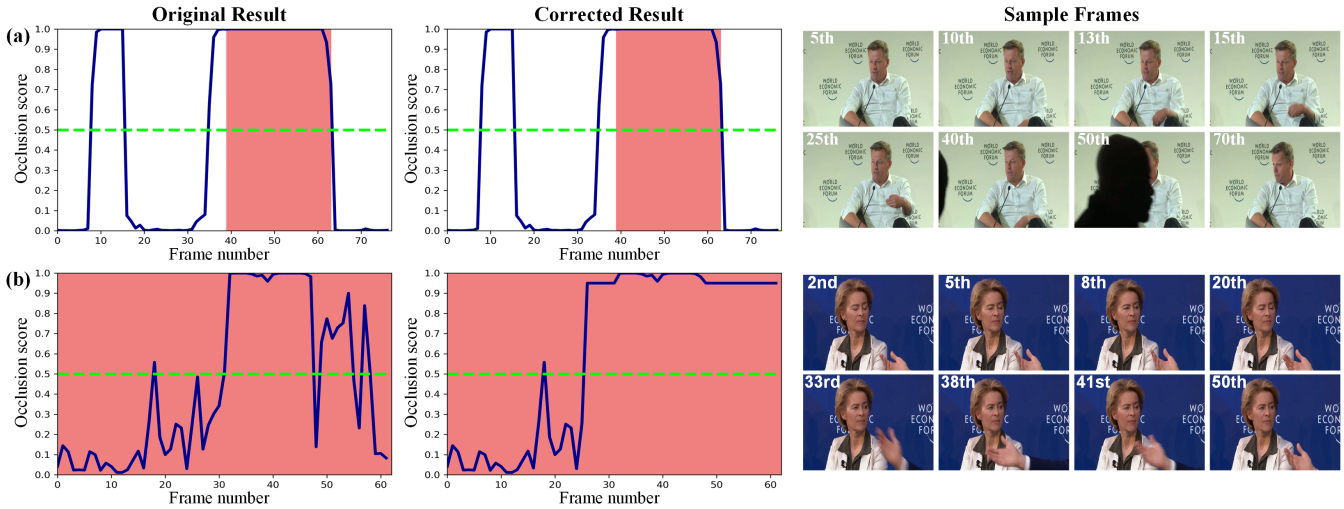

Figure 7: Negative results of our method on testing videos. Colored window shows ground truth of occlusion region. The solid blue line represents the predicted occlusion score of our method, and the green dotted line is the occlusion threshold.

Table 4 illustrates the comparison results of our method with previous methods in the amount of model parameters and the accuracy of the testing set. The comparison of ROC curve and AUC is shown in Figure 5. Experimental results illustrate that our method is significantly better than the existing methods.

According to the results, using similarity score [13] displays the worst effect of occlusion detection, although it performed well when identifying occlusions that cause large screen changes like black screens. However, when the occlusion only occupies small proportion of a frame, the similarity still maintain at a high score and leads to miss detection. The method of Zhang et al. [45] has higher accuracy and AUC value, but its accuracy still fails to reach 50%. The reason of its poor performance could be concluded as three reasons: 1) the regional division is based on pedestrians, while this division may not be the optimal solution in our task. 2) during labeling, only when the occlusion covers more than 50% area would the frame be regarded as occlusion by their method, thus some small occlusions are ignored. 3) Zhang et al. [45] used a micro-neural network to calculate the visibility score, which is prone to under fitting due to the small amount of parameters. Since the three classic CNN models [11, 15, 29] do not introduce temporal information, their accuracy can only reach about 60%. As for the method of action recognition, although the R(2+1)D [33] model could reach accuracy of 57.4%, its ROC curve and AUC value are still not satisfactory. The advantage of this model is that it can reduce the model parameters by splitting the 3D convolution, which makes it easier to optimize. In our task, in order to keep the temporal dimension to be fixed, the relevant parameters will be modified, but such modification

would affect the extraction of temporal information. R(2+1)D with 1D convolution in time series may not be as good as 3D convolution in temporal information extraction. Therefore, the performance of R(2+1)D is not as good as C3D [32] based on 3D convolution. We also compared single-output C3D and multi-output C3D, and found that the performance of multi-output C3D is slightly better. The video shots occlusion detection task needs to classify every frames in videos. Therefore, it might cause errors if using one result to represent the entire sequence since the input sequence may contain both occlusion frames and normal frames. The modifications in the temporal dimension also affected C3D, so we introduce the Non-Local block for temporal information compensation to achieve better performance. On the other hand, the amount of parameters of the proposed model only needs half of the original C3D since the addition of Non-Local block and other modification. In addition, the occlusion correction algorithm based on the actual situation and the prediction results improved the accuracy by 1.9%, which proved its effectiveness.

5.3 Case Analysis

In Figure 6, some positive prediction results of the proposed method on seven scene videos are illustrated, where (a) to (g) represent the prediction results of a representative video from the academic conferences, activities, annual conferences, movies, panel speak, pre-game press conferences and summits respectively. The proposed method is able to detect these occluded shots by predicting a high score for the occluded frames.

Two typical misrecognized cases are shown in Figure 7. For Figure 7(a), at the beginning of the video, the left hand of the subject is put behind the shot, but he suddenly raise his hand at about 10th frame, which is misdetected by the proposed model. And Figure 7(b) is an example of missed detection. There is a hand which could be regarded as an occlusion runs through the entire video, but our model only detects occlusion at some frames. We notice that the area occluded by the hand becomes larger and more body parts of the subject are occluded with the movement of the hand. So we consider the reason for the missed detection may be that the area of occlusion is relatively small in first part of the video. It can also be seen from Figure 7(b) that the proposed occlusion correction algorithm completes the correction for areas with frequent changes in occlusion state.

The underlying assumption of our method is that, given a large number of videos with positive and negative frame-level labels, the neural network could automatically detect the occlusion frame after learning. Through experiments, our method achieves the highest occlusion detection accuracy of 84.0% and the highest AUC of 0.91 on the self-built dataset. However, according to the two misrecognized cases illustrated in Figure 7, there are some conflicts when the proposed model defines the occlusions in videos, which denote that the video shots occlusion detection task is very complex and remains further improvements.

6 CONCLUSIONS

In this paper, we propose occlusion detection method for automatic video editing to optimize the HCI procedure of vidoe editors. To verify the effectiveness of the proposed method, we introduce a new large scale occlusion detection dataset. The experimental results on this dataset illustrate that the proposed occlusion detection algorithm is significantly better than other baseline methods. In future work, we will keep improving the performance of the occlusion detection algorithm and attempt to locate the occlusion for further processing like video inpainting.

ACKNOWLEDGMENTS

This work is supported in part by the Science and Technology Department of Sichuan Province under Grant, No.:2019YFG0184; in part by the Foundation of Science and Technology on Communication Security Laboratory under Grant, No.:6142103190415; in part by Project 61902333 supported by National Natural Science Foundation of China, by the Shenzhen Institute of Artificial Intelligence and Robotics for Society (AIRS); and in part by Project 61902267 supported by National Natural Science Foundation of China.

REFERENCES

[1] Alper Ayvaci, Michalis Raptis, and Stefano Soatto. 2012. Sparse occlusion detection with optical flow. *International journal of computer vision* 97, 3 (2012), 322–338.

[2] Sunghyun Cho, Jue Wang, and Seungyong Lee. 2012. Video deblurring for hand-held cameras using patch-based synthesis. *ACM Transactions on Graphics (TOG)* 31, 4 (2012), 1–9.

[3] Navneet Dalal and Bill Triggs. 2005. Histograms of oriented gradients for human detection. In *2005 IEEE computer society conference on computer vision and pattern recognition (CVPR'05)*, Vol. 1. IEEE, 886–893.

[4] Yi Deng, Qiong Yang, Xueyin Lin, and Xiaoou Tang. 2007. Stereo correspondence with occlusion handling in a symmetric patch-based graph-cuts model. *IEEE transactions on pattern analysis and machine intelligence* 29, 6 (2007), 1068–1079.

[5] Piotr Dollar, Christian Wojek, Bernt Schiele, and Pietro Perona. 2011. Pedestrian detection: An evaluation of the state of the art. *IEEE transactions on pattern analysis and machine intelligence* 34, 4 (2011), 743–761.

[6] Mark Everingham, Luc Van Gool, Christopher KI Williams, John Winn, and Andrew Zisserman. 2007. The PASCAL visual object classes challenge 2007 (VOC2007) results. (2007).

[7] Pedro F Felzenszwalb, Ross B Girshick, David McAllester, and Deva Ramanan. 2009. Object detection with discriminatively trained part-based models. *IEEE transactions on pattern analysis and machine intelligence* 32, 9 (2009), 1627–1645.

[8] Fabio Galasso, Naveen Shankar Nagaraja, Tatiana Jimenez Cardenas, Thomas Brox, and Bernt Schiele. 2013. A unified video segmentation benchmark: Annotation, metrics and analysis. In *Proceedings of the IEEE International Conference on Computer Vision*. 3527–3534.

[9] Andreas Geiger, Philip Lenz, and Raquel Urtasun. 2012. Are we ready for autonomous driving? the kitti vision benchmark suite. In *2012 IEEE Conference on Computer Vision and Pattern Recognition*. IEEE, 3354–3361.

[10] Ross Girshick. 2015. Fast r-cnn. In *Proceedings of the IEEE international conference on computer vision*. 1440–1448.

[11] 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.

[12] 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.

[13] 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 Conference on Computer Vision and Pattern Recognition*. 7183–7192.

[14] Edward Hsiao and Martial Hebert. 2014. Occlusion reasoning for object detectionunder arbitrary viewpoint. *IEEE transactions on pattern analysis and machine intelligence* 36, 9 (2014), 1803–1815.

[15] 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 (CVPR)*.

[16] 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.

[17] 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.

[18] Neel Joshi, Wolf Kienzle, Mike Toelle, Matt Uyttendaele, and Michael F Cohen. 2015. Real-time hyperlapse creation via optimal frame selection. *ACM Transactions on Graphics (TOG)* 34, 4 (2015), 1–9.

[19] Sing Bing Kang, Richard Szeliski, and Jinxiang Chai. 2001. Handling occlusions in dense multi-view stereo. In *Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition. CVPR 2001*, Vol. 1. IEEE, I–I.

[20] 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.

[21] Dieter Koller, Joseph Weber, and Jitendra Malik. 1994. Robust multiple car tracking with occlusion reasoning. In *European Conference on Computer Vision*. Springer, 189–196.

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

[23] Jinde Liu, Kaiqi Huang, and Tieniu Tan. 2015. Learning occlusion patterns using semantic phrases for object detection. In *2015 IEEE International Conference on Image Processing (ICIP)*. IEEE, 686–690.

[24] 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.

[25] Xue Mei, Haibin Ling, Yi Wu, Erik P Blasch, and Li Bai. 2013. Efficient minimum error bounded particle resampling L1 tracker with occlusion detection. *IEEE Transactions on Image Processing* 22, 7 (2013), 2661–2675.

[26] 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.

[27] Wanli Ouyang and Xiaogang Wang. 2013. Joint deep learning for pedestrian detection. In *Proceedings of the IEEE International Conference on Computer Vision*. 2056–2063.

[28] Bojan Pepikj, Michael Stark, Peter Gehler, and Bernt Schiele. 2013. Occlusion patterns for object class detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 3286–3293.

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

[30] Jian Sun, Yin Li, Sing Bing Kang, and Heung-Yeung Shum. 2005. Symmetric stereo matching for occlusion handling. In *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, Vol. 2. IEEE, 399–406.

[31] 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.

[32] 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.

[33] 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.

[34] 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.

[35] 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.

[36] Jialiang Wang and Todd Zickler. 2019. Local detection of stereo occlusion boundaries. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 3818–3827.

[37] 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 (TOG)* 38, 6 (2019), 1–13.

[38] Xiaolong Wang, Ross Girshick, Abhinav Gupta, and Kaiming He. 2018. Non-local neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 7794–7803.

[39] Xiaoyu Wang, Tony X Han, and Shuicheng Yan. 2009. An HOG-LBP human detector with partial occlusion handling. In *2009 IEEE 12th international conference on computer vision*. IEEE, 32–39.

[40] 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.

[41] Zhengyang Wu, Fuxin Li, Rahul Sukthankar, and James M Rehg. 2015. Robust video segment proposals with painless occlusion handling. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 4194–4203.

[42] Wei-Qi Yan and Mohan S Kankanhalli. 2002. Detection and removal of lighting & shaking artifacts in home videos. In *Proceedings of the tenth ACM international conference on Multimedia*. 107–116.

[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] Congxuan Zhang, Zhen Chen, Mingrun Wang, Ming Li, and Shaofeng Jiang. 2017. Robust non-local TV-L1 optical flow estimation with occlusion detection. *IEEE Transactions on Image Processing* 26, 8 (2017), 4055–4067.

[45] Shifeng Zhang, Longyin Wen, Xiao Bian, Zhen Lei, and Stan Z Li. 2018. Occlusion-aware R-CNN: detecting pedestrians in a crowd. In *Proceedings of the European Conference on Computer Vision (ECCV)*. 637–653.

[46] Chunluan Zhou and Junsong Yuan. 2017. Multi-label learning of part detectors for heavily occluded pedestrian detection. In *Proceedings of the IEEE International Conference on Computer Vision*. 3486–3495.

[47] 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.

[48] C Lawrence Zitnick and Takeo Kanade. 2000. A cooperative algorithm for stereo matching and occlusion detection. *IEEE Transactions on pattern analysis and machine intelligence* 22, 7 (2000), 675–684.