

Cube Padding for Weakly-Supervised Saliency Prediction in 360° Videos

Hsien-Tzu Cheng¹, Chun-Hung Chao¹, Jin-Dong Dong¹, Hao-Kai Wen², Tyng-Luh Liu³, Min Sun¹

¹National Tsing Hua University ²Taiwan AI Labs ³Academia Sinica

hsientzucheng@gapp.nthu.edu.tw {raul.c.chao, mark840205}@gmail.com

hao.kai@ailabs.tw liutyng@iis.sinica.edu.tw sunmin@ee.nthu.edu.tw

Abstract

Automatic saliency prediction in 360° videos is critical for viewpoint guidance applications (e.g., Facebook 360 Guide). We propose a spatial-temporal network which is (1) weakly-supervised trained and (2) tailor-made for 360° viewing sphere. Note that most existing methods are less scalable since they rely on annotated saliency map for training. Most importantly, they convert 360° sphere to 2D images (e.g., a single equirectangular image or multiple separate Normal Field-of-View (NFoV) images) which introduces distortion and image boundaries. In contrast, we propose a simple and effective Cube Padding (CP) technique as follows. Firstly, we render the 360° view on six faces of a cube using perspective projection. Thus, it introduces very little distortion. Then, we concatenate all six faces while utilizing the connectivity between faces on the cube for image padding (i.e., Cube Padding) in convolution, pooling, convolutional LSTM layers. In this way, CP introduces no image boundary while being applicable to almost all Convolutional Neural Network (CNN) structures. To evaluate our method, we propose Wild-360, a new 360° video saliency dataset, containing challenging videos with saliency heatmap annotations. In experiments, our method outperforms baseline methods in both speed and quality.

1. Introduction

The power of 360° camera is to capture the entire viewing sphere (referred to as sphere for simplicity) surrounding its optical center, providing a complete picture of the visual world. This ability goes beyond the traditional perspective camera and the human visual system which both have a limited Field of View (FoV). Videos captured using 360° camera (referred to as 360° videos) are expected to have a great impact in domains like virtual reality (VR), autonomous robots, surveillance systems in the near future. For now, 360° videos already gained its popularity thanks to low-cost hardware on the market, and supports of video streaming on YouTube and Facebook.

Despite the immersive experience and complete view-

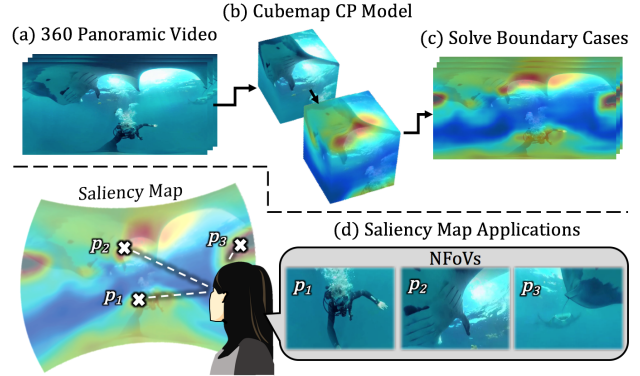


Figure 1. Saliency prediction in a 360° video. Panel (a) shows a challenging frame in equirectangular projection with two marine creatures. One is near the north polar and the other is near the horizontal boundary. Panel (b) shows that Cubemap projection with Cube Padding (CP) mitigate distortion and cuts at image boundaries. As a result, we predict high-quality saliency map on the Cubemap. In panel (c), when visualizing our predicted saliency map on equirectangular, both marine creatures are recalled. In panel (d), desirable Normal Field of Views (NFoVs) are obtained from high-quality saliency map.

point selection freedom provided by 360° videos, many works recently show that it is important to guide viewers' attention. [22, 27, 52, 51] focus on selecting the optimal viewing trajectory in a 360° video so that viewers can watch the video in Normal FoV (NFoV). [30, 29] focus on providing various visual guidance in VR display so that the viewers are aware of all salient regions. Most recently, Chou *et al.* [10] propose to guide viewers' attention according to the scripts in a narrated video such as a tour guide video. Yu *et al.* [64] propose to generate a highlight video according to spatial-temporal saliency in a 360° video. All methods above involve predicting or require the existence of spatial-temporal saliency map in a 360° video.

Existing methods face two challenges in order to predict saliency on 360° videos. Firstly, 360° videos capture the world in a wider variety of viewing angles compared to videos with an NFoV. Hence, existing image [9, 24]

or video [37] saliency datasets are not ideal for training saliency prediction model. One way to overcome this challenge is to collect saliency dataset directly on 360° videos. In this direction, Facebook [1] and Youtube [2] are collecting users’ viewing history on 360° videos. However, there is a chicken and egg problem. Without the ability to predict saliency and provide attention guidance, users will only be interested in viewing some popular 360° videos. Moreover, these data are proprietary which are not publicly available to the research community.

Secondly, most existing methods [22, 27, 52, 51] apply their techniques to process images with the equirectangular projection (referred to as equirectangular images). However, equirectangular images introduce image boundaries, and create distortion significantly at the top and bottom regions. Both of them lead to prediction artifacts and make the learning task harder. An alternative is to divide the 360° sphere into multiple but “separate” perspective images. Although it avoids distortion, it will introduce more image boundaries. We can also divide the 360° sphere into multiple “overlapping” perspective images. Then, we only take the saliency prediction in a center sub-region in order to combine all predictions onto the whole sphere. However, this will require many more perspective images and significantly slow down the prediction process. Recently, Su and Grauman [50] propose a new training procedure and a new network structure (i.e., spherical convolution) to overcome this challenge. We argue that a simpler modification on existing CNN can overcome this challenge.

In this work, we propose a spatial-temporal network consisting of a static model and a ConvLSTM module to form a temporal model. The static model is inspired by [66] which computes a class-wise activation map per image. We remove the global average pooling and convert the last fully connected layer into a 1-by-1 convolutional layer to obtain a static feature per image. Note that this static model is weakly-supervised trained by only monocular image level supervision, i.e. without 360° videos. After that, static features at several timesteps are fed into a ConvLSTM [63] module to aggregate temporal information. Our temporal model is also designed to be trained in an unsupervised manner. During training, our loss function mainly enforces temporal consistency on two consecutive predicted saliency maps given precomputed optical flow. Note that the ConvLSTM module and temporal consistency loss encourage the predicted saliency map to be temporally smooth and motion-aware.

Most importantly, our model is tailor-made for 360° videos (see Fig. 1). Firstly, we project the 360° sphere on six faces of a cube, which introduces very little distortion. Then, we concatenate all six faces as an input while utilizing the connectivity between faces on the cube for image padding (referred to as Cube Padding (CP)) in convolution,

pooling, convolutional LSTM layers. In this way, CP introduces no image boundary while utilizing existing CNN layers. To evaluate our method, we propose Wild-360, a new 360° video saliency dataset, containing challenging videos with saliency heatmap annotations. According to experimental results, our method outperforms all baselines in both speed and quality.

We summarize our contributions as below:

1. We propose an **weakly-supervised** trained spatial-temporal saliency prediction model. This ensures that our approach is scalable in overcoming large viewpoint variation in 360° videos. To the best of our knowledge, it is the first method to tackle the 360° video saliency map prediction in an weakly-supervised manner.
2. We introduce Cube Padding tailor-made for 360° videos to mitigate **distortion and image boundaries**. This module is fast, effective, and generally applicable to almost all existing CNN architectures.
3. We collect a **new Wild-360 dataset** with challenging 360 videos. One-third of our dataset is annotated with per-frame saliency heatmap for evaluation. Similar to [1, 2], we collect heatmap by aggregating viewers’ trajectories, consisting of 80 viewpoints per-frame.
4. Experimental results show that our method outperforms baseline methods both in **speed and quality**.

2. Related work

To better comprehend the proposed method and the potential contributions, we discuss the recent developments of relevant techniques, including saliency map prediction, localization via weakly-supervised or unsupervised learning, and 360° vision.

Saliency map prediction. Predicting where humans look in an image has been a popular task in computer vision. [32, 19, 4, 57, 65, 44] focus on detecting salient regions in images. [31, 23, 11, 43, 42, 8, 60, 59, 53] employ deep learning to achieve much better results. For videos, [12, 18, 33, 48, 37, 28] rely on low-level appearance and motion cues as inputs. In addition, [26, 17, 47, 34, 15] consider information such as face, people, objects, *etc.* However, all these approaches demand heavy saliency supervision while our method requires no manual saliency annotations.

Weakly-supervised localization. Recent techniques typically leverage the power of CNNs to localize the targets in an image, where the CNNs are only trained with image-level labels. The approach in [41] designs a Global Max Pooling (GMP) layer to carry out object localization by activating discriminative parts of objects. Subsequently, Zhou *et al.* [66] propose Global Average Pooling (GAP) to achieve a much better result on activating the object regions. [58, 14, 5] instead consider using other pooling layers. Our method treats the deep features from the last convolutional layer, encoded with objectness clues, as saliency features for further processing. Having obtained the spatial saliency

maps by selecting maximum per-pixel responses, we can then use these spatial heatmaps to learn or predict temporal saliency. More recently, Hsu *et al.* [21] develop two coupled ConvNets, one for image-level classifier and the other for pixel-level generator. By designing a well-formulated loss function and top-down guidance from class labels, the generator is demonstrated to output saliency estimation of good quality.

Unsupervised localization. One of the popular schemes for designing unsupervised deep-learning model is to train the underlying DNN with respect to the reconstruction loss. The reconstruction loss between an input and a warped image can be used for optical flow estimation [67] and for single-view depth estimation [16]. Turning now our attention to the unsupervised learning methods for video object segmentation, the two-stream neural network with visual memory by Tokmakov *et al.* [55] is the current state-of-the-art for the benchmark, DAVIS [45]. They generalize the popular two-stream architecture with ConvGRU [7] to achieve the good performance. Although the network architecture of our method is not two-stream, it does explore the two-stream information sequentially, as shown in Figure 2. That is, the ConvLSTM [63] adopted in our approach is used to learn how to combine both spatial and temporal (including motion) information. While both [55] and our work use self-supervision from video dynamics, we specifically focus on developing a general technique to solve the pole distortion and boundary discontinuity in processing 360° videos.

360° Video. Different from the conventional, 360° videos bring in a whole distinct viewing experience with immersive content. The new way of recording yields, in essence, a spherical video that allows the users to choose the viewing directions for abundant scenarios as if they were in the center of filming environment. In particular, techniques related to virtual cinematography are introduced in [52, 51, 22, 27] to guide the user to make the FoV selection when viewing a 360° video. Nevertheless, such a strategy targets selecting a specific FoV and eliminates most of the rich content in a 360° video, while our proposed model generates a saliency map to activate multiple regions of interest. Indeed only a few attempts for estimating the saliency information in 360° videos have been made. The work by Monroy *et al.* [39] is the first to tackle this problem. To generate a saliency map for a 360° spherical patch, their method computes the corresponding 2D perspective image, and detect the saliency map using model pre-trained on SALICON dataset. Taking account of where the spherical patch is located at, the final result of saliency detection can be obtained by refining the 2D saliency map. However, defects due to the image boundaries are not explicitly handled. In SALTINET [6], Assens *et al.* propose to predict scan-path of a 360° image with heavy manual annotations. Unlike our approach, these

methods all require strong supervision.

Dataset. One of the main contributions of our work is the effort to establish a new **Wild-360** dataset. We thus briefly describe the current status of major collections relevant to (our) 360° video analysis. The MIT300 [9] includes 300 benchmark images of indoor or outdoor scenes, collected from 39 observers using an eye tracker. It also comes with AUC-Judd and AUC-Borji evaluation metrics, which are adopted in our work. SALICON [24] has 10,000 annotations on MS COCO images, collected by a mouse-contingent multi-resolution paradigm based on neurophysiological and psychophysical studies of peripheral vision to simulate the natural viewing behavior of humans. The Densely-Annotated Video Segmentation (DAVIS) [45] is a new dataset with 50 high-resolution image sequences with all their frames annotated with pixel-level object masks. DIEM [38] has, by far, collected data from over 250 participants watching 85 different videos, and the fixations are reported with respect to the user’s gaze. Finally, the Freiburg-Berkeley Motion Segmentation Dataset [40] comprises a total of 720 frames, annotated with pixel-accurate segmentation annotation of moving objects. However, none of the datasets motioned above provides ground truth saliency map annotation on 360° videos to evaluate our proposed method.

3. Our method

In this section, we present our overall method as shown in Fig. 2, which consists of projection processes, static model, temporal model and loss functions. We describe Cube Padding and potential impacts in Sec. 3.2, our static model in Sec. 3.3, temporal model in Sec. 3.4. Before that, we first introduce the various notations used in our formulation.

3.1. Notations

Given a 360° equirectangular 2D map $\mathcal{M} \in \mathbb{R}^{c \times q \times p}$ with the number of channels c , width p and height q , we define a projection function P to transform \mathcal{M} to a cube-map representation $M \in \mathbb{R}^{6 \times c \times w \times w}$ with the edge length of the cube set to w . Specifically, M is a stack of 6 faces $\{M^B, M^D, M^F, M^L, M^R, M^T\}$, where each face $M^j \in \mathbb{R}^{c \times w \times w}$, and $j \in \{B, D, F, L, R, T\}$ represents the *Back*, *Down*, *Front*, *Left*, *Right*, and *Top* face, respectively. We can further inverse transform M back to \mathcal{M} by $\mathcal{M} = P^{-1}(M)$. Note that a RGB equirectangular image \mathcal{I} is, in fact, a special 2D map where $c = 3$ and $I \in \mathbb{R}^{6 \times 3 \times w \times w}$ is a special cubemap with RGB value. For details of the projection function P please refer to the supplementary material.

3.2. Cube padding

Traditionally, Zero Padding (ZP) is applied at many layers in a Convolutional Neural Network (CNN) such as convolution and pooling. However, in our case, M consists of 6 2D faces in a batch, observing the whole 360° viewing

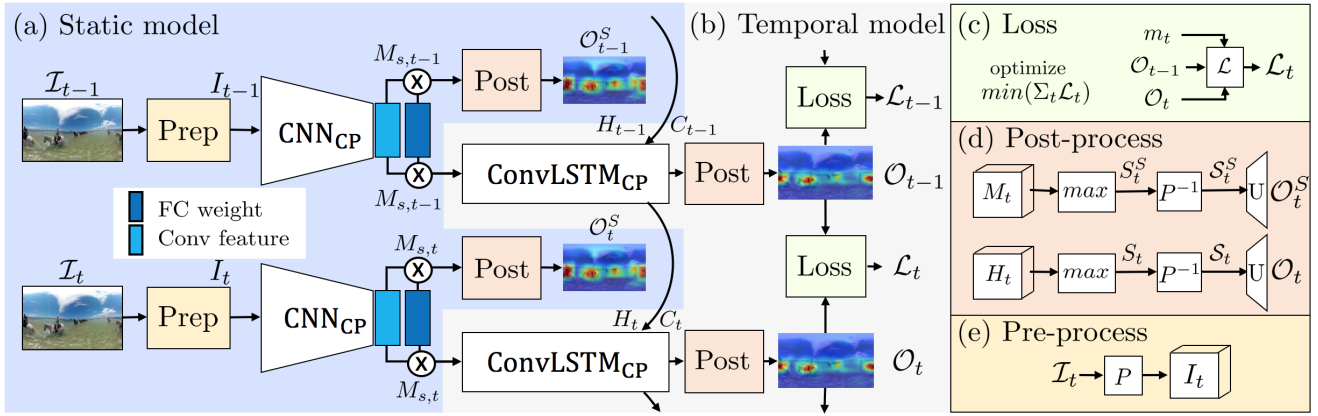


Figure 2. Visualization of our system. Panel (a) shows our static model: (1) the pre-process to project an equirectangular image \mathcal{I} to a cubemap image I , (2) the CNN with Cube Padding (CP) to extract a saliency feature M_s , (3) the post-process to convert M_s into an equirectangular saliency map \mathcal{O}^S . Panel (b) shows our temporal model: (1) the convLSTM with CP to aggregate the saliency feature M_s through time into H , (2) the post-process to convert H into an equirectangular saliency map \mathcal{O} , (3) our self-supervised loss function to compute \mathcal{L}_t given current \mathcal{O}_t and previous \mathcal{O}_{t-1} . Panel (c) shows the total loss to be minimized. Panel (d) shows the post-process module including a max-pooling, inverse projection (P^{-1}), and upsampling (U). Panel (e) shows the pre-processing module with cubemap projection.

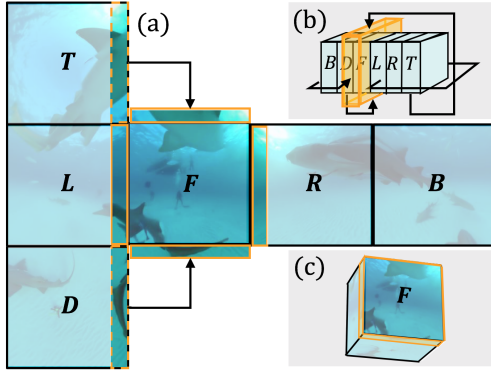


Figure 3. Illustration of Cube Padding (CP). In panel (a), we apply CP for the face F which leverages information (in yellow rectangles) on face T, L, R, D naturally rather than padding with zero values (i.e., zero padding). Panel (b) shows that this can be done in cubemap matrix representation $M \in \mathbb{R}^{6 \times c \times w \times w}$. Panel (c) shows how to fold the faces back to a cube.

sphere. If we put M to normal architecture with ZP in every single layer, the receptive field will be restricted inside each face, separating 360° contents into 6 non-connected fields. To solve this problem, we use Cube Padding (CP) to enable neurons to see across multiple faces by the interconnection between different faces in M . For an input M , CP takes the adjacent regions from the neighbor faces and concatenate them to the target face to produce a padded feature map. Fig. 3 illustrates a case of target face M^F which is adjacent with M^R, M^T, M^L and M^D . CP then simply considers the corresponding pads as shown in yellow patches in Fig. 3 outside M^F , where these pads are concatenated with M^F . Panel (a) in Fig. 3 illustrates that the yellow CP patch on the cubemap in 3D is visually similar to padding

on sphere. Panel (b) shows the padding directions of M^F in M batch.

Although the padding size of CP is usually small, e.g. only 1 pixel for kernel size=3 and stride=1, by propagating M through multiple layers incorporated with CP, the receptive field will gradually become large enough to cover contents across nearby faces. Fig. 4 illustrates some responses of deep features from CP and ZP. While ZP fails to have responses near the face boundaries, CP enables our model to recognize patterns of an object across faces.

To sum up, Cube Padding (CP) has following advantages: (1) applicable to most kinds of layers in CNN (2) the CP generated features are trainable to learn 360° spatial correlation across multiple cube faces, (3) CP preserves the receptive field of neurons across 360° content without the need for additional resolution.

3.3. Static model

For each frame \mathcal{I} of an input video sequence, our static model feeds preprocessed I into the CNN. As shown in panel (a) of Fig. 2, CP module is incorporated in every convolutional and pooling layers in our CNN. The static model output M_S is obtained by multiplying the feature map M_ℓ generated from the last convolutional layer with the weight of the fully connected layer W_{fc} .

$$M_S = M_\ell * W_{fc} \quad (1)$$

where $M_S \in \mathbb{R}^{6 \times K \times w \times w}$, $M_\ell \in \mathbb{R}^{6 \times c \times w \times w}$, $W_{fc} \in \mathbb{R}^{c \times K \times 1 \times 1}$, c is the number of channels, w is corresponding feature width, “*” means the convolution operation and K is the number of classes for a model pre-trained on a specific classification dataset. To generate a static saliency map S , we simply pixel-wisely select the maximum value in M_S

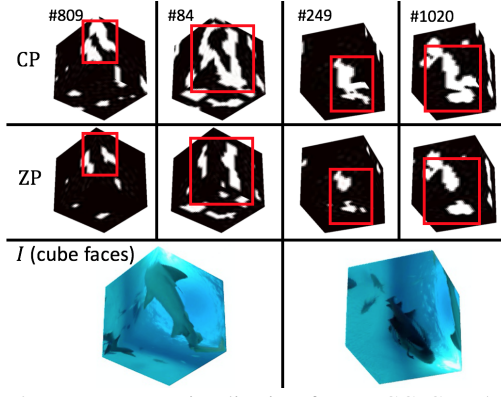


Figure 4. Feature map visualization from VGG Conv5_3 layer. When Cube Padding (CP) is used (the first row), the response continuous through the face boundaries. However, when Zero Padding (ZP) is used (the second row), the responses near the boundaries vanished since each face is processed locally and separately. The last row shows the corresponding cubemap images containing several marine creatures across face boundaries.

along the class dimension.

$$S^j(x, y) = \max_k \{M_S^j(k, x, y)\} ; \forall j \in \{B, D, F, L, R, T\}, \quad (2)$$

where $S^j(x, y)$ is the saliency score at location (x, y) of cube face j , and the saliency map in equirectangular projection \mathcal{S} can be obtained with $\mathcal{S} = P^{-1}(S)$. To get the final equirectangular output, we upsample \mathcal{S} to \mathcal{O} as shown in Fig. 2 panel (d).

3.4. Temporal model

Convolutional LSTM. Motivated by studies [46, 35, 36], human beings tend to put their attention on moving objects and changing scenes rather than static, we design our temporal model to capture dynamic saliency in a video sequence. As shown in the light gray block in Fig. 2, we use ConvLSTM as our temporal model, a recurrent model for spatio-temporal sequence modeling using 2D-grid convolution to leverage the spatial correlations in input data, which has been successfully applied to precipitation nowcasting [63] task. The ConvLSTM equations are given by

$$\begin{aligned} i_t &= \sigma(W_{xi} * M_{S,t} + W_{hi} * H_{t-1} + W_{ci} \circ C_{t-1} + b_i) \\ f_t &= \sigma(W_{xf} * M_{S,t} + W_{hf} * H_{t-1} + W_{cf} \circ C_{t-1} + b_f) \\ g_t &= \tanh(W_{xc} * X_t + W_{hc} * H_{t-1} + b_c) \\ C_t &= i_t \circ g_t + f_t \circ C_{t-1} \\ o_t &= \sigma(W_{xo} * M_t + W_{ho} * H_{t-1} + W_{co} \circ C_t + b_o) \\ H_t &= o_t \circ \tanh(C_t), \end{aligned} \quad (3)$$

where \circ denotes the element-wise multiplication, $\sigma(\cdot)$ is the sigmoid function, all W_* and b_* are model parameters to be learned, i, f, o are the input, forget, and output control signals with value $[0, 1]$, g is the transformed input signal with value $[-1, 1]$, C is the memory cell value, $H \in R^{6 \times K \times w \times w}$ is the hidden representation as both the output and the recurrent input, M_S is the output of the static

model (see Eq. (1)), t is the time index which can be used in subscript to indicate timesteps.

We generate saliency map from H_t equivalent to Eq. (2).

$$S_t^j(x, y) = \max_k \{H_t^j(k, x, y)\} ; \forall j \in \{B, D, F, L, R, T\}, \quad (4)$$

where $S_t^j(x, y)$ is the generated saliency score at location (x, y) of cube face j at time step t . Similar to our static model, we upsample \mathcal{S} to \mathcal{O} to get the final equirectangular output.

Temporal consistent loss. Inspired by [21, 67, 16] that model correlation between discrete images in an self-supervised manner by per-pixel displacement warping, smoothness regularization, etc., we design 3 loss functions to train our model and refine O_t by temporal constraints: temporal reconstruction loss \mathcal{L}_t^{recons} , smoothness loss \mathcal{L}_t^{smooth} , and motion masking loss \mathcal{L}_t^{motion} . The total loss function of each time step t can be formulated as:

$$\mathcal{L}_t^{total} = \lambda_r \mathcal{L}_t^{recons} + \lambda_s \mathcal{L}_t^{smooth} + \lambda_m \mathcal{L}_t^{motion} \quad (5)$$

In the following equations, i.e. Eqs. (6)–(9), N stands for the number of pixels along spatial dimensions of one feature map, $O_t(p)$ is the output at pixel position p at time step t , and m is optical flow by [62]. \mathcal{L}_t^{recons} is computed as the photometric error between the true current frame O_t and the warped last frame $O_{t-1}(p + m)$:

$$\mathcal{L}_t^{recons} = \frac{1}{N} \sum ||O_t(p) - O_{t-1}(p + m)||^2 \quad (6)$$

The reconstruction loss is formed by an assumption: the same pixel across different short-term time step should have a similar saliency score. This term helps to refine the saliency map to be more consistent in patches i.e. objects with similar motion patterns. \mathcal{L}_t^{smooth} is computed by the current frame and the last frame as:

$$\mathcal{L}_t^{smooth} = \frac{1}{N} \sum ||O_t(p) - O_{t-1}(p)||^2 \quad (7)$$

The smoothness term is used to constrain the nearby frames to have a similar response without large changes. It also restrains the other 2 terms with motion included, since the flow could be noisy or drifting. \mathcal{L}_t^{motion} is used for motion masking:

$$\mathcal{L}_t^{motion} = \frac{1}{N} \sum ||O_t(p) - O_t^m(p)||^2 \quad (8)$$

$$O_t^m = \begin{cases} 0, & \text{if } |m(p)| \leq \epsilon; \\ O_t(p), & \text{elsewhere.} \end{cases} \quad (9)$$

We set ϵ in Eq. (9) as a small margin to eliminate the pixel response where motion magnitude lowers than ϵ . If a pattern in a video remains steady for several time steps, it is intuitively that the video saliency score of these non-moving pixels should be lower than changing patches.

For sequence length of ConvLSTM set to Z , the aggregated loss will be $\mathcal{L}^{total} = \sum^Z \mathcal{L}_t^{total}$. By optimizing our model with these loss functions jointly to \mathcal{L}^{total} throughout the sequence, we can get the final saliency result by considering temporal patterns through Z frames.

4. Dataset

For the purpose of testing and benchmarking saliency prediction on 360° videos, a first and freshly collected dataset named Wild-360 is presented in our work. Wild-360 contains 85 360° video clips, totally about 55k frames. 60 clips within our dataset are for training and the rest 25 clips are for testing. All the clips are cleaned and trimmed from 45 raw videos obtained from YouTube. We manually select raw videos from keywords “Nature”, “Wildlife”, and “Animals”; these keywords were selected in order to get videos with the following aspects: (i) sufficiently large number of search results of 360° video on YouTube, (ii) multiple salient objects in a single frame with diverse categories, (iii) dynamic contents inside the videos to appear in regions of any viewing angles including polar and borders. The Wild-360 dataset is also designed to be diverse in object presence and free from the systematic bias. We rotate each testing video in both longitude and latitude angle to prevent the center-bias in ground truth saliency.

Recently, [1, 2] both announced to collect saliency heatmap of 360° videos by aggregating the viewers’ trajectories during manipulation with view ports. To adopt the similar approach, but also giving the global perspective to viewers to easily capture multiple salient regions without missing hot spots, we adopt HumanEdit interface from [52]. HumanEdit, as the Wild-360 labeling platform, encourages labelers to directly record attention trajectories based on their intuition. 30 labelers were recruited to label the videos in testing set, and they were asked to annotate from several viewing angles $\psi \in \{0^\circ, 90^\circ, 180^\circ\}$. Therefore, there are about totally 80 viewpoints in a single frame. During annotation, videos and 3 rotation angles are shuffled to avoid order effect. In this setting, various positions could be marked as salient regions. Similar to [54], we further apply Gaussian mask to every viewpoint to get aggregated saliency heatmap. Typical frames with ground truth heatmap (GT) are shown in the supplementary material. In order to foster future research related to saliency prediction in 360 videos, we plan to release the dataset, once the paper is published.

5. Experiments

We compare our saliency prediction accuracy and speed performance with many baseline methods. In the following, we first give the implementation details. Then, we describe the baseline methods and evaluation metric. Finally, we report the performance comparison.

5.1. Implementation details

We use ResNet-50 [20] and VGG-16 [49] pretrained on ImageNet [13] to construct our static model. For temporal model, we set Z of ConvLSTM to 5 and train it for 1 epoch with ADAM optimizer and learning rate 10^{-6} . We set the hyperparameters of temporal loss function to balance each term for steady loss decay. We set $\lambda_r = 0.1$, $\lambda_s = 0.7$, $\lambda_t = 0.001$. To measure the computational cost and quality performance of different settings, we set $w = 0.25p$, where w and p is the width of the cubemap and equirectangular image respectively. Moreover, the width of the equirectangular is 2 times the height of the equirectangular image, $q = 0.5p$. This setting is equivalent to [3] and fixes the total area ratio between cubemap and equirectangular image to 0.75. We implement all the padding mechanism rather than using built-in backend padding for fair comparison.

To generate ground truth saliency map of Wild-360, referring to [54] and heatmap providers [1], the saliency distribution was modeled by aggregating viewpoint-centered Gaussian kernels. We set $\sigma = 5$ to lay Gaussian inside the NFoV for invisible boundaries. To avoid the criterion being too loose, only locations on heatmap with value larger than $\mu + 3\sigma$ were considered “salient” when creating the binary mask for the saliency evaluation metrics, e.g. AUC.

5.2. Baseline methods

Our variants.

Equirectangular (EQUI) — We directly feed each equirectangular image in a 360° video to our static model.

Cubemap+ZP (Cubemap) — As Sec. 3.2 mentioned, our static model takes the six faces of the cube as an input to generate the saliency map. However, unlike CP, Zero Padding (ZP) is used by the network operations, i.e. convolution, pooling, which causes the loss of the continuity of the cube faces.

Overlap Cubemap+ZP (Overlap) — We set FoV = 120° so that each face overlaps with each other by 15°. This variant can be seen as a simplified version of CP that process with larger resolution to cover the content near the border of each cube face. Note that this variant has no interconnection between faces, which means only ZP is used.

EQUI + ConvLSTM — We feed each equirectangular image to our temporal model to measure how much better the temporal model improves over static model.

Existing methods.

Motion Magnitude — As Sec. 3.4 mentioned, most salient regions in our videos are non-stationary. Hence we directly use the normalized magnitude of [62] as saliency map to see how much motion clue contributes to video saliency.

Consistent Video Saliency — [61] detects salient regions in spatio-temporal structure based on the gradient flow and energy optimization. It was the state-of-the-art video saliency detection methods on SegTrack [56] and FBMS [40].

SalGAN — [42] proposed a Generative Adversarial Network (GAN) to generate saliency map prediction. SalGAN

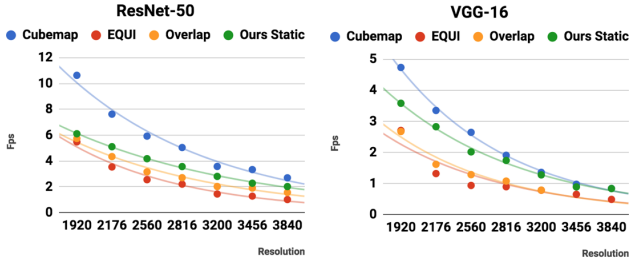


Figure 5. Speed of static methods. h-axis represents image resolution, v-axis represents FPS. As the resolution increase, the speed of Ours Static becomes closer to Cubemap. Besides, Ours Static exceeds EQUI and Overlap in FPS for all the tested resolutions.

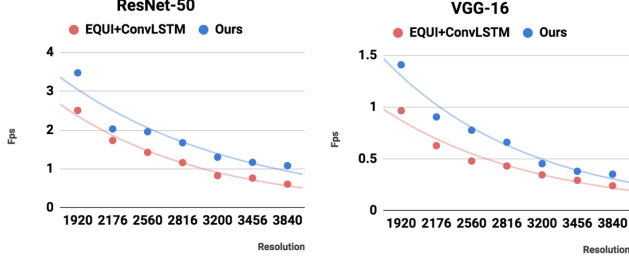


Figure 6. Speed of temporal methods. h-axis represents image resolution, v-axis represents FPS. Ours is faster than EQUI + ConvLSTM.

is the current state-of-the-art model on well-known traditional 2D saliency dataset SALICON [24] and MIT300 [9]. Note that this work focuses on saliency prediction on single image and needs ground truth annotations to do supervised learning. Hence, it cannot be trained on our dataset.

5.3. Computational efficiency

To compare the inference speed of our approach with other baselines with common resolution scale of 360° videos, we conduct an experiment to measure the Frame-Per-Second (FPS) along different resolutions. Fig. 5 shows the speed of static methods including Cubemap, EQUI, Overlap, and our static model (Ours Static). Fig. 6 shows the speed comparison between two methods using ConvLSTM: EQUI+ConvLSTM and our temporal model (Ours). The left and right side of both figures is for ResNet-50 and VGG-16, respectively. The resolutions are set from 1920 (Full HD) to 3840 (4K). The result of Fig 5 shows that Ours Static is slower than Cubemap but faster than Overlap and EQUI. Note that at the same amount of time, Ours Static has the ability to compute with a frame much larger than EQUI. Additionally, Fig. 6 shows that Ours is significantly faster than EQUI+ConvLSTM. We evaluate the computational efficiency on NVIDIA Tesla M40 GPU.

5.4. Evaluation metrics

We refer to the MIT Saliency Benchmark [9] and report three common metrics:

AUC-Judd (AUC-J). AUC-Judd [25] measures differences between our saliency prediction and the human labeled ground truth by calculating the true positive and false positive rate for the viewpoints.

<i>VGG-16</i>	CC	AUC-J	AUC-B
Cubemap	0.338	0.797	0.757
Overlap	0.380	0.836	0.813
EQUI	0.285	0.714	0.687
EQUI + ConvLSTM	0.330	0.823	0.771
Ours Static	0.381	0.825	0.797
Ours	0.383	0.863	0.843

<i>ResNet-50</i>	CC	AUC-J	AUC-B
Cubemap	0.413	0.855	0.836
Overlap	0.383	0.845	0.825
EQUI	0.331	0.778	0.741
EQUI + ConvLSTM	0.337	0.839	0.783
Ours Static	0.448	0.881	0.852
Ours	0.420	0.898	0.859

<i>Baselines</i>	CC	AUC-J	AUC-B
Motion Magnitude [62]	0.288	0.687	0.642
ConsistentVideoSal [61]	0.085	0.547	0.532
SalGAN [42]	0.312	0.717	0.692

Table 1. Saliency prediction accuracy in CC, AUC-J, and AUC-B. Our methods (bold font) consistently outperform all baseline methods. See Sec. 5.2 for the baseline methods compared.

AUC-Borji (AUC-B). AUC-Borji score uniformly and randomly samples image pixels as negative and defines the saliency map values above the threshold at these pixels as false positives.

Linear Correlation Coefficient (CC). The linear correlation coefficient is a distribution based metric to measure the linear relationship of given saliency maps and the viewpoints. The coefficient value is bounded between -1 and 1, representing the linear dependencies between our output and ground truth.

5.5. Saliency comparison

From saliency comparison shown in Table. 1, we observe the following: (1) *Our temporal model is the best* in all evaluation metrics except one where our static model is better; (2) *ConvLSTM improves performance* since typically EQUI+ConvLSTM outperforms EQUI and Ours outperforms Ours Static. Typical examples are shown in Fig. 7, where we compare EQUI, Ours Static, with Ours. Our temporal model typically predicts smooth saliency map in time and is more effective to salient regions on image boundaries or in the top/bottom part of the image.

5.6. NFoV piloting

We use Wild-360 dataset and a set of Drone videos to demonstrate our result. Our scenario is to generate class specific NFoV trajectory. We use our per-frame saliency feature \mathcal{M} to get the score of each sampled viewing angle by average all the scores inside its corresponding NFoV on \mathcal{M} . Then \mathcal{O} is extracted by $\mathcal{M}(\hat{c})$, where \hat{c} is a class index decided by user. To link per-frame NFoV's to a trajectory

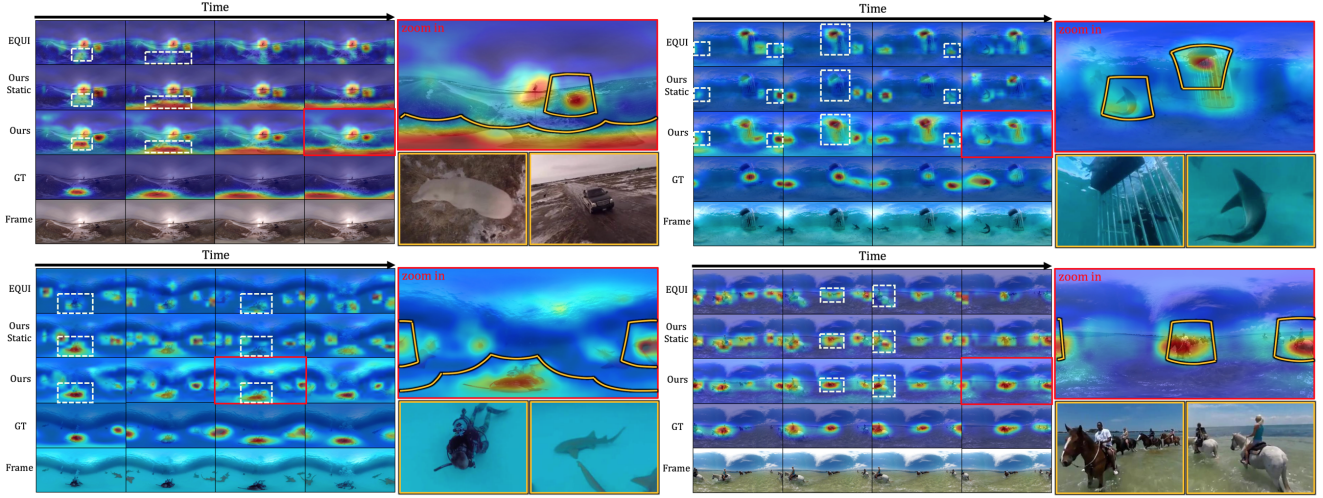


Figure 7. Qualitative examples. In each block, consecutive frames of various methods, ground truth, and raw videos are shown in the left panel. We highlight regions for comparison using white dash rectangles. In the right panel, one example is zoom-in (red box) and two salient NFOVs (yellow boxes) are rendered. Our temporal method (Ours) significantly outperforms others in overcoming distortion, image boundaries, and smoothness in time. See more examples in supplementary materials.

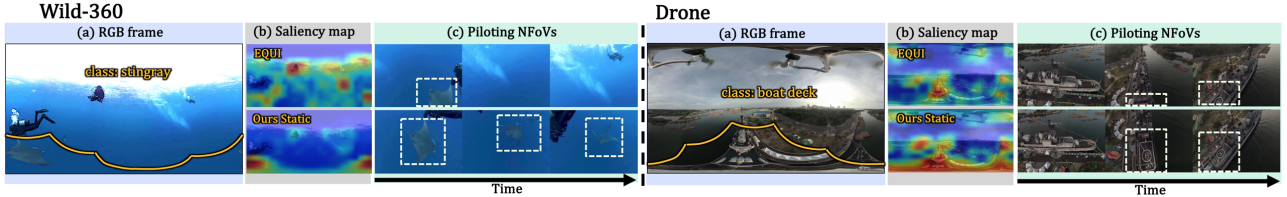


Figure 8. NFOV piloting results. The left and right sides show the result of Wild-360 dataset and Drone data, respectively. Panel (a) shows a video frame with targeted FoV drawn in orange. Panel (b) shows EQUI (top) and Ours Static (bottom) saliency maps. Panel (c) shows NFOVs in multiple time steps with white dash boxes indicating the targeted regions. NFOV Piloting with Ours Static in all 4 videos is more capable to capture the target regions.

Methods	win / loss
Ours Static vs. EQUI	95 / 65
Ours Static vs. Cubemap	97 / 63
Ours vs. Ours Static	134 / 26
Ours vs. GT	70 / 90

Table 2. Human evaluation results. We have 16 viewers watching 10 clips in each row so the number of wins and losses combined is 160. The result shows that Ours Static outperforms other static baselines; Ours outperforms Ours Static and is comparable to GT.

by saliency score, we use AUTOCAM [52] to find a feasible path of salient viewpoints. Fig. 8 shows that the NFOV tracks we generated are able to capture salient viewpoints better than equirectangular.

5.7. Human evaluation

We design 4 user tests and compare our methods with different image formats (Ours Statics vs. EQUI and Ours Static vs. Cubemap) and the performance of our temporal model (Ours vs. Ours Static and Ours vs. GT). Settings of the 4 tests are shown in Table. 2. In each test, we pick 10 different clips from the Wild-360 test set and generate saliency map prediction from 2 different methods. We ask

16 viewers to select the saliency map prediction which (1) activates on salient regions more correctly, (2) is smoother across frames. We further conduct a two-tailed binomial test and it shows that Ours Static is statistically superior to EQUI and Cubemap with p-value < 0.05 . This implies that Ours Static is a strong preference over using EQUI and Cubemap. Moreover, Ours significantly outperforms Ours Static with p-value < 0.01 . When compared with ground truth, the p-value is 0.13 which implies that the saliency map generated by Ours is comparable with ground truth.

6. Conclusion

We propose a novel spatial-temporal network which is (1) weakly-supervised trained without 360° supervision and (2) tailor-made for 360° viewing sphere, where a simple and effective Cube Padding (CP) technique is introduced. On a newly collected Wild-360 dataset with challenging videos and saliency heatmap annotations, our method outperforms state-of-the-art methods in both speed and quality.

Acknowledgements

We thank MOST-106-3114-E-007-008 and MediaTek for their support.

References

- [1] <https://www.facebook.com/facebookmedia/get-started/discovery-tools-insights>. 2, 6
- [2] <https://youtube-creators.googleblog.com/2017/06/hot-and-cold-heatmaps-in-vr.html>. 2, 6
- [3] <https://code.facebook.com/posts/1638767863078802/under-the-hood-building-360-video/>. 6
- [4] R. Achanta, S. S. Hemami, F. J. Estrada, and S. Ssstrunk. Frequency-tuned salient region detection. In *CVPR*, 2009. 2
- [5] X. Alameda-Pineda, A. Pilzer, D. Xu, E. Ricci, and N. Sebe. Viraliency: Pooling local virality. *CVPR*, 2017. 2
- [6] M. Assens, K. McGuinness, X. Giro, and N. E. O'Connor. Saltinet: Scan-path prediction on 360 degree images using saliency volumes. In *arXiv*, 2017. 3
- [7] N. Ballas, L. Yao, C. Pal, and A. Courville. Delving deeper into convolutional networks for learning video representations. *arXiv preprint arXiv:1511.06432*, 2015. 3
- [8] N. D. B. Bruce, C. Catton, and S. Janjic. A deeper look at saliency: Feature contrast, semantics, and beyond. In *CVPR*, 2016. 2
- [9] Z. Bylinskii, T. Judd, A. Borji, L. Itti, F. Durand, A. Oliva, and A. Torralba. Mit saliency benchmark. 1, 3, 7
- [10] S.-H. Chou, Y.-C. Chen, K.-H. Zeng, H.-N. Hu, J. Fu, and M. Sun. Self-view grounding given a narrated 360 video. In *AAAI*, 2018. 1
- [11] M. Cornia, L. Baraldi, G. Serra, and R. Cucchiara. A deep multi-level network for saliency prediction. In *ICPR*, 2016. 2
- [12] X. Cui, Q. Liu, and D. Metaxas. Temporal spectral residual: fast motion saliency detection. In *ACM Multimedia*, 2009. 2
- [13] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. Imagenet: A large-scale hierarchical image database. In *CVPR*, 2009. 6
- [14] T. Durand, T. Mordan, N. Thome, and M. Cord. Wildcat: Weakly supervised learning of deep convnets for image classification, pointwise localization and segmentation. 2017. 2
- [15] A. Fathi, Y. Li, and J. M. Rehg. Learning to recognize daily actions using gaze. In *ECCV*, 2012. 2
- [16] R. Garg, G. Carneiro, and I. Reid. Unsupervised cnn for single view depth estimation: Geometry to the rescue. In *ECCV*, 2016. 3, 5
- [17] S. Goferman, L. Zelnik-Manor, and A. Tal. Context-aware saliency detection. *TPAMI*, 2012. 2
- [18] C. Guo, Q. Ma, and L. Zhang. Spatio-temporal saliency detection using phase spectrum of quaternion fourier transform. In *CVPR*, 2008. 2
- [19] J. Harel, C. Koch, and P. Perona. Graph-based visual saliency. In *NIPS*, 2006. 2
- [20] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *CVPR*, 2016. 6
- [21] K.-J. Hsu, Y.-Y. Lin, and Y.-Y. Chuang. Weakly supervised saliency detection with a category-driven map generator. *BMVC*, 2017. 3, 5
- [22] H.-N. Hu, Y.-C. Lin, M.-Y. Liu, H.-T. Cheng, Y.-J. Chang, and M. Sun. Deep 360 pilot: Learning a deep agent for piloting through 360 degree sports video. *CVPR*, 2017. 1, 2, 3
- [23] S. Jetley, N. Murray, and E. Vig. End-to-end saliency mapping via probability distribution prediction. In *CVPR*, 2016. 2
- [24] M. Jiang, S. Huang, J. Duan, and Q. Zhao. Salicon: Saliency in context. In *CVPR*, 2015. 1, 3, 7
- [25] T. Judd, F. Durand, and A. Torralba. A benchmark of computational models of saliency to predict human fixations. In *MIT Technical Report*, 2012. 7
- [26] T. Judd, K. Ehinger, F. Durand, and A. Torralba. Learning to predict where humans look. In *ICCV*, 2009. 2
- [27] W. S. Lai, Y. Huang, N. Joshi, C. Buehler, M. H. Yang, and S. B. Kang. Semantic-driven generation of hyperlapse from 360 video. *TVC*, 2017. 1, 2, 3
- [28] T. Lee, M. Hwangbo, T. Alan, O. Tickoo, and R. Iyer. Low-complexity hog for efficient video saliency. In *ICIP*, 2015. 2
- [29] Y.-C. Lin, Y.-J. Chang, H.-N. Hu, H.-T. Cheng, C.-W. Huang, and M. Sun. Tell me where to look: Investigating ways for assisting focus in 360 video. In *CHI*, 2017. 1
- [30] Y.-T. Lin, Y.-C. Liao, S.-Y. Teng, Y.-J. Chung, L. Chan, and B.-Y. Chen. Outside-in: Visualizing out-of-sight regions-of-interest in a 360 video using spatial picture-in-picture previews. In *UIST*, 2017. 1
- [31] N. Liu and J. Han. Dhsnet: Deep hierarchical saliency network for salient object detection. In *CVPR*, 2016. 2
- [32] T. Liu, Z. Yuan, J. Sun, J. Wang, N. Zheng, X. Tang, and H.-Y. Shum. Learning to detect a salient object. *TPAMI*, 2011. 2
- [33] V. Mahadevan and N. Vasconcelos. Spatiotemporal saliency in dynamic scenes. *TPAMI*, 2010. 2
- [34] S. Mathe and C. Sminchisescu. Actions in the eye: Dynamic gaze datasets and learnt saliency models for visual recognition. *TPAMI*, 2015. 2
- [35] H. S. Meyerhoff, S. Schwan, and M. Huff. Interobject spacing explains the attentional bias toward interacting objects. *Psychonomic bulletin and review*, 2014. 5
- [36] H. S. Meyerhoff, S. Schwan, and M. Huff. Perceptual animacy: Visual search for chasing objects among distractors. *Journal of experimental psychology: human perception and performance*, 2014. 5
- [37] P. Mital, T. Smith, R. Hill, and J. Henderson. Clustering of gaze during dynamic scene viewing is predicted by motion. *Cognitive Computation*, 2011. 2
- [38] P. K. Mital, T. J. Smith, R. L. Hill, and J. M. Henderson. Clustering of gaze during dynamic scene viewing is predicted by motion. *Cognitive Computation*, 2011. 3
- [39] R. Monroy, S. Lutz, T. Chalasani, and A. Smolic. Salnet360: Saliency maps for omni-directional images with cnn. *arXiv preprint arXiv:1709.06505*, 2017. 3
- [40] P. Ochs, J. Malik, and T. Brox. Segmentation of moving objects by long term video analysis. *TPAMI*, 2014. 3, 6
- [41] M. Oquab, L. Bottou, I. Laptev, and J. Sivic. Is object localization for free?-weakly-supervised learning with convolutional neural networks. In *CVPR*, 2015. 2
- [42] J. Pan, C. Canton, K. McGuinness, N. E. O'Connor, J. Torres, E. Sayrol, and X. Giro-i Nieto. Salgan: Visual saliency prediction with generative adversarial networks. *arXiv preprint arXiv:1701.01081*, 2017. 2, 6, 7

- [43] J. Pan, K. McGuinness, E. Sayrol, N. O'Connor, and X. Giro-i Nieto. Shallow and deep convolutional networks for saliency prediction. In CVPR, 2016. 2
- [44] F. Perazzi, P. Krähenbühl, Y. Pritch, and A. Hornung. Saliency filters: Contrast based filtering for salient region detection. In CVPR, 2012. 2
- [45] J. Pont-Tuset, F. Perazzi, S. Caelles, P. Arbeláez, A. Sorkine-Hornung, and L. Van Gool. The 2017 davis challenge on video object segmentation. arXiv preprint arXiv:1704.00675, 2017. 3
- [46] J. Pratt, P. V. Radulescu, R. M. Guo, and R. A. Abrams. Its alive! animate motion captures visual attention. Psychological Science, 2010. 5
- [47] D. Rudoy, D. B. Goldman, E. Shechtman, and L. Zelnik-Manor. Learning video saliency from human gaze using candidate selection. In CVPR, 2013. 2
- [48] H. Seo and P. Milanfar. Static and space-time visual saliency detection by self-resemblance. Journal of Vision, 2009. 2
- [49] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. arXiv preprint arXiv:1409.1556, 2014. 6
- [50] Y.-C. Su and K. Grauman. Learning spherical convolution for fast features from 360 imagery. In UIST, 2017. 2
- [51] Y.-C. Su and K. Grauman. Making 360 video watchable in 2d: Learning videography for click free viewing. CVPR, 2017. 1, 2, 3
- [52] Y.-C. Su, D. Jayaraman, and K. Grauman. Pano2vid: Automatic cinematography for watching 360 videos. In ACCV, 2016. 1, 2, 3, 6, 8
- [53] Y. Tang and X. Wu. Saliency detection via combining region-level and pixel-level predictions with cnns. In ECCV, 2016. 2
- [54] B. W. Tatler. The central fixation bias in scene viewing: Selecting an optimal viewing position independently of motor biases and image feature distributions. Journal of Vision, 2007. 6
- [55] P. Tokmakov, K. Alahari, and C. Schmid. Learning video object segmentation with visual memory. arXiv preprint arXiv:1704.05737, 2017. 3
- [56] D. Tsai, M. Flagg, and J. M. Rehg. Motion coherent tracking with multi-label mrf optimization. BMVC, 2010. 6
- [57] J. Wang, A. Borji, C.-C. J. Kuo, and L. Itti. Learning a combined model of visual saliency for fixation prediction. TIP, 2016. 2
- [58] L. Wang, H. Lu, Y. Wang, M. Feng, D. Wang, B. Yin, and X. Ruan. Learning to detect salient objects with image-level supervision. In CVPR, 2017. 2
- [59] L. Wang, L. Wang, H. Lu, P. Zhang, and X. Ruan. Saliency detection with recurrent fully convolutional networks. In ECCV, 2016. 2
- [60] Q. Wang, W. Zheng, and R. Piramuthu. Grab: Visual saliency via novel graph model and background priors. In CVPR, 2016. 2
- [61] W. Wang, J. Shen, and L. Shao. Consistent video saliency using local gradient flow optimization and global refinement. TIP, 2015. 6, 7
- [62] P. Weinzaepfel, J. Revaud, Z. Harchaoui, and C. Schmid. Deepflow: Large displacement optical flow with deep matching. In ICCV, 2013. 5, 6, 7
- [63] S. Xingjian, Z. Chen, H. Wang, D.-Y. Yeung, W.-K. Wong, and W.-c. Woo. Convolutional lstm network: A machine learning approach for precipitation nowcasting. In NIPS, 2015. 2, 3, 5
- [64] Y. Yu, S. Lee, J. Na, J. Kang, and G. Kim. A deep ranking model for spatio-temporal highlight detection from a 360 video. In AAAI, 2018. 1
- [65] J. Zhang and S. Sclaroff. Exploiting surroundedness for saliency detection: a boolean map approach. TPAMI, 2016. 2
- [66] B. Zhou, A. Khosla, A. Lapedriza, A. Oliva, and A. Torralba. Learning deep features for discriminative localization. In CVPR, 2016. 2
- [67] Y. Zhu, Z. Lan, S. Newsam, and A. G. Hauptmann. Guided optical flow learning. CVPR BNMW Workshop, 2017. 3, 5