In Proceedings of the Asian Conference on Computer Vision (ACCV), 2016

Pano2Vid: Automatic cinematography for watching 360° videos

Yu-Chuan Su, Dinesh Jayaraman, and Kristen Grauman

The University of Texas at Austin

Abstract. We introduce the novel task of Pano2Vid — *automatic cinematography* in panoramic 360° videos. Given a 360° video, the goal is to direct an imaginary camera to *virtually* capture natural-looking normal field-of-view (NFOV) video. By selecting "where to look" within the panorama at each time step, Pano2Vid aims to free both the videographer and the end viewer from the task of determining what to watch. Towards this goal, we first compile a dataset of 360° videos downloaded from the web, together with human-edited NFOV camera trajectories to facilitate evaluation. Next, we propose AutoCam, a data-driven approach to solve the Pano2Vid task. AutoCam leverages NFOV web video to discriminatively identify space-time "glimpses" of interest at each time instant, and then uses dynamic programming to select optimal human-like camera trajectories. Through experimental evaluation on multiple newly defined Pano2Vid performance measures against several baselines, we show that our method successfully produces informative videos that could conceivably have been captured by human videographers.

1 Introduction

A 360° video camera captures the entire visual world as observable from its optical center. This is a dramatic improvement over standard *normal field-of-view (NFOV)* video, which is usually limited to 65°. This increase in field of view affords exciting new ways to record and experience visual content. For example, imagine a scientist in a shark cage studying the behavior of a pack of sharks that are swimming in circles all around her. It would be impossible for her to observe each shark closely in the moment. If, however, she had a 360° camera continuously recording spherical panoramic video of the scene all around her, she could later replay her entire visual experience hundreds of times, "choosing her own adventure" each time, focusing on a different shark etc. Similarly, a footballer wearing a 360° camera could review a game from his in-game positions at all times, studying passes which were open to him that he had not noticed in the heat of the game.

In such cases and many others, 360° video offers (1) a richer way to experience the visual world unrestricted by a limited field of view, attention, and cognitive capacity, even compared to actually being present *in situ*, while (2) partially freeing the videographer of camera control. Indeed, 360° videos are growing increasingly popular as consumer- and production-grade 360° cameras

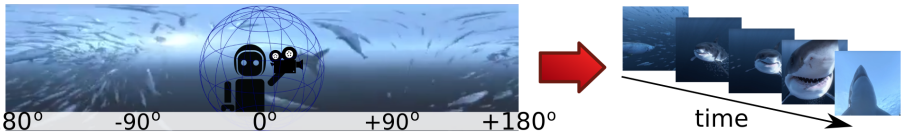


Fig. 1: The "Pano2Vid" task: convert input 360° video to output NFOV video.

(e.g., Ricoh, Bublcam, 360Fly, GoPro) enter the market, and websites such as YouTube and Facebook begin to support 360° content with viewers.

This new medium brings with it some new challenges. Foremost, it largely transfers the choice of "where to look" from the videographer to the viewer. This makes 360° video hard to view effectively, since a human viewer must now somehow make the "where to look" choice and convey it to a video player in real time. Currently, there are three main approaches. In the first approach, the user navigates a 360° video manually. A standard viewer displays a small portion of the 360° video corresponding to a normal field-of-view (NFOV) camera[1]. The user must either drag using a mouse, or click on up-down-left-right cursors, to adjust the virtual camera pose continuously, for the full duration of the video. A second approach is to show the user the entire spherical panoramic video unwrapped into its warped, equirectangular projection[2]. While less effort for the user, the distortions in this projected view make watching such video difficult and unintuitive. The third approach is to wear a virtual reality headset, a natural way to view 360° video that permits rich visual experiences. However, a user must usually be standing and moving about, with a headset obscuring all real-world visual input, for the full duration of the video. This can be uncomfortable and/or impractical over long durations. Plus, similar to the click-based navigation, the user remains "in the dark" about what is happening elsewhere in the scene, and may find it difficult to decide where to look to find interesting content in real time. In short, all three existing paradigms have interaction bottlenecks, and viewing 360° video remains cumbersome.

To address this difficulty, we define "Pano2Vid", a new computer vision problem (see Fig 1). The task is to design an algorithm to automatically control the pose and motion of a virtual NFOV camera within an input 360° video. The output of the system is the NFOV video captured by this virtual camera. Camera control must be optimized to produce video that could conceivably have been captured by a human observer equipped with a *real* NFOV camera. A successful Pano2Vid system would therefore take the burden of choosing "where to look" off both the videographer and the end viewer: the videographer could enjoy the moment without consciously directing her camera, while the end viewer could watch intelligently-chosen portions of the video in the familiar NFOV format.

For instance, imagine a Pano2Vid system that automatically outputs hundreds of NFOV videos for the 360° shark cage video, e.g., focusing on different sharks/subgroups of sharks in turn. This would make analysis much easier for the scientist, compared to manually selecting and watching hundreds of different camera trajectories through the original 360° video. A machine-selected camera

[1] See, e.g., `https://www.youtube.com/watch?v=HNOT_feL27Y`,
[2] See, e.g., `https://www.youtube.com/watch?v=Nv6MCkaR5mc`

trajectory could also serve as a useful default initialization for viewing 360° content, where a user is not forced to interact with the video player continuously, but could *opt* to do so when they desire. Such a system could even be useful as an editing aid for cinema. 360° cameras could partially offload camera control from the cinematographer to the editor, who might start by selecting from machine-recommended camera trajectory proposals.

This work both formulates the Pano2Vid problem and introduces the first approach to address it. The proposed "AutoCam" approach first learns a discriminative model of human-captured NFOV web video. It then uses this model to identify candidate viewpoints and events of interest to capture in 360° video, before finally stitching them together through optimal camera motions using a dynamic programming formulation for presentation to human viewers. Unlike prior attempts at automatic cinematography, which focus on virtual 3D worlds and employ heuristics to encode popular idioms from cinematography [1–6], AutoCam is (a) the first to tackle real video from dynamic cameras and (b) the first to consider directly *learning* cinematographic tendencies from data.

The contributions of this work are four-fold: (1) we formulate the computer vision problem of automatic cinematography in 360° video (Pano2Vid), (2) we propose a novel Pano2Vid algorithm (AutoCam), (3) we compile a dataset of 360° web video, annotated with ground truth human-directed NFOV camera trajectories[3] and (4) we propose a comprehensive suite of objective and subjective evaluation protocols to benchmark Pano2Vid task performance. We benchmark AutoCam against several baselines and show that it is the most successful at virtually capturing natural-looking NFOV video.

2 Related Work

Video summarization Video summarization methods condense videos in *time* by identifying important events [7]. A summary can take the form of a keyframe sequence [8–13], a sequence of video highlight clips [14–19], or montages of frames [20] or video clip excerpts [21]. Among these, our proposed AutoCam shares with [10–12, 18, 19] the idea of using user-generated visual content from the web as exemplars for informative content. However, whereas existing methods address *temporal* summarization of NFOV video, we consider a novel form of *spatial* summarization of 360° video. While existing methods decide *which frames to keep* to shorten a video, our problem is instead to choose *where to look* at each time instant. Moreover, existing summarization work assumes video captured intentionally by a camera person (or, at least, a well-placed surveillance camera). In contrast, our input videos largely lack this deliberate control. Moreover, we aim not only to capture all *important* content in the original 360° video, but to do so in a *natural, human-like* way so that the final output video resembles video shot by human videographers with standard NFOV cameras.

Camera selection for multi-video summarization Some efforts address *multi-video* summarization [22–24], where the objective is to select, at each time instant, video feed from one camera among many to include in a summary video.

[3] `http://vision.cs.utexas.edu/projects/Pano2Vid`

The input cameras are human-directed, whether stationary or dynamic [24]. In contrast, we deal with a single hand-held 360° camera, which is not intentionally directed to point anywhere.

Video retargeting Video retargeting aims to adapt a video to better suit the aspect ratio of a target display, with minimal loss of content that has already been purposefully selected by an editor or videographer [25–29]. In our setting, 360° video is captured *without* human-directed content selection; instead, the system must automatically select the content to capture. Furthermore, the spatial downsampling demanded by Pano2Vid will typically be much greater than that required in retargeting.

Visual saliency Salient regions are usually defined as those that capture the visual attention of a human observer, e.g., as measured by gaze tracking. While saliency detectors most often deal with static images [30–33], some are developed for video [34–38], including work that models temporal continuity in saliency [37]. Whereas saliency methods aim to capture where human eyes move subconsciously during a free-viewing task, our Pano2Vid task is instead to capture where human videographers would *consciously point their cameras*, for the specific purpose of capturing a video that is *presentable to other human viewers*. In our experiments, we empirically verify that saliency is not adequate for automatic cinematography.

Virtual cinematography Ours is the first attempt to automate cinematography in complex real-world settings. Existing virtual cinematography work focuses on camera manipulation within much simpler *virtual* environments/video games [1–4], where the perception problem is bypassed (3-D positions and poses of all entities are knowable, sometimes even controllable), and there is full freedom to position and manipulate the camera. Some prior work [5, 6] attempts virtual camera control within restricted *static* wide field-of-view video of classroom and video conference settings, by tracking the centroid of optical flow in the scene. In contrast, we deal with unrestricted 360° web video of complex real-world scenes, captured by moving amateur videographers with shaky hand-held devices, where such simple heuristics are insufficient. Importantly, our approach is also the first to *learn content-based camera control from data*, rather than relying on hand-crafted guidelines/heuristics as all prior attempts do.

3 Approach

We first define the Pano2Vid problem in more detail (Sec. 3.1) and describe our data collection process (Sec. 3.2). Then we introduce our AutoCam approach (Sec. 3.3). Finally, we introduce several evaluation methodologies for quantifying performance on this complex task (Sec. 3.4), including an annotation collection procedure to gather human-edited videos for evaluation (Sec. 3.5).

3.1 Pano2Vid Problem Definition

First, we define the Pano2Vid task of automatic videography for 360° videos. Given a dynamic panoramic 360° video, the goal is to produce "natural-looking"

normal-field-of-view (NFOV) video. For this work, we define NFOV as spanning a horizontal angle of 65.5° (corresponding to a typical 28mm focal length full-frame Single Lens Reflex Camera [39]) with a 4:3 aspect ratio.

Broadly, a natural-looking NFOV video is one which is indistinguishable from human-captured NFOV video (henceforth "HumanCam"). Our ideal video output should be such that it could conceivably have been captured by a human videographer equipped with an NFOV camera whose optical center coincides exactly with that of the 360° video camera, with the objective of best presenting the event(s) in the scene. In this work, we do not allow skips in time nor camera zoom, so the NFOV video is defined completely by the camera trajectory, i.e., the time sequence of the camera's principal axis directions. To solve the Pano2Vid problem, a system must determine a NFOV camera trajectory through the 360° video to carve it into a HumanCam-like NFOV video.

3.2 Data Collection: 360° Test Videos and NFOV Training Videos

Human-directed camera trajectories are content-based and often present scenes in *idiomatic* ways that are specific to the situations, and with specific intentions such as to tell a story [40]. Rather than hand-code such characteristics through cinematographic rules/heuristics [1–4], we propose to *learn* to capture NFOV videos, by observing HumanCam videos from the web. The following overviews our data collection procedure.

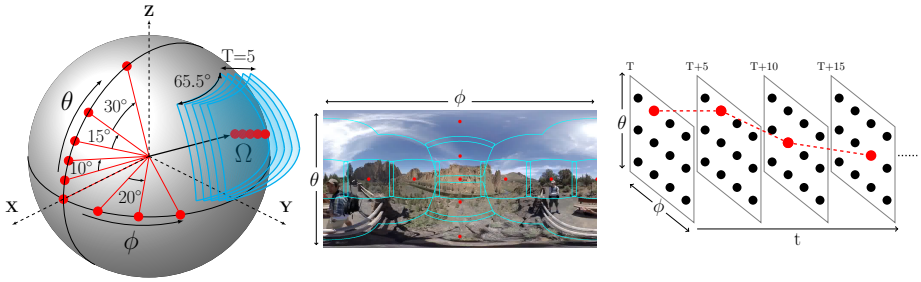
360° videos We collect 360° videos from YouTube using the keywords "Soccer," "Mountain Climbing," "Parade," and "Hiking." These terms were selected to have (i) a large number of relevant 360° video results, (ii) dynamic activity, i.e., spatio-temporal *events*, rather than just static scenes, and (iii) possibly multiple regions/events of interest at the same time. For each query term, we download the top 100 videos sorted by relevance and filter out any that are not truly 360° videos (e.g., animations, slide shows of panoramas, restricted FOV) or have poor lighting, resolution, or stitching quality. This yields a Pano2Vid test set of 86 total 360° videos with a combined length of 7.3 hours. See the project webpage[3] for example videos.

HumanCam NFOV videos In both the learning stage of AutoCam (Sec 3.3) and the proposed evaluation methods (Sec 3.4), we need a model for HumanCam. We collect a large diverse set of HumanCam NFOV videos from YouTube using the same query terms as above and imposing a per-video max length of 4 minutes. For each query term, we collect about 2,000 videos, yielding a final HumanCam set of 9,171 videos totalling 343 hours. See Supp. for details.

3.3 AutoCam: Proposed Solution for the Pano2Vid Task

We now present AutoCam, our approach to solve the Pano2Vid task. The input to the system is an arbitrary 360° video, and the output is a natural looking NFOV video extracted from it.

AutoCam works in two steps. First, it evaluates all virtual NFOV spatio-temporal "glimpses" (ST-glimpses) sampled from the 360° video for their "capture-worthiness"—their likelihood of appearing in HumanCam NFOV video. Next,



(a) Sample ST-glimpses and score capture-worthiness. **(b)** Stitch glimpses.

Fig. 2: AutoCam first samples and scores the capture-worthiness of ST-glimpses. It then jointly selects a glimpse for each time step and stitches them together to form the output NFOV video. Best viewed in color.

it selects virtual NFOV camera trajectories, prioritizing both (i) high-scoring ST-glimpses from the first step, and (ii) smooth human-like camera movements. AutoCam is fully automatic and does not require any human input. Furthermore, as we will see next, the proposed learning approach is unsupervised—it learns a model of human-captured NFOV video simply by watching clips people upload to YouTube.

Capture-worthiness of spatio-temporal glimpses The first stage aims to find content that is likely to be captured by human videographers. We achieve this by scoring the capture-worthiness of candidate ST-glimpses sampled from the 360° video. An ST-glimpse is a five-second NFOV video clip recorded from the 360° video by directing the camera to a fixed direction in the 360° camera axes. One such glimpse is depicted as the blue stack of frame excerpts on the surface of the sphere in Fig 2a. These are not rectangular regions in the equirectangular projection (Fig 2a, right) so they are projected into NFOV videos before processing. We sample candidate ST-glimpses at longitudes $\phi \in \Phi = \{0, 20, 40, \ldots, 340\}$ and latitudes $\theta \in \Theta = \{0, \pm 10, \pm 20, \pm 30, \pm 45, \pm 75\}$ and intervals of 5 seconds. Each candidate ST-glimpse is defined by the camera principal axis direction (θ, ϕ) and time t: $\Omega_{t,\theta,\phi} \equiv (\theta_t, \phi_t) \in \Theta \times \Phi$. See Supp.

Our approach learns to score capture-worthiness from HumanCam data. We expect capture-worthiness to rely on two main facets: content and composition. The *content* captured by human videographers is naturally very diverse. For example, in a mountain climbing video, people may consider capturing the recorder and his companion as well as a beautiful scene such as the sunrise as being equally important. Similarly, in a soccer video, a player dribbling and a goalkeeper blocking the ball may both be capture-worthy. Our approach accounts for this diversity both by learning from a wide array of NFOV HumanCam clips and by targeting related domains via the keyword query data collection described above. The *composition* in HumanCam data is a meta-cue, largely independent of semantic content, that involves the framing effects chosen by a human videographer. For example, an ST-glimpse that captures only the bottom half of a human face is not capture-worthy, while a framing that captures the full face is; a composition for outdoor scenes may tend to put the horizon towards the middle of the frame, etc.

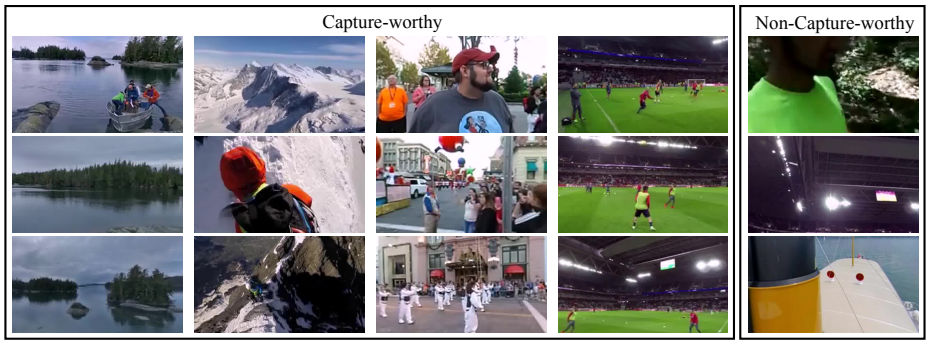


Fig. 3: Example glimpses scored by AutoCam. Left 4 columns are glimpses considered capture-worthy by our method; each column is from the same time step in the same video. Right column shows non-capture-worthy glimpses.

Rather than attempt to characterize capture-worthiness through rules, AutoCam *learns* a data-driven model. We make the following hypotheses: (i) the majority of content in HumanCam NFOV videos were considered capture-worthy by their respective videographers (ii) most random ST-glimpses would *not* be capture-worthy. Based on these hypotheses, we train a capture-worthiness classifier. Specifically, we divide each HumanCam video into non-overlapping 5-second clips, to be used as positives, following (i) above. Next, *all* candidate ST-glimpses extracted from (disjoint) 360° videos are treated as negatives, per hypothesis (ii) above. Due to the weak nature of this supervision, both positives and negatives may have some label noise.

To represent each ST-glimpse and each 5s HumanCam clip, we use off-the-shelf convolutional 3D features (C3D) [41]. C3D is a generic video feature based on 3D (spatial+temporal) convolution that captures appearance and motion information in a single vector representation, and is known to be useful for recognition tasks. We use a leave-one-video-out protocol to train one capture-worthiness classifier for each 360° video. Both the positive and negative training samples are from videos returned by the same keyword query term as the test video, and we sub-sample the 360° videos so that the total number of negatives is twice that of positives. We use logistic regression classifiers; positive class probability estimates of ST-glimpses from the left-out video are now treated as their capture-worthiness scores.

Fig 3 shows examples of "capture-worthy" and "non-capture-worthy" glimpses as predicted by our system. We see that there may be multiple capture-worthy glimpses at the same moment, and both the content and composition are important for capture-worthiness. Please see the Supp. file for further analysis, including a study of how our predictions correlate with the viewpoint angles.

Camera trajectory selection After obtaining the capture-worthiness score of each candidate ST-glimpse, we construct a camera trajectory by finding a path over the ST-glimpses that maximizes the *aggregate* capture-worthiness score, while simultaneously producing human-like smooth camera motions. A naive solution would be to choose the glimpse with the maximum score at each step. This trajectory would capture the maximum aggregate capture-worthiness, but

the resultant NFOV video may have large/shaky unnatural camera motions. For example, when two ST-glimpses located in opposite directions on the viewing sphere have high capture-worthiness scores, such a naive solution would end up switching between these two directions at every time-step, producing unpleasant and even physically impossible camera movements.

Algorithm 1 Camera trajectory selection

$C \leftarrow$ Capture-worthiness scores
$\epsilon \leftarrow$ Valid camera motion
for all θ, ϕ **do**
  $Accum[\Omega_{1,\theta,\phi}] \leftarrow C[\Omega_{1,\theta,\phi}]$
end for
for $t \leftarrow 2, T$ **do**
  for all θ, ϕ **do**
    $\Omega_{t-1,\theta',\phi'} \leftarrow \arg\max_{\theta',\phi'} Accum[\Omega_{t-1,\theta',\phi'}]$
      $s.t.\ |\Omega_{t,\theta,\phi} - \Omega_{t-1,\theta',\phi'}| \leq \epsilon$
    $Accum[\Omega_{t,\theta,\phi}] \leftarrow Accum[\Omega_{t-1,\theta',\phi'}] + C[\Omega_{t,\theta,\phi}]$
    $TraceBack[\Omega_{t,\theta,\phi}] \leftarrow \Omega_{t-1,\theta',\phi'}$
  end for
end for
$\Omega \leftarrow \arg\max_{\theta,\phi} Accum[\Omega_{T,\theta,\phi}]$
for $t \leftarrow T, 1$ **do**
  $Traj[t] \leftarrow \Omega$
  $\Omega \leftarrow TraceBack[\Omega]$
end for

Instead, to construct a trajectory with more human-like camera operation, we introduce a *smooth motion* prior when selecting the ST-glimpse at each time step. Our prior prefers trajectories that are stable over those that jump abruptly between directions. For the example described above, the smooth prior would suppress trajectories that switch between the two directions constantly and promote those that focus on one direction for a longer amount of time. In practice, we realize the smooth motion prior by restricting the trajectory from choosing an ST-glimpse that is displaced from the previous ST-glimpse by more than $\epsilon = 30°$ in both longitude and latitude, i.e.

$$|\Delta\Omega|_\theta = |\theta_t - \theta_{t-1}| \leq \epsilon,\ |\Delta\Omega|_\phi = |\phi_t - \phi_{t-1}| \leq \epsilon. \tag{1}$$

Given (i) the capture-worthiness scores of all candidate ST-glimpses and (ii) the smooth motion constraint for trajectories, the problem of finding the trajectories with maximum aggregate capture-worthiness scores can be reduced to a shortest path problem. Let $C(\Omega_{t,\theta,\phi})$ be the capture-worthiness score of the ST-glimpse at time t and viewpoint (θ, ϕ). We construct a 2D lattice per time slice, where each node corresponds to an ST-glimpse at a given angle pair. The edges in the lattice connect ST-glimpses from time step t to $t+1$, and the weight for an edge is defined by:

$$E\left(\Omega_{t,\theta,\phi}, \Omega_{t+1,\theta',\phi'}\right) = \begin{cases} -C(\Omega_{t+1,\theta',\phi'}), & |\Omega_{t,\theta,\phi} - \Omega_{t+1,\theta',\phi'}| \leq \epsilon \\ \infty, & \text{otherwise,} \end{cases} \tag{2}$$

where the difference above is shorthand for the two angle requirements in Eq. 1. See Fig 2b, middle and right.

The solution to the shortest path problem over this graph now corresponds to camera trajectories with maximum aggregate capture-worthiness. This solution can be efficiently computed using dynamic programming. See pseudocode in Alg 1. At this point, the optimal trajectory indicated by this solution is "discrete" in the sense that it makes jumps between discrete directions after each 5-second

time-step. To smooth over these jumps, we linearly interpolate the trajectories between the discrete time instants, so that the final camera motion trajectories output by AutoCam are continuous. In practice, we generate K NFOV outputs from each 360° input by (i) computing the best trajectory ending at each ST-glimpse location (of 198 possible), and (ii) picking the top K of these.

Note that AutoCam is an offline batch processing algorithm that watches the entire video before positioning the virtual NFOV camera at each frame. This matches the target setting of a human editing a pre-recorded 360° video to capture a virtual NFOV video, as the human is free to watch the video in full. In fact, we use human-selected edits to help evaluate AutoCam (Sec 3.5, 4.2).

3.4 Quantitative Evaluation of Pano2Vid Methods

Next we present evaluation metrics for the Pano2Vid problem. A good metric must measure how close a Pano2Vid algorithm's output videos are to human-generated NFOV video, while simultaneously being reproducible for easy benchmarking in future work. We devise two main criteria:

- **HumanCam-based metrics**: Algorithm outputs should look like HumanCam videos—the more indistinguishable the algorithm outputs are from real manually captured NFOV videos, the better the algorithm.
- **HumanEdit-based metrics**: Algorithms should select camera trajectories close to human-selected trajectories ("HumanEdit")—The closer algorithmically selected camera motions are to those selected by humans editing the same 360° video, the better the algorithm.

The following fleshes out a family of metrics capturing these two criteria. All of which can easily be reproduced and compared to easily, given the same training/testing protocol is applied.

HumanCam-based metrics We devise three HumanCam-based metrics:

Distinguishability*: Is it possible to distinguish Pano2Vid and HumanCam outputs?* Our first metric quantifies *distinguishability* between algorithmically generated and HumanCam videos. For a fully successful Pano2Vid algorithm, these sets would be entirely indistinguishable. This method can be considered as an automatic Turing test that is based on feature statistics instead of human perception; it is also motivated by the adversarial network framework [42] where the objective of the generative model is to disguise the discriminative model. We measure distinguishability using 5-fold cross validation performance of a discriminative classifier trained with HumanCam videos as positives, and algorithmically generated videos as negatives. Training and testing negatives in each split are generated from disjoint sets of 360° video.

HumanCam-likeness*: Which Pano2Vid method gets closer to HumanCam?* This metric directly compares outputs of multiple Pano2Vid methods using their relative distances from HumanCam videos in a semantic feature space (e.g., C3D space). Once again a classifier is trained on HumanCam videos as positives, but this time with *all* algorithm-generated videos as negatives. Similar to exemplar

SVM [43], each algorithm-generated video is assigned a ranking based on its distance from the decision boundary (i.e. HumanCam-likeness), using a leave-one-360°-video-out training and testing scheme. We rank all Pano2Vid algorithms for each 360° video and compare their normalized mean rank; lower is better. We use classification score rather than raw feature distance because we are only interested in the factors that distinguish Pano2Vid and HumanCam. Since this metric depends on the relative comparison of all methods, it requires the output of all methods to be available during evaluation.

Transferability*: Do semantic classifiers transfer between Pano2Vid and HumanCam video?* This metric tries to answer the question: if we learn to distinguish between the 4 classes (based on search keywords) on HumanCam videos, would the classifier perform well on Pano2Vid videos (Human → Auto), and vice versa (Auto → Human)? Intuitively, the more similar the domains, the better the transferability. A similar method is used to evaluate automatic image colorization in [44]. To quantify transferability, we train a multi-class classifier on Auto(/Human) videos generated by a given Pano2Vid method and test it on Human(/Auto) videos. This test accuracy is the method's transferability score.

HumanEdit-based metrics Our metrics thus far compare Pano2Vid outputs with generic NFOV videos. We now devise a set of HumanEdit-based metrics that compare algorithm outputs to human-selected NFOV camera trajectories, given the *same input 360° video*. Sec 3.5 will explain how we obtain HumanEdit trajectories. Note that a single 360° video may have several equally valid HumanEdit camera trajectory annotations, e.g. from different annotators.

Mean cosine similarity*: How closely do the camera trajectories match?* To compute this metric, we first measure the frame-wise cosine distance (in the 360° camera axes) between the virtual NFOV camera principal axes selected by Pano2Vid and HumanEdit. These frame-wise distances are then pooled into one score in two different ways: (1) **Trajectory pooling**: Each Pano2Vid trajectory is compared to its best-matched HumanEdit trajectory. Frame-wise cosine distances to each human trajectory are first averaged. Each Pano2Vid output is then assigned a score corresponding to the minimum of its average distance to HumanEdit trajectories. Trajectory pooling rewards Pano2Vid outputs that are similar to at least one HumanEdit trajectory *over the whole video*, and (2) **Frame pooling**: This pooling method rewards Pano2Vid outputs that are similar to different HumanEdit tracks in different portions of the video. First, each frame is assigned a score based on its minimum frame-wise cosine distance to a HumanEdit trajectory. Now, we simply average this over all frames to produce the "frame distance" score for that trajectory. Frame pooling rewards Pano2Vid outputs that are similar to any HumanEdit trajectory at each frame.

Mean overlap*: How much do the fields of view overlap?* The cosine distance between principal axes ignores the fact that cameras have limited FOV. To account for this, we compute "overlap" metrics on Pano2Vid and HumanEdit camera FOVs on the unit sphere. Specifically, we approximate the overlap using $\max(1 - \frac{\Delta\Omega}{\text{FOV}}, 0)$, which is 1 when the pricipal axes coincide, and 0 for all $\Delta\Omega >$ FOV. We apply both trajectory and frame pooling as for the cosine distance.

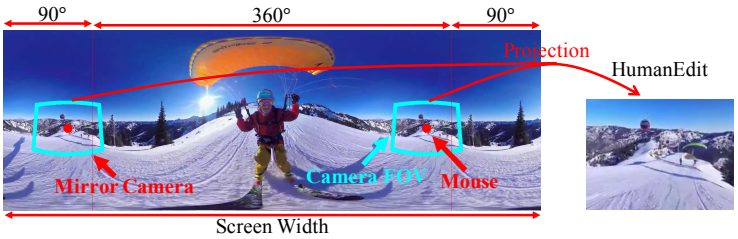


Fig. 4: HumanEdit interface. We display the 360° video in equirectangular projection and ask annotators to direct the camera using the mouse. The NFOV video is rendered and displayed to the annotator offline. Best viewed in color.

3.5 HumanEdit Annotation Collection

To collect human editors' annotations, we ask multiple annotators to watch the 360° test videos and generate the camera trajectories from them. We next describe the annotation collection process. We then analyze the consistency of collected HumanEdit trajectories.

Annotation interface and collection process Fig 4 shows the HumanEdit annotation interface. We display the entire 360° video in equirectangular projection. Annotators are instructed to move a cursor to direct a virtual NFOV camera. Virtual NFOV frame boundaries are backprojected onto the display (shown in cyan) in real time as the camera is moved.

We design the interface to mitigate problems due to discontinuities at the edges. First, we extend the panoramic strip by 90° on the left and right as shown in Fig 4. The cursor may now smoothly move over the 360° boundaries to mimic camera motion in the real world. Second, when passing over these boundaries, content is duplicated, and so is the cursor position and frame boundary rendering. When passing over an edge of this extended strip, the cursor is repositioned to the duplicated position that is already on-screen by this time. Finally, before each annotation, our interface allows the annotator to pan the panoramic strip to a chosen longitude to position important content centrally if they so choose. Please refer to Supp. for more visual examples and project webpage for the interface in action.

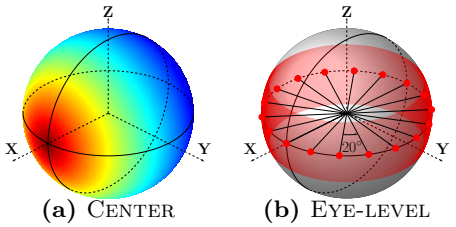
For each 360° video, annotators watch the full video first to familiarize themselves with its content. Thus, they have the same information as our AutoCam approach. Each annotator provides *two* camera trajectories per 360° video, to account for the possibility of multiple good trajectories. Each of 20 360° videos is annotated by 3 annotators, resulting in a final database with 120 human-annotated trajectories adding up to 202 minutes of NFOV video. Our annotators were 8 students aged between 24–30.

HumanEdit consistency analysis After collecting HumanEdit, we measure the consistency between trajectories annotated by different annotators using the metrics described in Sec 3.4.

Table 1 shows the results. The average cosine distance between human trajectories is 0.520, which translates to 59° difference in camera direction at every

Table 1: HumanEdit consistency.

Cosine Similarity	Trajectory	0.520
	Frame	0.803
Overlap	Trajectory	0.462
	Frame	0.643



(a) CENTER **(b)** EYE-LEVEL

Table 2: Baseline illustration. CENTER generates random trajectories biased toward the 360° video center. EYE-LEVEL generates static trajectories on the equator.

moment. The difference is significant, considering the NFOV is 65°. Frame differences, however, are much smaller—37° on average, and overlap of $> 50\%$ across annotators at every frame. These differences indicate that there is more than one natural trajectory for each 360° video, and different annotators may pick different trajectories. Still, with $> 50\%$ overlap at any given moment, we see that there is often something in the 360° video that catches everyone's eye; different trajectories arise because people place different priority on the content and choose to navigate through them in different manner. Overall, this analysis justifies our design to ask each annotator to annotate twice and underscores the need for metrics that take the multiple trajectories into account, as we propose.

4 Experiment

Baseline We compare our method to the following baselines.

- CENTER + Smooth motion prior—This baseline biases camera trajectories to lie close to the "center" of the standard 360° video axes, accounting for the fact that user-generated 360° videos often have regions of interest close to the centers of the standard equirectangular projection. We sample from a Gaussian starting at the center, then centered at the current direction for subsequent time-steps. See Table 2a.
- EYE-LEVEL prior— This baseline points the NFOV camera to some pre-selected direction on the equator (i.e. at 0° latitude) throughout the video. 0° latitude regions often correspond to eye-level in our dataset. We sample at 20° longitudinal intervals along the equator. See Table 2b.
- SALIENCY trajectory—This baseline uses our pipeline, but replaces our discriminative capture-worthiness scores with the saliency score from a popular saliency method [30].
- AUTOCAM W/O STITCHING—This is an ablated variant of our method that has no camera motion constraint. We generate multiple outputs by sampling ST-glimpses at each time step based on their capture-worthiness scores.

For each method we generate K=20 outputs per 360° video, and average their results. See Supp. for details.

Table 3: Pano2Vid performance: HumanCam-based metrics. The arrows in column 3 indicate whether lower scores are better (⇓), or higher scores (⇑).

			Center	Eye-level	Saliency	AutoCam w/o stitching (ours)	AutoCam (ours)
Distinguishability	Error rate (%)	⇑	1.3	2.8	5.2	4.0	**7.0**
HumanCam-Likeness	Mean Rank	⇓	0.659	0.571	0.505	0.410	**0.388**
Transferability	Human → Auto	⇑	0.574	0.609	0.595	**0.637**	0.523
	Auto → Human		0.526	0.559	0.550	0.561	**0.588**

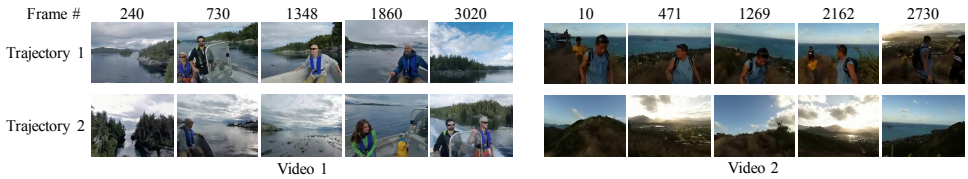


Fig. 5: Example AutoCam outputs. We show the result for two 360° videos, and two trajectories for each.

Implementation Details Following [41], we split the input video into 16 frame clips then extract the $fc6$ activation for each clip and average them as the C3D video features (whether on glimpses or full HumanCam clips). We use temporally non-overlapping clips to reduce computation cost. We use the Sport1M model provided by the authors without fine-tuning. We use logistic regression with $C = 1$ in all experiments involving a discriminative classifier.

4.1 HumanCam-based Evaluation

We first evaluate our method using the HumanCam-based metrics (defined in Sec 3.4). Table 3 shows the results. Our full method (AutoCam) performs the best in nearly all metrics. It improves the Distinguishability and HumanCam-Likeness by 35% and 23% respectively, compared to the best baseline. The advantage in Transferability is not as significant, but is still 5% better in Auto → Human transfer. AutoCam w/o stitching is second-best overall, better than all other 3 baselines. These results establish that both components of our method—(i) capture-worthy ST-glimpse selection, and (ii) smooth camera motion selection—capture important aspects of human-like NFOV videos.

Among the remaining baselines, Saliency, which is content-based and also uses our smooth motion selection pipeline, performs significantly better than Center and Eye-level, which are uniformly poor throughout. However, Saliency falls well short of even AutoCam w/o stitching, establishing that saliency is a poor proxy for capture-worthiness.

We observe that the transferability metric results are asymmetric, and AutoCam only does best on transferring semantic classifiers in the Auto → Human direction. Interestingly, AutoCam w/o stitching is best on Human → Auto, but the smooth motion constraint adversely affects this score for AutoCam. This performance drop may be caused by the content introduced when trying to stitch two spatially disjoint capture-worthy ST-glimpses. While AutoCam w/o stitching can jump directly between such glimpses, AutoCam is constrained to move through less capture-worthy content connecting them. Moreover, intu-

Table 4: Pano2Vid performance: HumanEdit-based metrics. Higher is better.

		Center	Eye-level	Saliency	AutoCam w/o stitching (ours)	AutoCam (ours)
Cosine Similarity	Trajectory	0.257	0.268	0.063	0.184	**0.304**
	Frame	0.572	0.575	0.387	0.541	**0.581**
Overlap	Trajectory	0.194	0.243	0.094	0.202	**0.255**
	Frame	0.336	**0.392**	0.188	0.354	0.389

itively, scene recognition relies more on content selection than on camera motion, so incoherent motion might not disadvantage AutoCam w/o stitching.

Fig 5 shows example output NFOV videos of our algorithm for two 360° videos. For each video, we show two different generated trajectories. Our method is able to find multiple natural NFOV videos from each input. See project webpage for video examples and comparisons of different methods.

4.2 HumanEdit-based Evaluation

Next, we evaluate all methods using the HumanEdit-based metrics (Sec 3.4). Table 4 shows the results. Once again, our method performs best on all but the frame-pooling overlap metrics.

On the cosine distance metric in particular, AutoCam w/o stitching suffers significantly from having incoherent camera motion. Eye-level is second best on these metrics. It does better on frame-wise metrics, suggesting that humans rarely choose static eye-level trajectories. Further, Eye-level does better on overlap metrics, even outperforms AutoCam on average per-frame overlap, suggesting a tendency to make large mistakes which are penalized by cosine metrics but not by overlap metrics. Saliency scores poorly throughout; even though saliency may do well at predicting human gaze fixations, as discussed above, this is not equivalent to predicting plausible NFOV excerpts.

To sum up, our method performs consistently strongly across a wide range of metrics based on both resemblance to generic YouTube NFOV videos, and on closeness to human-created edits of 360° video. This serves as strong evidence that our approach succeeds in capturing human-like virtual NFOV videos.

5 Conclusion

We formulate Pano2Vid: a new computer vision problem that aims to produce a natural-looking NFOV video from a dynamic panoramic 360° video. We collect a new dataset for the task, with an accompanying suite of Pano2Vid performance metrics. We further propose AutoCam, an approach to learn to generate camera trajectories from human-generated web video. We hope that this work will provide the foundation for a new line of research that requires both scene understanding and active decision making. In the future, we plan to explore supervised approaches to leverage HumanEdit data for learning the properties of good camera trajectories and incorporate more task specific features such as human detector.

Acknowledgement. This research is supported in part by NSF IIS -1514118 and a gift from Intel. We also gratefully acknowledge the support of Texas Advanced Computing Center (TACC).

References

1. Christianson, D.B., Anderson, S.E., He, L.w., Salesin, D.H., Weld, D.S., Cohen, M.F.: Declarative camera control for automatic cinematography. In: AAAI/IAAI, Vol. 1. (1996)
2. He, L.w., Cohen, M.F., Salesin, D.H.: The virtual cinematographer: a paradigm for automatic real-time camera control and directing. In: ACM CGI. (1996)
3. Elson, D.K., Riedl, M.O.: A lightweight intelligent virtual cinematography system for machinima production. In: AIIDE. (2007)
4. Mindek, P., Čmolík, L., Viola, I., Gröller, E., Bruckner, S.: Automatized summarization of multiplayer games. In: ACM CCG. (2015)
5. Foote, J., Kimber, D.: Flycam: Practical panoramic video and automatic camera control. In: ICME. (2000)
6. Sun, X., Foote, J., Kimber, D., Manjunath, B.: Region of interest extraction and virtual camera control based on panoramic video capturing. In: IEEE TOM. (2005)
7. Truong, B.T., Venkatesh, S.: Video abstraction: A systematic review and classification. In: ACM TOMM. (2007)
8. Goldman, D.B., Curless, B., Salesin, D., Seitz, S.M.: Schematic storyboarding for video visualization and editing. In: ACM TOG. (2006)
9. Lee, Y.J., Ghosh, J., Grauman, K.: Discovering important people and objects for egocentric video summarization. In: CVPR. (2012)
10. Kim, G., Xing, E.: Jointly aligning and segmenting multiple web photo streams for the inference of collective photo storylines. In: CVPR. (2013)
11. Khosla, A., Hamid, R., Lin, C.J., Sundaresan, N.: Large-scale video summarization using web-image priors. In: CVPR. (2013)
12. Xiong, B., Grauman, K.: Detecting snap points in egocentric video with a web photo prior. In: ECCV. (2014)
13. Gong, B., Chao, W.L., Grauman, K., Sha, F.: Diverse sequential subset selection for supervised video summarization. In: NIPS. (2014)
14. Gygli, M., Grabner, H., Riemenschneider, H., Van Gool, L.: Creating summaries from user videos. In: ECCV. (2014)
15. Gygli, M., Grabner, H., Van Gool, L.: Video summarization by learning submodular mixtures of objectives. In: CVPR. (2015)
16. Potapov, D., Douze, M., Harchaoui, Z., Schmid, C.: Category-specific video summarization. In: ECCV. (2014)
17. Zhao, B., Xing, E.: Quasi real-time summarization for consumer videos. In: CVPR. (2014)
18. Sun, M., Farhadi, A., Seitz, S.: Ranking domain-specific highlights by analyzing edited videos. In: ECCV. (2014)
19. Song, Y., Vallmitjana, J., Stent, A., Jaimes, A.: Tvsum: Summarizing web videos using titles. In: CVPR. (2015)
20. Sun, M., Farhadi, A., Taskar, B., Seitz, S.: Salient montages from unconstrained videos. In: ECCV. (2014)
21. Pritch, Y., Rav-Acha, A., Gutman, A., Peleg, S.: Webcam synopsis: Peeking around the world. In: ICCV. (2007)
22. Fu, Y., Guo, Y., Zhu, Y., Liu, F., Song, C., Zhou, Z.H.: Multi-view video summarization. In: IEEE TOM. (2010)
23. Dale, K., Shechtman, E., Avidan, S., Pfister, H.: Multi-video browsing and summarization. In: CVPR. (2012)
24. Arev, I., Park, H.S., Sheikh, Y., Hodgins, J., Shamir, A.: Automatic editing of footage from multiple social cameras. In: ACM TOG, ACM (2014)

25. Liu, F., Gleicher, M.: Video retargeting: automating pan and scan. In: ACM MM. (2006)
26. Avidan, S., Shamir, A.: Seam carving for content-aware image resizing. In: ACM TOG. (2007)
27. Rubinstein, M., Shamir, A., Avidan, S.: Improved seam carving for video retargeting. In: ACM TOG. (2008)
28. Krähenbühl, P., Lang, M., Hornung, A., Gross, M.: A system for retargeting of streaming video. In: ACM TOG. (2009)
29. Khoenkaw, P., Piamsa-Nga, P.: Automatic pan-and-scan algorithm for heterogeneous displays. In: Springer MTA. (2015)
30. Harel, J., Koch, C., Perona, P.: Graph-based visual saliency. In: NIPS. (2006)
31. Liu, T., Yuan, Z., Sun, J., Wang, J., Zheng, N., Tang, X., Shum, H.Y.: Learning to detect a salient object. In: PAMI. (2011)
32. Achanta, R., Hemami, S., Estrada, F., Susstrunk, S.: Frequency-tuned salient region detection. In: CVPR. (2009)
33. Perazzi, F., Krähenbühl, P., Pritch, Y., Hornung, A.: Saliency filters: Contrast based filtering for salient region detection. In: CVPR. (2012)
34. Itti, L., Baldi, P.F.: Bayesian surprise attracts human attention. In: NIPS. (2005)
35. Zhai, Y., Shah, M.: Visual attention detection in video sequences using spatiotemporal cues. In: ACM MM. (2006)
36. Guo, C., Zhang, L.: A novel multiresolution spatiotemporal saliency detection model and its applications in image and video compression. In: IEEE TIP. (2010)
37. Rudoy, D., Goldman, D., Shechtman, E., Zelnik-Manor, L.: Learning video saliency from human gaze using candidate selection. In: CVPR. (2013)
38. Wang, J., Borji, A., Kuo, C.C., Itti, L.: Learning a combined model of visual saliency for fixation prediction. In: IEEE TIP. (2016)
39. Xiao, J., Ehinger, K.A., Oliva, A., Torralba, A.: Recognizing scene viewpoint using panoramic place representation. In: CVPR. (2012)
40. Mascelli, J.V.: The five C's of cinematography: motion picture filming techniques. Silman-James Press (1998)
41. Tran, D., Bourdev, L., Fergus, R., Torresani, L., Paluri, M.: Learning spatiotemporal features with 3d convolutional networks. In: ICCV. (2015)
42. Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., Bengio, Y.: Generative adversarial nets. In: NIPS. (2014)
43. Malisiewicz, T., Gupta, A., Efros, A.A.: Ensemble of exemplar-svms for object detection and beyond. In: ICCV. (2011)
44. Zhang, R., Isola, P., Efros, A.A.: Colorful image colorization. In: arXiv preprint arXiv:1603.08511. (2016)