Route Tapestries: Navigating 360 Virtual Tour Videos Using Slit-Scan Visualizations

ANONYMOUS AUTHOR(S)*

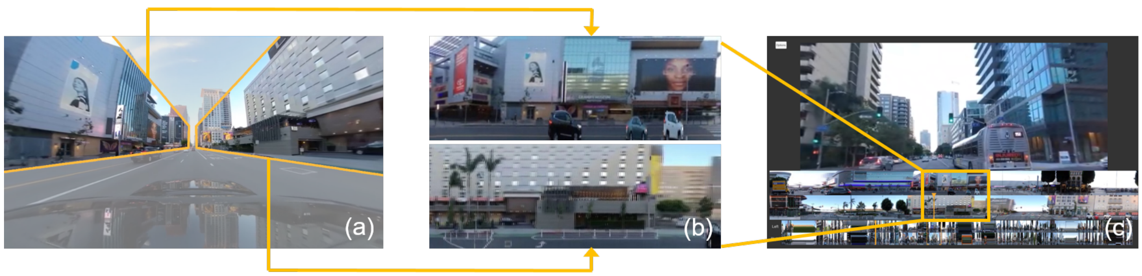

Fig. 1. (*a*, *b*) Route Tapestries are continuous visual summaries of the scenes along the camera route of a 360 virtual tour video. (*c*) Tapestry Player uses Route Tapestries as its timelines for efficient navigation of 360 virtual tour videos.

360 virtual tour videos show the surrounding view while traveling through an environment, and they have become a popular way of experiencing remote places. However, finding particular locations in these videos can be difficult, as current interfaces rely on distorted frame previews for navigation. We propose *Route Tapestries*—continuous visual summaries of scenes along camera routes—for efficiently navigating 360 virtual tour videos. We first present an algorithm for constructing Route Tapestries from a 360 video inspired by the slit-scan photography technique. We then present a desktop video player interface using Route Tapestry timelines for navigation. An online study using a scene-seeking task showed that users were more efficient with Route Tapestries than two existing baseline approaches. An interaction pattern analysis suggests that participants skipped through irrelevant locations faster using Route Tapestries. We conclude by discussing opportunities for supporting HMD interactions and fully automated tapestry generation.

CCS Concepts: • **Computer systems organization** → **Embedded systems**; *Redundancy*; Robotics; • **Networks** → Network reliability.

Additional Key Words and Phrases: 360 video, navigation, slit-scan, video content summary

ACM Reference Format:
Anonymous Author(s). 2018. Route Tapestries: Navigating 360 Virtual Tour Videos Using Slit-Scan Visualizations . In *Woodstock '18: ACM Symposium on Neural Gaze Detection, June 03–05, 2018, Woodstock, NY*. ACM, New York, NY, USA, 21 pages. https://doi.org/10.1145/1122445.1122456

1 INTRODUCTION

360 virtual tour videos show the surrounding view while traveling through an environment, and allow users to freely look around the environment. They convey a strong sense of immersion and have become a popular way for people to

Fig. 2. A 360 virtual tour video frame projected through (*a*) perspective projection (NFOV) ((*b*) equirectangular projection (*c*) Little Planet projection

experience and explore remote places. For example, families may use them to compare vacation destinations; students may visit prospective college campuses, or office workers may seek a brief covert respite from their workplace. Online video platforms such as YouTube[1] have dedicated virtual tour channels, where tours of college campuses, wilderness areas, and urban landscapes are common; here, these channels include both normal field-of-view (NFOV) and 360 virtual tour content. While this emerging media is popular, it also creates new usability challenges for viewers and content creators: current 360 video playback interfaces are still largely based on designs for NFOV videos, and have not considered the affordances provided by 360 content.

One important problem is that contemporary interfaces for quickly navigating video content are poorly suited for navigating 360 videos. It is rare for users to watch tour videos uninterrupted from beginning; instead, users may want to quickly skip over parts of a long tour and resume watching once a particular scene is in sight (i.e. navigate through the video). Because the viewer can only see a portion of the full 360 scene at a time, simple scrubbing on the timeline may cause them to miss relevant visual information. With a few exceptions [27, 42], current systems primarily rely on planar thumbnails as an overview of an entire frame to facilitate temporal navigation. The thumbnail is displayed as a preview when the user selects a frame by placing the pointer over the player timeline or dragging the timeline playhead slider. The previews are created using projection methods to warp the spherical 360 images into 2D visualizations (e.g. equirectangular projection [22], and stereographic—a.k.a. Little Planet—projection [24]). While these projection techniques have their strengths [4, 24], the techniques distort the landscapes and architectural features in 360 virtual tour videos (Figure 2). The single-frame preview also removes a place from the broader spatial context it sits in. These two issues combined make finding particular targets more challenging.

Our approach to this problem is to introduce *Route Tapestries*, which supports video navigation through strip-shaped 'tapestries' constructed with a continuously captured scene along the route. We drew inspiration from Video Tapestry [3], which explored video navigation for NFOV videos using strip-shaped content summaries. Our visualization technique uses slit-scan imaging, where a scene is captured one slice at a time while a moving camera 'scans' the scene (as illustrated in Figure 3). In Route Tapestries, we capture these scenes (typically to the left and right of the path of travel) as continuous 'strips' extending along the camera path (Figure 3), and leverage such strips for video navigation. Relying on visual summaries for video skimming and navigation has a long history of success in the research literature [3, 10, 25, 41]. As a kind of visual summary, Route Tapestries leverage the scene continuity in 360 virtual tour videos to produce a compact yet coherent pictorial presentation of the video content to facilitate navigation. Instead of going through a video frame by frame, users can look at the captured environments in their entirety and make more contextualized navigation decisions.

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

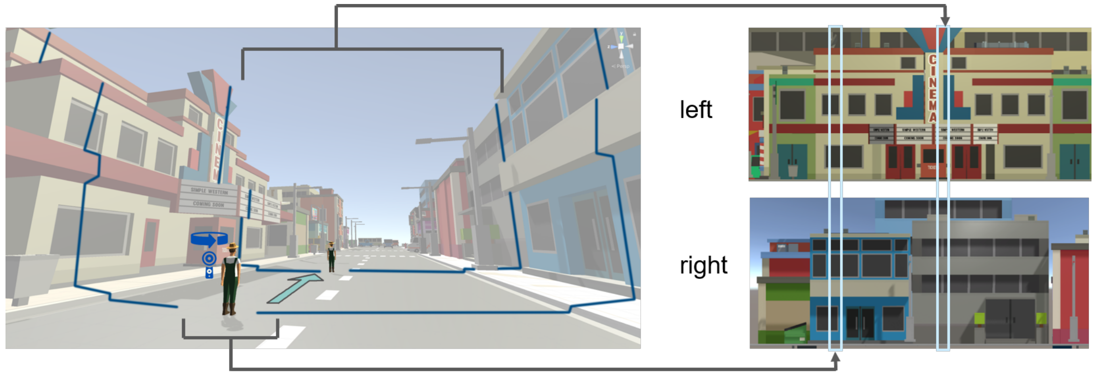

Fig. 3. Using the slit-scan imaging technique, a moving 360 camera captures the scenes along the route as long strips by 'scanning' them. At short intervals, pixels from the part of the scene marked by the blue lines are captured and concatenated to form the eventual Route Tapestries.

Our technique takes advantage of the characteristics of 360 virtual tour videos. First, because these tour videos' have a long continuous nature, they typically feature strong scene continuity, enabling the creation of compact yet informative visual summaries. Second, most 360 virtual tour videos have exciting scenes in a few fixed directions from the camera path: along the tour route and often to the route's left and right sides (as the cameraperson passes points of interest). Third, camerapersons of virtual tour videos usually follow stable and smooth paths, allowing us to leverage the slit-scan techniques.

We first devised an semi-automated algorithm that generates Route Tapestries from an 360 virtual tour video based on a small number of user-supplied parameters, and built Tapestry Player, a desktop-based 360 video player prototype using Route Tapestries for timeline navigation. We then conducted a controlled experiment where 12 participants completed scene-finding tasks using Tapestry Player and two existing baseline techniques (Little Planet previews [24], and equirectangular previews as on YouTube). The study results show that with Route Tapestries the participants completed the tasks 21.2% faster than equirectangular previews and 40.5% faster than Little Planet previews. Further, our exploration of participants' pointer traces reveals that with Route Tapestries, participants tended to study the summaries one section at a time, allowing them to skip over irrelevant content much faster.

As a first step in exploring the concept of Route Tapestries, in this work we focus on its application specifically for virtual tour videos on a desktop environment. We conclude the paper with discussing adapting the current approach for head-mounted displays and a broader range of 360 video content types as sports and documentaries.

We make three contributions in this work: first, we contribute the concept of Route Tapestries as an efficient way to navigate 360 virtual tour videos, and a method for generating them; second, we contribute the design and implementation of Tapestry player; and third, we demonstrate through a user study the benefits of using content summaries (like Route Tapestries) for video navigation.

2 RELATED WORK

This work is situated within the growing area of research exploring new ways of interacting with 360 videos and is directly built on prior work in video navigation interfaces. Our technical approach is inspired by the slit-scan techniques applied in both photography and computer science. We also review research in multi-perspective panoramas, which are visually similar visualizations to ours.

2.1 Interacting with 360 Videos

The full panoramic view of 360 videos enables strong immersion but also brings about new usability challenges. Several recent projects aimed to assist *spatial navigation* in 360 video players to help viewers locate important characters or events out of their field-of-view. Pavel et al. [30] proposed two techniques, pre-aligning the points of interest with the viewing direction at each cut and actively reorienting the shot to reveal relevant content upon user input, to help viewers keep track of important scenes. Lin et al. [18] introduced Outside-In, which signals off-screen points of interest through picture-in-picture previews. Other projects aimed to automate spatial navigation [5, 14, 38, 39] by finding points of interesting using computational measures, such as saliency [37], and generating smooth camera movements to cover them. Pavel et al. [30] also discussed similar approaches to automate the shot reorientation techniques.

Temporal navigation of 360 videos is also a challenge. Some producer-oriented tools [11, 23, 24, 42, 43] offer temporal navigation interface specifically tailored for 360 videos. Nguyen et al. [24] presented Vremiere, a system for editing 360 videos directly in HMDs. Vremiere displayed a 'Little Planet' thumbnail for timeline navigation and highlighted its benefits in promoting spatial awareness. ConvCut [42] used content analysis to provide support for efficiently editing long 360 conversation footage into short highlights. It augmented raw footage with conversation transcripts and other semantic information to aid temporal navigation. Neng and Chambell [22] presented a desktop 360 video player that showed rectangular thumbnails for selected frames in the videos. Hand gestures have been used for 360 video navigational input [31, 35] but were limited to linear controls such as play/stop or fast-forward.

Our work focuses on temporal navigation for a specific type of 360 videos which deliver virtual tour experiences. Geo-tags along the camera routes [16, 27] can facilitate navigating such videos. However, camera position information is not always available for arbitrary videos. In comparison to temporal navigation interfaces for generic 360 videos, such as Little Planet [24] or equirectangular previews, our technique aims to improve efficiency by creating visual summaries that leveraging the camera motion and scene characteristics of 360 virtual tour videos.

2.2 Video Navigation Interfaces

Research on NFOV video navigation has a long and rich history. Earlier approaches included improving the timeline slider to allow for more fine-grained adjustments [12, 33]. Some later research explored aiding navigation with extra information derived either from video content analysis or external annotations. Low-level visual features [36], scene boundaries [2], and salient frames [32] can help viewers leverage extra visual information, but they usually do not support seeking arbitrary scenes. SceneSkim [29] enabled searching and browsing movies through synchronized captions and plot summaries. Kim et al. [15] enhanced MOOC video timelines with user interaction traces to help learners find essential parts. These approaches typically rely on clear semantic structures, but such structures are less common in virtual tour videos. Direct manipulation of objects in the videos [8, 26] offers an intuitive navigation method, but they are not suitable for virtual tour videos where scenes change frequently. The Swift technique [19], which displays pre-cached low-resolution frames during scrubbing, has been shown to improve scene-finding performance. Similar features can also be found on online video platforms such as YouTube and are incorporated into our systems.

Also relevant to our approach is video navigation through content summaries. One kind of summaries consists of a selection of keyframes. The early Hierarchical Video Magnifier [21] marked the video timeline with thumbnails for evenly sampled frames and supported recursive zooming. Later research expanded their method with more sophisticated thumbnail selection and clustering schemes [7]. Thumbnails can also be presented in a grid layout to provide an overview of either local [20] or global [13] content. In contrast, other summaries integrate relevant visual elements to compose a coherent narrative. Goldman et al. [10] turn video input into storyboard-style images with arrows that illustrate character motion and can be dragged along for video scrubbing. Video Tapestry [3] merged visually similar video keyframes to form a navigation timeline. Although their user study did not find an efficiency improvement in navigation, we are inspired by the concept and visual style. Video Summagator [25] transforms a 2D video into a 3D volume, in which navigation can be achieved through moving along the extra dimension. While these techniques are not broadly applicable to generic videos, they exploit scene or character continuity in source videos to create visually appealing summaries and practical navigation tools. Our method is inspired by prior NFOV video navigation interfaces based on content summarization, leveraging scene continuity to produce a compact visualization for 360 virtual tour videos.

2.3 Slit-Scan Visualizations

Slit-scan imaging has a long history in photography and has inspired both visual artists and computer scientists alike [34, 40, 45]. In slit-scan photography, a vertical slit is created between the camera and the scene, blocking incoming light rays except those passing through the slit. In some applications, as objects and entities move past the slit, slit-scan photography creates a visual timeline of objects passing by the slit (e.g., [40]). In other applications, the camera itself moves about the scene, thus creating a movement timeline across a scene and generate panoramas. Also, route panoramas containing detail visual information of an entire location can be generated using a moving camera with slit scanning. Zheng [45] introduced the Route Panorama system that employs digital image-processing techniques and renders route views. The Route Panorama approach requires scanning scenes continuously with a virtual slit in the camera frame to generate a long 2D image belt.

Slit-scan can also be applied to video by placing a slit over video frames to produce a composite image, thus achieving a similar effect. Slit scanning in video is especially useful for creating temporal visualizations for analyzing recorded events. Nunes et al. [28] introduced the TimeLine system, a slit scanning visualization approach, to explore for exploring temporal patterns in video history. Tang et al. [40] presented a video slicing approach consisting of drawing marks anywhere on the video, called slit-tears. These marks, depending on how they are placed, can indicate motion, small changes, directional movement, and relational patterns in video scenes. Our approach extends Route Panorama's slit-scan method to create Route Tapestries and exploits the slit-scan/slit-tear technique to explore and navigate 360 videos.

2.4 Multi-perspective Panorama

Multi-perspective panoramas provide visually appealing visualizations that aggregate location- or motion-based information by combining different perspectives from multiple pictures or videos. They are especially suited for illustrating planar scenes such as landscapes or street imagery. Roman et al. [34] introduced an interactive system for generating multi-perspective urban landscape images composed of serially blended cross-slits images from video frames captured by a moving vehicle. Agarwala et al. [1] employed an automatic Markov Random Field optimization approach to generate composited panoramas of street imagery In a different note, Street Slide [17] improves the flat multi-perspective panoramas produced by the previous approaches by adding parallax effects to create immersive panoramas.

Our approach builds on similar visualizations to multi-perspective panoramas. However, the approaches above prioritize visual quality and are usually demanding in resources, requiring manually labeling video content [34], or heavy pre-processing [1], or precise camera position [17].

3 ROUTE TAPESTRY GENERATION

Our route tapestry generation algorithm processes an input 360 virtual tour video and uses the slit-scan imaging technique to produce a Tapestry visualization for the scenes to a specified direction of the camera path (left and right by default). After turning the input video into the cubemap format (Figure 4.b), the algorithm goes through consecutive frames, taking vertical or horizontal slices of pixels on the cube face that faces the specified direction and adding them to the Tapestry image (Figure 4.c).

Our semi-automated implementation requires users to supply four parameters: camera movement direction (forward or backward), scene depth, camera speed, and additionally, whether top/bottom Tapestries should be generated besides the default left/right ones. In future work we plan to apply more advanced computer vision techniques for higher levels of automation. In the rest of this section, we describe detailed steps of the algorithm and presents its results.

3.1 Generating Route Tapestries Using Slit-Scan Imaging

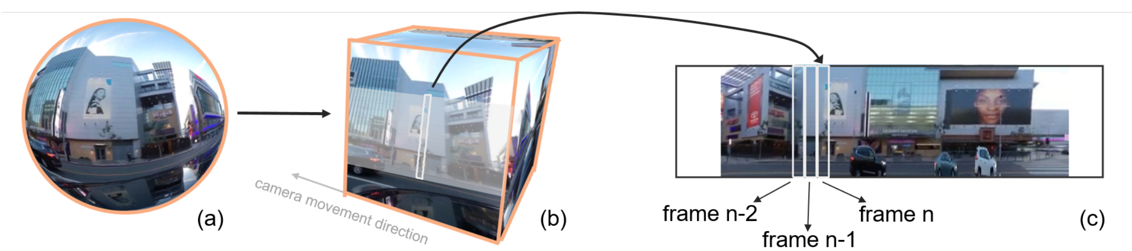

Fig. 4. Steps for generating Route Tapestries from an input 360 video.

Typical 360 virtual tour videos rely on a cameraperson or vehicle carrying a 360 camera, where the tour proceeds through an environment. They use long, continuous, and smooth shots to capture the environments in order to provide a sense of presence for viewers. The camera typically faces the moving direction, or against it if the cameraperson wants viewers to focus on her narrations. Here, the facing direction of the camera corresponds to the default forward direction of the viewer when watching the video. While individual videos vary from this basic template, the vast majority of videos follow this approach. Thus, our algorithm is built on top of these properties.

First, the input 360 video (in an equirectangular format) is converted to a cubemap format, where each faces of the cube shows one direction of the scene as a perspective-correct image (Figure 4.b). We rotate the video by 180° around the yaw (up) axis if the user has specified that the camera faces against the moving direction.

Second, a proper sampling rate $\mathcal{R}$ is calculated based on the user-supplied scene depth and camera moving speed parameter. Oversampling or undersampling cause the visual distortion of a scene being unnaturally stretched or compressed (Figure 5). In a simplified model where we assume the scene lies in a straight line and the camera is at a distance of d from the scene moving at a speed of v, Zhang [45] has shown that the proper sampling rate $\mathcal{R}$ can be calculated using

$$\mathcal{R} = vf/d \tag{1}$$

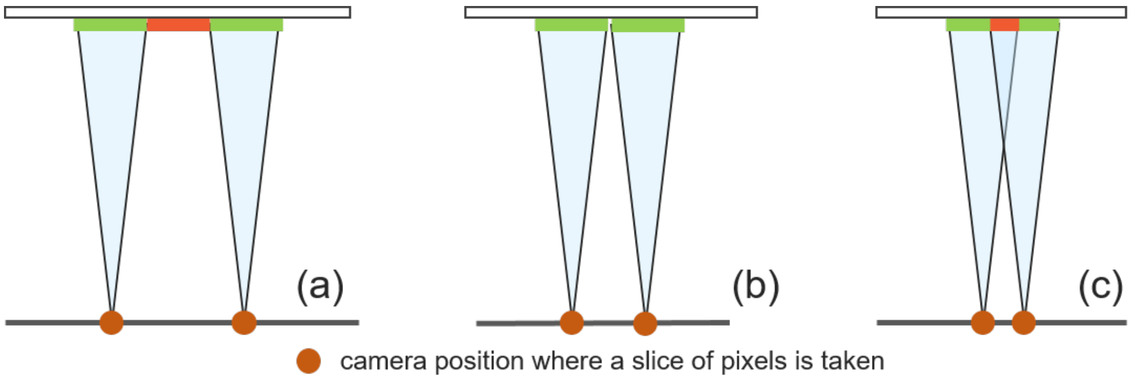

Fig. 5. Sampling rate and image quality in slit-scan imaging. (*a*) Over-sampling, where the scenes marked in green are captured properly but the scenes marked in red are missed (*b*) using the correct sampling rate (*c*) under-sampling, where the scenes marked in green are captured properly but the scenes marked in red are captured twice.

where f is the equivalent focal length of the camera under the pinhole model. Since any face on the video cubemap is equivalently formed through a pinhole camera with a 90° horizontal field-of-view, for the image on a cubemap face of size l in pixels, the equivalent focal lengths is $l/2$. Finally, we transform $\mathcal{R}$ into a more operational quantity n, which is the number of columns or rows of pixels taken from the video cubemap face. n is the sampling rate $\mathcal{R}$ divided by the native frame rate r of the input video:

$$n = \frac{\mathcal{R}}{r} = \frac{vf}{dr} = \frac{vl}{2dr} \tag{2}$$

Currently, the user needs to provide an estimated distance d and camera speed v, or choose from the typical walking or driving speed (walking: 1.4m/s, driving: 12m/s). Possibilities for automatically setting these parameters will be discussed in 7.3.2.

Finally, following the standard method in slit-scan imaging, n columns or rows at the center of the cubemap face are taken from successive video frames and added to the composite image (Figure 4.c). We uses a default 60° vertical field-of-view so that the resulting Tapestries match the field-of-view on desktop players, but this behavior can be overridden by users. In practice, to handle $n < 1$ and non-integer n values, at the k-th frame we take

$$\begin{cases} (\lfloor nk \rfloor - \lfloor n(k-1) \rfloor) \text{ columns or rows,} & \text{if } n >= 1. \\ 1 \text{ column or row,} & \text{if } n < 1 \text{ and } nk - \lfloor nk \rfloor <= n. \end{cases}$$

For a video with a total length of N frames, the eventual Tapestry spans nN horizontal pixels. Note that a full resolution Tapestry can be quite long; for a 30-fps 15-minute video and $n = 1$, the output Tapestry has a length of 27000 pixels. As full-resolution Tapestries can be many times wider than the video player window, only a small slice of them will be displayed at at a time. The detailed design will be introduced in Section 4.2. Since any horizontal span along the Tapestry can be linearly mapped to a temporal span in the video, horizontal positions on the Tapestry can directly reflect video playback progress.

Fig. 6. Example Route Tapestries *(top)* Walking indoor *(middle)* Driving outdoor *(bottom)* Ground seen from a drone

3.2 Algorithm Output

Experimentation showed that our approach can generate informative Route Tapestries for a wide range of 360 virtual tour videos: indoor/outdoor and driving/walking/flying. In Figure 6 we show examples of Route Tapestries for a diverse set of tour types.

As our current algorithm uses a fixed sampling rate to map video frames to Tapestry pixels, the landscapes on the Tapestries are stretched or compressed as the apparent motion of the scene in the video changes (e.g. the cameraperson speeds up or slows down). Such effect is especially pronounced when the camera comes to a complete stop and the slit-scan image become repetitions of the same slice of pixels (e.g. when a car-driven tour stops at the stop light). An interesting side effect of such patterns is that users can immediately identify when the camera is stationary.

4 TAPESTRY PLAYER: A 360 VIDEO PLAYER WITH ROUTE TAPESTRIES

To prototype interactions and conduct performance evaluation for Route Tapestries, we built Tapestry Player, a 360 video player using Route Tapestries as its timelines. We now describe its design and implementation.

4.1 Interface Overview

The overall layout of Tapestry Player is similar to consumer desktop 360 video players, but with the standard timeline replaced by two or more Route Tapestry timelines (Figure 7). Similar to existing 360 video playback systems, users can drag the video window to reorient their viewing direction in the 360 space.

4.2 Route Tapestry Timelines

The Route Tapestry timelines consists of two or more Route Tapestry strips and a yellow bar that vertically span all strips as the playhead slider (Figure 8). The Route Tapestry strips contain the downsized versions of the full-resolution Route Tapestries, which have been pre-generated. The Route Tapestries summarize the passing scenery of the entire video from different angles. Depending on the environment where the virtual tour took place, two or more strips are needed to cover all relevant angles. A left-facing and a right-facing view are sufficient for a typical ground tours (Figure 6(a)(b)). For tour videos with interesting ground or ceiling (sky) views, such as tours shot from a drone (Figure 6(c)), a top or

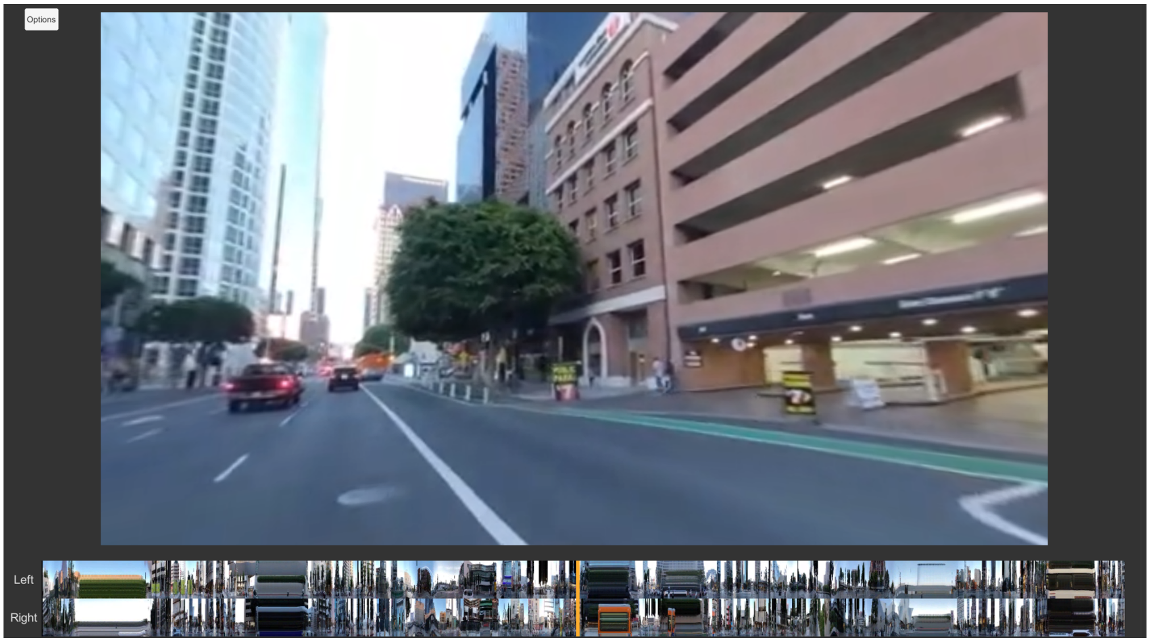

Fig. 7. The Tapestry Player interface.

bottom side Tapestry strip can be added. Like on standard timelines, the playhead slider moves horizontally along the Tapestries to reflect playback progress. It can also be dragged for scrubbing through the video. Navigation actions can thus be performed efficiently within a continuous and undistorted visual context.

Since the Route Tapestry strip is much smaller than the full-resolution Tapestry in dimension, it is often heavily compressed along the horizontal direction, making details hard to discern, especially in long videos. Therefore, when the pointer hovers over the Tapestry strip, a zoomed-in view showing its exact position on the full-resolution Tapestry

Fig. 8. Two Route Tapestry timelines corresponding to the left and right side of the tour path with the full-resolution Tapestries shown above in the zoomed-in view.

is displayed above the timeline (Figure 8). In the zoomed-in view, the pointer position is further highlighted by a pair of arrows linked by a vertical bar. Clicking a scene on the Tapestry immediately makes the playback jump to that scene, and reorients the viewer towards it. Dragging the playhead slider scrubs through the video. During scrubbing, the user can use the full-resolution Tapestries, as well as a *Swift*-style [19] low-resolution preview to check the scene of the current frame. The low-resolution preview is in the equirectangular format for covering the entire 360 field of view.

4.3 Implementation Details

Tapestry Player is implemented using the Unity game engine. The player window is 1920 pixel in width and 1080 pixel in height. Each Route Tapestry timeline strip has a length of 1830 pixels and a height of 64 pixels. The zoomed-in view of each strip is 128 pixels in height and spans the entire width of the window. The 360 video is rendered with a 16:9 aspect ratio. All Route Tapestries were pre-rendered using the algorithm introduced in Section 3.1 implemented in Python.

5 EVALUATION

We designed a controlled experiment to compare the efficiency of navigating 360 virtual tour videos using Route Tapestries against two baseline conditions. Our focus in the study was to understand which interface would allow participants to identify and locate specific scenes faster. Further, we were interested in how participants used the interfaces—for example, whether they choose to scrub through the video, or how fast they move the pointer on the timeline. To complete the study tasks, participants would navigate the through the 360 video to find the target scenes given to them using the previews and the timeline provided by each interface.

We chose two baseline interfaces to compare Route Tapestries against: equirectangular player, and Little Planet player. The equirectangular player models a typical 360 video player as found on YouTube, and we consider this to be a "standard" player. We selected the Little Planet technique [24] since prior work has demonstrated that it is effective for temporal navigation on 360 video editing tasks.

5.1 Design

The study was a repeated measures within-subject design, with the video player *interface* as the independent variable. The three interfaces were *Route Tapestry*, *equirectangular*, and *Little Planet*. Participants completed 14 trials of a scene-finding task on a single 360 virtual tour video. We gave participants a different 360 virtual tour video for each interface to reduce learning effects between videos. These three videos and the target scenes were chosen to be comparable in terms of duration, scene complexity, and style. To control the order effect of *interfaces*, we used the same video presentation order but fully counterbalanced the *interface* presentation order across participants. The order of presentation of the target locations was randomized.

5.2 Interfaces

Route Tapestry. The interface follows the Tapestry Player design as described in Section 4. It uses a left- and right-side Route Tapestries of the tour video as the timelines. The full-resolution Tapestries, which would be displayed in the zoomed-in view, ranged from 46126 to 46170 pixels in length and all had a height of 128 pixels. They were all generated with an estimated scene depth of 15 m and a camera speed of 12 m/s. The timeline strips uses smaller versions of the full-resolution Tapestries resized to 1830×64.

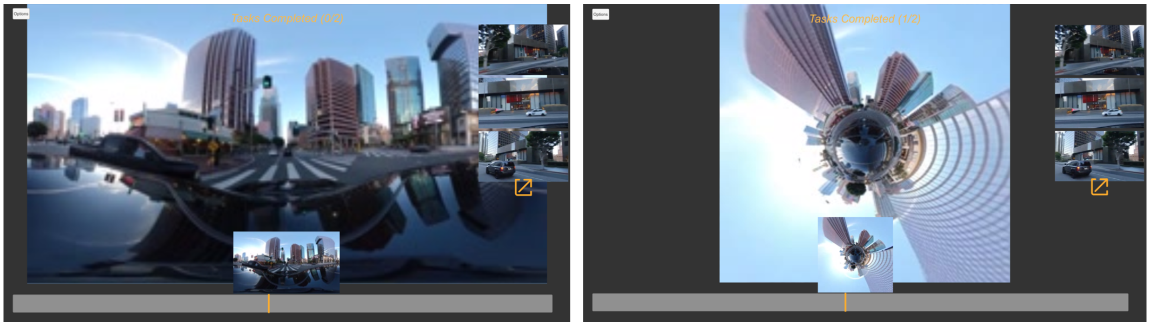

Fig. 9. The two baseline interfaces used in the study: *equirectangular* on the left, and *Little Planet* on the right. Both subfigures show the interface during scrubbing.

Equirectangular. We modeled this standard player around a YouTube desktop 360 player interface (Figure 9, left), where a timeline appears underneath the video window. When the user hovers the pointer on the timeline, an equirectangular frame thumbnail corresponding to pointer position on the timeline is displayed above the pointer. When the user scrubs the playhead slider, a low-resolution equirectangular preview for the currently selected frame is overlaid atop the video window.

In a 1920×1080 window, the timeline is 1830 pixels in length and 64 pixels in height, therefore of the same size as a single Route Tapestry strip. We chose not to strictly follow the scale of the YouTube player timeline to keep dragging and pointing input difficulty consistent across conditions. The equirectangular frame preview has a size of 360×210 in pixels, consistent with the YouTube player.

Little Planet. Modeled around the interface described in Vremiere [24], the interface is identical to *equirectangular* with two exceptions: first, when the user's pointer hovers on the timeline, a Little Planet frame preview is displayed; second, the Little Planet thumbnail was upscaled and overlaid on the video windows during scrubbing (Figure 9, right). The size of the Little Planet preview is 256×256 in pixels.

5.3 Procedure

Due to COVID-19 safety measures, we conducted this study as a remote, online study. Participants completed the study on their own personal computers with the experimenter connected via a videochat connection. The participants downloaded the study software from an online repository, and instructions were delivered via pre-recorded video tutorials to ensure instruction quality.

The study began with the participants watching an overview tutorial explaining study tasks. They then completed a demographics questionnaire. In the main body of the study, participants completed 2 practice and 14 timed trials of the study task for each of the three *interface* conditions. Participants used the practice trials to familiarize themselves with the interface on a separate video not used for any timed trials. In each trial, they were first presented with three pictures depicting the target scenes, as shown in Figure. 10. They were instructed to take time studying these pictures before clicking a 'start' button to begin the timed trial. During the trial, they could see the target scene pictures in a side bar. When the trial was completed (or a 4-minute timeout counter was reached), the software would show the next target scene.

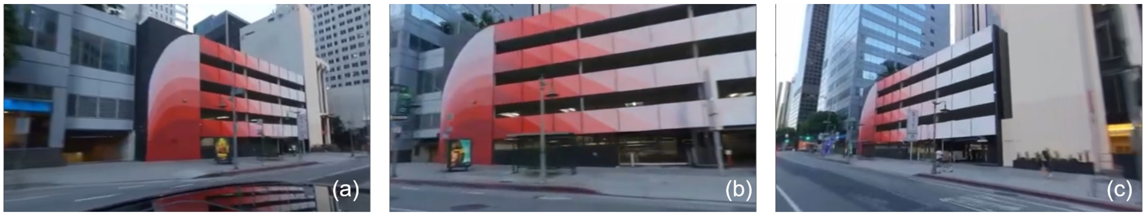

Fig. 10. Each target scene was presented to participants by three NFOV screenshots from the task video. These pictures showed the target as seen from the its left (*a*), center (*b*), and right (*c*) side for an observer facing it.

5.4 Scene-Finding Task

In each trial of the scene-finding task, participants were to locate target scene within the 360 video using the given player interface. Each target scene showed a segment of the urban landscapes, i.e. one or more buildings, that appeared in the video. It was given to the participant through three NFOV pictures showing the buildings as seen from the left, center, and right for an observer who faced it (Figure 10 (a-c)). The pictures were screenshots taken from the task videos, and chosen in a way to minimize visual cues that suggested whether the scene was on the left or right side of the camera path. To complete the task, the participant needed to navigate to the part of the video where the camera passed by the target buildings, adjust the viewing direction to face them, and press the space key to pause the playback and confirm selection. The task was considered completed once the participant had paused the video playback within view of the target, and had adjusted the viewing direction to face the target.

We recorded task completion time of each trials, number of timeouts, pointer traces, and viewing direction changes within the 360 videos.

5.5 Video and Target Selection

We chose four similar 360 virtual tour videos as the study materials. Three of them were used for timed trials and we refer to them as 'task videos'. One was used for training only. The videos were captured from a car driving through an urban landscape. Each was edited to be 15 minutes long.

We carefully selected 14 target scenes from each task video to include a variety of targets while keeping the target sets comparable across the three videos in terms of the scene lengths and positions in the videos.

In particular, all task videos have an equal number of targets (7) chosen in the first half and the second half of the video. Half of the targets (7) in each video were on the left side of the camera moving path while the other half were on the right. For each target, we identified a "valid segment" of the video for when we would consider the task to be completed. These would vary in duration, but would begin when the closer edge of the target appears 45° to the left or right of the camera moving direction, and then ends when the closer edge of the target is at about 135° to the left or right of the camera forward direction. The positions of the target scenes in the video and the length of their valid segments were chosen to cover a wide range of target types meanwhile following comparable distributions across the three videos (Figure 11 and Table 1). Additionally, we selected 12 target scenes for the training video.

5.6 Participants

Twelve paid participants (4 females, $M_{age} = 24.3, SD_{age} = 2.5$) were recruited to take part in our online study. Each participated in the study individually from their homes using their own computers. The majority of the participants

Fig. 11. Distribution of target scenes in the task videos. The lengths of the markers corresponds to the duration of the scene. The colors of the markers denote whether they are to the left or the right of the camera path.

had no prior experience with virtual reality (7 out of 12) or 360 videos (8 out of 12). Two participants were experienced 360 video and virtual reality content consumers.

5.7 Apparatus

The participants completed the study with their personal computers and used an external mouse as the input device. To control the effect of display size on performance, we asked participants to run the study software in the full-screen mode on a 13"-16" display (M_{size} = 14.35, SD_{size} = 1.02). All the computers that the participants used for the study met the requirements of the study software.

6 EVALUATION RESULTS AND ANALYSIS

We compared the task performance of the three *interfaces* in terms of completion time and failure rate. Additionally, we analyzed the participants' input traces to understand their usage patterns and potential causes of the perforamnce differences. The results are reported below.

6.1 Task Performance

We computed average task completion time and number of failure trials to compare the participants' performance using the three *interfaces*. Note that failure trials, i.e. those that did not receive a correct response before the 4-minute timeout (50 out of 504, 9.9%) were not included in task completion time calculation.

6.1.1 Task Completion Time. A repeated measures ANOVA (RM-ANOVA) on task completion time showed a significant difference between the three *interfaces* ($F_{2,22} = 28.90, p < 0.01$). Post-hoc tests[2] showed that *Route Tapestry* was significantly faster than *equirectangular* and that *equirectangular* was significantly faster than *Little Planet* (Figure 12, left). Overall, the participants took 21.2% and 40.5% less time completing the tasks using *Route Tapestry* ($M = 59.2s$, $SD = 13.5s$) than *equirectangular* and *Little Planet*, respectively. Task completion time for *equirectangular* ($M = 75.1s$, $SD = 19.2s$) was 24.5% lower than *Little Planet* ($M = 99.5s$, $SD = 18.6s$).

[2] All post-hoc tests for RM-ANOVA used paired t-test with Bonferroni-Holm correction.

Table 1. Target Location Statistics by Video

Video	Avg. Valid Segment Length	Avg. Position in Video
Video 1	8.2s (±3.6s)	8m04s (±3m:10s)
Video 2	8.1s (±3.8s)	7m24s (±3m:54s)
Video 3	8.3s (±2.8s)	7m43s (±5m:23s)

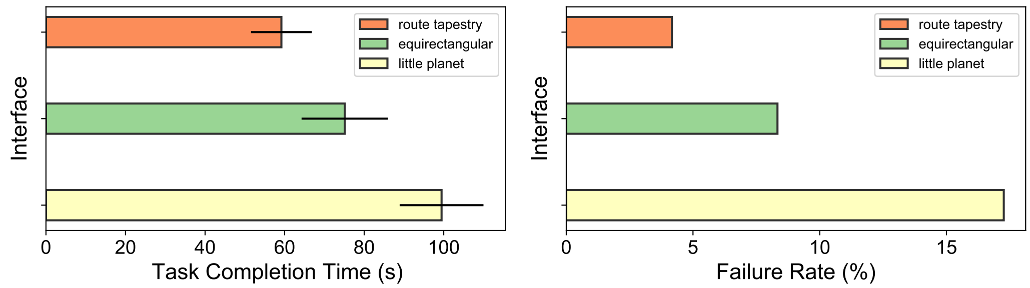

Fig. 12. Average task completion time (*left*) and failure rate (*right*) by interface. Error bars represent 0.95 CI.

6.1.2 Failure Rate. Overall, *Little Planet* had the most failure trails (29, 17.3%), followed by *equirectangular* (14, 8.3%) and then *Route Tapestry* (7, 4.2%). We compared failure rate for the three *interfaces*.

A Friedman test on failure rate found a significant difference between *interfaces* ($\chi^2 = 11.5$, $p < 0.01$). Post-hoc pairwise comparison using paired Wilcoxon tests showed that *Route Tapestry* ($M = 4.2\%$, $SD = 4.8\%$) had significantly lower failure rate than *Little Planet* ($M = 17.3\%$, $SD = 9.4\%$). On average, the failure rate of *Route Tapestry* was about the half as *equirectangular*, and the *equirectangular* condition had less than the half of the failure rate of *Little Planet* (Figure 12, right).

6.2 Intra-Condition Learning Effect

Since the presentation order of target scenes were randomized, the participants could have been exposed to other target scense while searching for the current one. Therefore, we would expect that as they found more target scenes, they could gradually build knowledge about the video and use it to accelerate later trials. We partitioned the 14 trials per condition into three groups based on their *presentation order*—the first five (*early*), the middle four (*middle*), and last five (*late*)—and compared their task completion time to study this effect.

A two-way RM-ANOVA on task completion time using *interface* and *presentation order* as the within-subject factor showed significant main effects for both *interface* ($F_{2,22} = 25.9$, $p < 0.01$) and *presentation order* ($F_{2,22} = 8.7$, $p < 0.01$), and no significant interaction between the two. Post-hoc tests showed that tasks in the *early* group were completed significantly slower than those in the *middle* and *late* group. In Figure 13 we can see that *Little Planet* had much lower initial performance ($M = 121.3s$, $SD = 39.7s$) than *equirectangular* ($M = 86.8s$, $SD = 21.4s$) and *Route Tapestry* ($M = 65.0s$, $SD = 25.7s$) but gained notable improvement for tasks in both *middle* (12.0%) and *late* (33.0%) groups. In contrast, *equirectangular* and *Route Tapestry* conditions only saw improvements from *early* to *middle* but not any further. A possible explanation to this observation is that the participants took more trials to learn the video content with *Little Planet* than with *Route Tapestry* or *equirectangular*. Additional studies may be needed to understand the impact of each technique on long term comprehension of video content.

6.3 Analysis of Participants' Interaction Traces

We further conducted an analysis on user input traces (i.e. pointer movement and clicks) to understand usage patterns and identify potential factors behind the performance gaps between the three *interfaces*. This analysis explored how participants reoriented the viewing angle (necessary since the video player does not show all 360 degrees of the video at the same time) across the three conditions, how they used the preview thumbnails and dragging across the timeline,

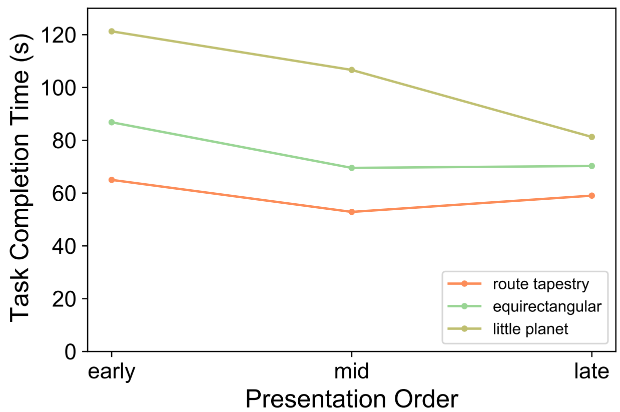

Fig. 13. Average task completion time of trials presented *early* (1-5), *middle* (6-9), and *late* (10-14), by *interface*

and how frequently they would "pause" while browsing. Our analysis here explores all the data collected in the study, including from trials that ended with timeout.

6.3.1 Viewing Direction Change. As the participants needed to have the target scene in their field-of-view to complete a task, it was possible that the difference in time spent on viewing direction manipulation contributed to the performance gap. The direction snapping feature of *Route Tapestry*—immediate reorientation when the user clicked on one of the Tapestry timelines—may gave it a strong advantage over *equirectangular* and *Little Planet*. To better understand this factor, we calculated the total time spent on camera manipulation in success trials with the three *interfaces*, from the records of viewing direction change.

We found that although the average viewing direction manipulation time per trial was shorter with *Route Tapestry* (M = 3.85s, SD = 1.94s) than *equirectangular* (M = 5.57s, SD = 3.33s) and *Little Planet* (M = 7.75s, SD = 4.87s), it was not a major component in the full task completion time, and the proportions were similar for all three *interfaces* (6%-8%). RM-ANOVA and post-hoc pairwise tests still showed the same trends on task completion time with viewing direction manipulation time removed (RM-ANOVA: $F_{2,22} = 27.0$, $p < 0.01$). This suggested that the performance gaps were more likely due to differences in the time spent on temporal navigation between the *interfaces*.

6.3.2 Pointer Input Modality: Drag/Hover. With all three *interfaces*, the participants had the choice of hovering the pointer over the timeline to see the small frame preview or dragging the playhead slider to see the large preview. We tracked pointer hovering and dragging traces on the timelines separately and used this data to study which modality the participants decided to employ with different *interfaces*. Both hovering and dragging traces were timestamped and updated every 100ms.

We found a sharp contrast between *Route Tapestry* and the other two conditions (Figure 14): hovering was common with *Route Tapestry* (89.0%) but only used about a quarter of the time for *equirectangular* (25.5%) and *Little Planet* (27.0%). During the study, some participants commented on the physical fatigue due to dragging and their stronger preference to hovering. The modality choices suggested that in comparison to *Route Tapestry*, the hover-triggered small previews were less useful for the participants when using *equirectangular* and *Little Planet*, driving them to use dragging, a potentially more physically-demanding modality.

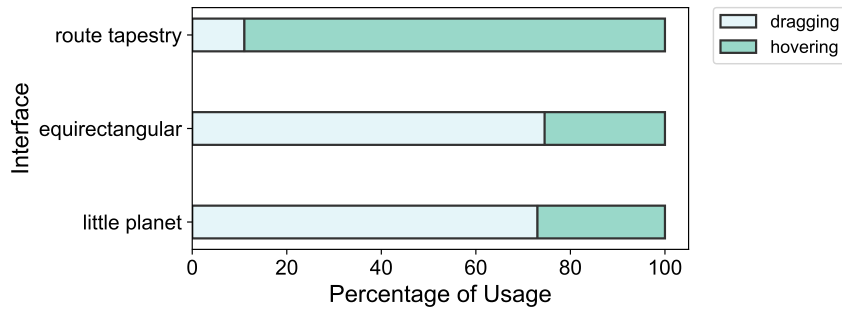

Fig. 14. Percentage of time of pointer hovering vs pointer dragging on the timelines, by interface.

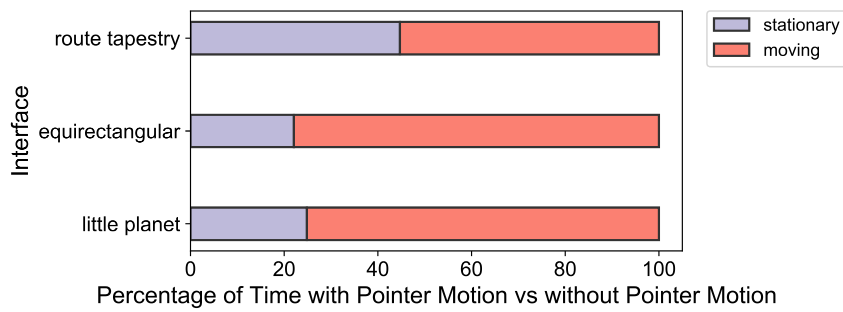

Fig. 15. Percentage of task completion time with pointer motion vs. with no pointer motion, by interface.

6.3.3 Pauses during Browsing. We noticed that the participants tended to move the pointers in short flicks with *Route Tapestry* but in long and slow strokes with *equirectangular* and *Little Planet*. Our analysis of the trace data confirms this. First, we calculated pointer speed between consecutive trace records for both pointer hovering and dragging, and obtained the times for when the pointer was moving or stationary for the three *interfaces*. We found that the participants stopped pointer movement, for a longer period of time in proportion to the total task completion time for *Route Tapestry* than *equirectangular* and *Little Planet*. A Friedman tests on the ratio between the time without any pointer movement and the total task completion time per trial showed a significant effect of the within-subject factor *interface* ($\chi^2 = 13.2$, $p < 0.01$). Further post-hoc tests confirmed that the pointer motion was paused significantly less in proportion for both *Little Planet* ($M = 22.1\%$, $SD = 15.0\%$) and *equirectangular* ($M = 24.8\%$, $SD = 15.1\%$) than *Route Tapestry* ($M = 44.7\%$, $SD = 13.9\%$). Short pauses, i.e. those over 500ms, took 74.4% of all pause time for *Route Tapestry*, signaling that most of the pauses were brief.

The analysis partially confirmed our hypothesis by showing that the pointer movements when using *Route Tapestry* were interleaved with more short pauses. We further verified the hypothesis by studying the pointer movement speed, which will be discussed below.

6.3.4 Average Pointer Speed. We examined the average pointer speed with different *interfaces* when the pointer was in motion. To account for both pointer dragging and hovering, we defined the *weighted pointer speed* $\bar{v}$ of an *interface* condition, which was the average of the mean pointer dragging speed v_d and hovering speed v_h, weighted by their respective time duration t_d and t_h

$$\bar{v} = (v_d t_d + v_h t_h)/(t_d + t_h)$$

For every participant and every condition, we used consecutive timestamped pointer positions to calculate v_d, t_d and v_h, t_h. Note that t_h and t_d did not include the time when the pointer was still. Extremely fast flicks, which were likely not for video browsing but pointer re-positioning, were removed with a threshold of 500px/s. All pointer traces had been scaled to a 1920×1080 window before calculation.

A Friedman test on weighted pointer speed showed a significant effect of *interface* ($\chi^2 = 13.2$, $p < 0.01$). Post-hoc pairwise tests found that the weighted pointer speed for *Route Tapestry* ($M = 60.3$px/s, $SD = 23.2$px/s) was significantly faster than *equirectangular* ($M = 42.5$px/s, $SD = 12.9$px/s) and *Little Planet* ($M = 36.2$px/s, $SD = 13.3$px/s). This result (Figure 16), in combination with the findings about pauses in the pointer movements, suggested that the participants were indeed using more short flicks with *Route Tapestry* on the timelines, whereas with *equirectangular* and *Little Planet* they tended to move the pointer in a slower and more continuous manner.

7 DISCUSSION

Based on the analysis of the user performance and interaction patterns of Route Tapestries and the two baselines, we discuss further implication of the study results, limitations of this work, and a few promising future directions.

7.1 Study Results

Overall, the study results show that for scene-finding tasks in 360 virtual tour videos, participants were faster and more efficient using *Route Tapestry* compared to the two baselines that used equirectangular and Little Planet previews. A number of factors might have led to these differences.

Our data showed that the participants' pointer motion followed different patterns when using the Route Tapestry timelines from the two other interfaces. The movements on the Route Tapestry timelines were faster and interleaved with short pauses, suggesting that the participants tended to browse the Tapestry one section (e.g. a building, part of a block, etc.) at time. On the two other timelines, however, they had to rely on slower, continuous movements to closely examine individual frame previews. As Route Tapestries present the summaries of the scenes along the route within a

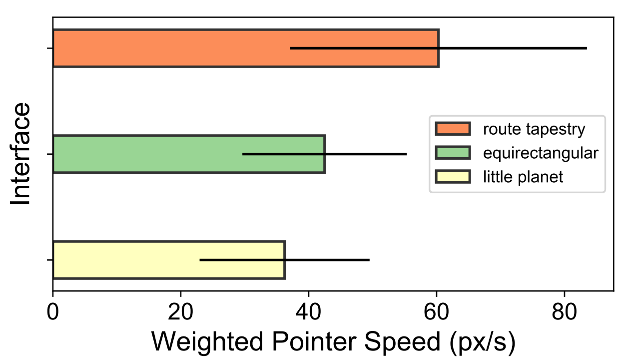

Fig. 16. Weighted pointer speed by interface. Error bars represents standard error.

continuous context, search could be done by inspecting the summary rather than frame by frame. It also allowed them to skip entire section of irrelevant scenes altogether, leading to a higher level of efficiency. This finding further lends support to our initial rationale for using a visual-summary-based approach for navigation.

While both equirectangular and Little Planet previews contain strong distortion, recognizing landscapes and architectural features turned out to be less demanding with equirectangular. For typical 360 virtual tour videos, the Little Planet projection stretches objects located higher above the camera and compresses those at the camera level, where more recognizable features of landscapes tend to be found. Moreover, the sky or ceiling is projected to an disproportionately large area in an Little Planet preview image. While this creates an appealing visual style and gives a good sense of object height, it reduces the space on the projected image for the ground environment, which is usually more relevant to location seeking. In contrast, the equirectangular projection expands the objects at the camera level when the camera passes them, making recognizing location features easier (see Figure 2).

The distortion in both equirectangular and Little Planet frame previews makes them hard to interpret. Therefore, the participants tended to rely on the larger preview shown during scrubbing for navigation, despite dragging the slider added to their physical effort.

7.2 Limitations and Opportunities for Improving Route Tapestries

Our design, implementation, and evaluation of Route Tapestries were subject to a number of limitations.

Since our Route Tapestry generation algorithm is based on slit-scan imaging, it requires a moving camera and does not effectively summarize static shots, which are common in other kinds of 360 videos, especially 360 films. Other video summarization techniques visualize changes in static shots with motion trajectories [6] or arrows [10]. In the future, we would like to combine these techniques with slit-scan imaging to accommodate both static and dynamic shots and support navigation for a wider range of 360 videos beyond virtual tours.

We noted that as videos get longer, the information provided by the global low-resolution Tapestry strips become limited to more obvious patterns such as camera stops, occurrences of large buildings, or building density. Inspired by prior work on video navigation through maps [27], we believe an opportunity to further improve the efficiency of Route Tapestries is to annotate the global Tapestry strips with semantic labels such as building boundaries, or significant patterns in the scenes. As such, the timelines can support more efficient focus+context search [9].

Our algorithm is robust to a reasonable range of scene depth and camera speed variations, but would produce noticeable scene compression or stretch when the changes are significant, e.g. entering a small shop from a wide street or cycling after walking. In future work, we plan to apply computer vision methods to make the method more adaptive. More details will be discussed in 7.3.2.

7.3 Future Work

Our exploration of Route Tapestries suggests several exciting new design opportunities, including adapting them for HMD interaction and automating route tapestry generation.

7.3.1 Adapting Route Tapestries for HMD Interactions . Watching 360 virtual tour videos in head-mounted displays (HMDs) offers a strong sense of immersion and presence in the remote space. The current 360 video player interface designs for HMDs are similar to their desktop versions. Most players use a linear timeline, which displays an equirectangular frame thumbnail upon cursor hovering. The player interfaces of mainstream platforms (YouTube, Oculus Theatre) do not yet support scrubbing. Therefore, the current HMD 360 video player could have similar navigation issues as their desktop versions face.

The design of the desktop Tapestry player can be applied to HMD with a few small modifications. To better match the immersive viewing experience, the Route Tapestries can be rendered in a circular rather than linear manner, and possibly surrounding the viewer. Direction snapping, i.e., an immediate reorientation towards the location selected in the Route Tapestry, can be removed or slowed down to prevent motion sickness.

Even more exciting opportunities lie in redesigning Tapestry Player to fully leveraging the 3D interaction space offered by HMDs. The Route Tapestries can be positioned to match the scenes' spatial layout that they depict and promote more vital spatial awareness. For the driving tour videos used in the study, the two tapestries showing the landscapes on the left and the right sides of the street can be placed on their corresponding sides of the viewer, extending forward. The sizeable available space in 3D also allows the displayed Route Tapestries to include more objects that are much higher or lower than the camera, such as the top of a building.

7.3.2 Automating Route Tapestry Generation. Our current Route Tapestry generation procedure requires the user to provide a small number of parameters to ensure visual quality. In future work, we plan to use computer vision methods to automate Route Tapestry generation further and make the results better adapt to a broader range of video and scene types.

One option is to detect the great circle with the most massive optical flow displacement between consecutive spherical frames and generate a slit-scan 'tube' using the great circles' pixels. The sampling rate can also be directly derived from the optical flow displacement values. However, this method cannot adapt to the scene depth variance within a frame, especially along the great circle. Recent computer vision research has explored depth estimation from 360 videos [44]. Depth information could be used to find the foreground of the scene, which should (i) be relevant for video navigation, and (ii) have an overall consistent depth. Route Tapestries of better quality can then be generated from only the pixels in these foreground areas. Other multi-perspective imaging techniques, such as cross-slit imaging [34], can also be applied to improve the resulting tapestry quality.

8 CONCLUSION

In this paper, we introduce Route Tapestries, a method for navigating 360 virtual tour videos using slit-scan visualizations. People adopt 360 virtual tour videos to visit places remotely, but they face challenges when searching for their desired scenes in these videos with current interfaces, which primarily support navigation through 2D video frame previews. These previews are created through various sphere-to-plane projection methods, such as equirectangular and Little Planet, and contain significant visual distortion.

Inspired by research on using video content summary for navigation, we propose browsing and navigating 360 virtual tour videos through their continuous visual summaries, which we call Route Tapestries. We present an algorithm for generating Route Tapestries using the slit-scan imaging technique, and the design and evaluation of Tapestry Player, a desktop 360 video player incorporating Route Tapestries timelines. We conducted a user study comparing Tapestry Player with two alternative designs which used equirectangular and Little Planet previews, using a scene seeking task. The study results showed that the users were more efficient when using Tapestry Player and employed less physically demanding input modality. Our further analysis revealed that with Route Tapestries, the users were able to achieve higher efficiency through previewing video one section a time, rather than frame-by-frame.

We hope Route Tapestries can inspire 360 video player designs that make virtual tours more enjoyable for people who want to visit remote places but choose not to do so physically because of time, health, cost, or any other reasons. In our future work, we plan to extend the Route Tapestry approach for navigating videos consisting of a wider variety of shot types, explore Route Tapestries for HMDs, and develop automatic, fully automated tapestry generation algorithms.

REFERENCES

[1] Aseem Agarwala, Maneesh Agrawala, Michael Cohen, David Salesin, and Richard Szeliski. 2006. Photographing long scenes with multi-viewpoint panoramas. In *ACM SIGGRAPH 2006 Papers*. 853–861.

[2] Werner Bailer, Christian Schober, and Georg Thallinger. 2006. Video Content Browsing Based on Iterative Feature Clustering for Rushes Exploitation.. In *TRECVID*. Citeseer.

[3] Connelly Barnes, Dan B Goldman, Eli Shechtman, and Adam Finkelstein. 2010. Video tapestries with continuous temporal zoom. In *ACM SIGGRAPH 2010 papers*. 1–9.

[4] Wutthigrai Boonsuk, Stephen Gilbert, and Jonathan Kelly. 2012. The impact of three interfaces for 360-degree video on spatial cognition. In *Proceedings of the SIGCHI conference on human factors in computing systems*. 2579–2588.

[5] Seunghoon Cha, Jungjin Lee, Seunghwa Jeong, Younghui Kim, and Junyong Noh. 2020. Enhanced Interactive 360° Viewing via Automatic Guidance. *ACM Transactions on Graphics (TOG)* 39, 5 (2020), 1–15.

[6] Carlos D. Correa and Kwan-Liu Ma. 2010. Dynamic Video Narratives. *ACM Trans. Graph.* 29, 4, Article 88 (July 2010), 9 pages. https://doi.org/10.1145/1778765.1778825

[7] Anastasios D Doulamis and Nikolaos D Doulamis. 2004. Optimal content-based video decomposition for interactive video navigation. *IEEE Transactions on Circuits and Systems for Video Technology* 14, 6 (2004), 757–775.

[8] Pierre Dragicevic, Gonzalo Ramos, Jacobo Bibliowitcz, Derek Nowrouzezahrai, Ravin Balakrishnan, and Karan Singh. 2008. Video browsing by direct manipulation. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. 237–246.

[9] Susan Dumais, Edward Cutrell, and Hao Chen. 2001. Optimizing search by showing results in context. In *Proceedings of the SIGCHI conference on Human factors in computing systems*. 277–284.

[10] Dan B Goldman, Brian Curless, David Salesin, and Steven M Seitz. 2006. Schematic storyboarding for video visualization and editing. *Acm transactions on graphics (tog)* 25, 3 (2006), 862–871.

[11] Jeremy Hartmann, Stephen Diverdi, Cuong Nguyen, and Daniel Vogel. 2020. View-Dependent Effects for 360° Virtual Reality Video. In *Proceedings of the 33rd Annual ACM Symposium on User Interface Software and Technology - UIST '20*. ACM. https://doi.org/10.1145/3379337.3415846

[12] Wolfgang Hürst. 2006. Interactive audio-visual video browsing. In *Proceedings of the 14th ACM international conference on Multimedia*. 675–678.

[13] Dan Jackson, James Nicholson, Gerrit Stoeckigt, Rebecca Wrobel, Anja Thieme, and Patrick Olivier. 2013. Panopticon: A parallel video overview system. In *Proceedings of the 26th annual ACM symposium on User interface software and technology*. 123–130.

[14] Kyoungkook Kang and Sunghyun Cho. 2019. Interactive and automatic navigation for 360° video playback. *ACM Transactions on Graphics (TOG)* 38, 4 (2019), 1–11.

[15] Juho Kim, Philip J Guo, Carrie J Cai, Shang-Wen Li, Krzysztof Z Gajos, and Robert C Miller. 2014. Data-driven interaction techniques for improving navigation of educational videos. In *Proceedings of the 27th annual ACM symposium on User interface software and technology*. 563–572.

[16] Don Kimber, Jonathan Foote, and Surapong Lertsithichai. 2001. Flyabout: spatially indexed panoramic video. In *Proceedings of the ninth ACM international conference on Multimedia*. 339–347.

[17] Johannes Kopf, Billy Chen, Richard Szeliski, and Michael Cohen. 2010. Street slide: browsing street level imagery. *ACM Transactions on Graphics (TOG)* 29, 4 (2010), 1–8.

[18] Yung-Ta Lin, Yi-Chi Liao, Shan-Yuan Teng, Yi-Ju Chung, Liwei Chan, and Bing-Yu Chen. 2017. Outside-in: Visualizing out-of-sight regions-of-interest in a 360 video using spatial picture-in-picture previews. In *Proceedings of the 30th Annual ACM Symposium on User Interface Software and Technology*. 255–265.

[19] Justin Matejka, Tovi Grossman, and George Fitzmaurice. 2012. Swift: reducing the effects of latency in online video scrubbing. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. 637–646.

[20] Justin Matejka, Tovi Grossman, and George Fitzmaurice. 2013. Swifter: improved online video scrubbing. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. 1159–1168.

[21] Michael Mills, Jonathan Cohen, and Yin Yin Wong. 1992. A magnifier tool for video data. In *Proceedings of the SIGCHI conference on Human factors in computing systems*. 93–98.

[22] Luís AR Neng and Teresa Chambel. 2010. Get around 360 hypervideo. In *Proceedings of the 14th International Academic MindTrek Conference: Envisioning Future Media Environments*. 119–122.

[23] Cuong Nguyen, Stephen DiVerdi, Aaron Hertzmann, and Feng Liu. 2017. CollaVR: Collaborative in-headset review for VR video. In *Proceedings of the 30th Annual ACM Symposium on User Interface Software and Technology*. 267–277.

[24] Cuong Nguyen, Stephen DiVerdi, Aaron Hertzmann, and Feng Liu. 2017. Vremiere: in-headset virtual reality video editing. In *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems*. 5428–5438.

[25] Cuong Nguyen, Yuzhen Niu, and Feng Liu. 2012. Video summagator: an interface for video summarization and navigation. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. 647–650.

[26] Cuong Nguyen, Yuzhen Niu, and Feng Liu. 2013. Direct manipulation video navigation in 3D. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. 1169–1172.

[27] Gonçalo Noronha, Carlos Álvares, and Teresa Chambel. 2012. Sight surfers: 360º videos and maps navigation. In *Proceedings of the ACM multimedia 2012 workshop on Geotagging and its applications in multimedia*. 19–22.

[28] Michael Nunes, Saul Greenberg, Sheelagh Carpendale, and Carl Gutwin. 2007. What did I miss? Visualizing the past through video traces. In *ECSCW 2007*. Springer, 1–20.

[29] Amy Pavel, Dan B. Goldman, Björn Hartmann, and Maneesh Agrawala. 2015. SceneSkim: Searching and Browsing Movies Using Synchronized Captions, Scripts and Plot Summaries. In *Proceedings of the 28th Annual ACM Symposium on User Interface Software amp; Technology* (Charlotte, NC, USA) *(UIST '15)*. Association for Computing Machinery, New York, NY, USA, 181–190. https://doi.org/10.1145/2807442.2807502

[30] Amy Pavel, Björn Hartmann, and Maneesh Agrawala. 2017. Shot orientation controls for interactive cinematography with 360 video. In *Proceedings of the 30th Annual ACM Symposium on User Interface Software and Technology*. 289–297.

[31] Benjamin Petry and Jochen Huber. 2015. Towards effective interaction with omnidirectional videos using immersive virtual reality headsets. In *Proceedings of the 6th Augmented Human International Conference*. 217–218.

[32] Suporn Pongnumkul, Jue Wang, Gonzalo Ramos, and Michael Cohen. 2010. Content-aware dynamic timeline for video browsing. In *Proceedings of the 23nd annual ACM symposium on User interface software and technology*. 139–142.

[33] Gonzalo Ramos and Ravin Balakrishnan. 2003. Fluid interaction techniques for the control and annotation of digital video. In *Proceedings of the 16th annual ACM symposium on User interface software and technology*. 105–114.

[34] Augusto Roman, Gaurav Garg, and Marc Levoy. 2004. Interactive design of multi-perspective images for visualizing urban landscapes. In *IEEE visualization 2004*. IEEE, 537–544.

[35] Gustavo Alberto Rovelo Ruiz, Davy Vanacken, Kris Luyten, Francisco Abad, and Emilio Camahort. 2014. Multi-viewer gesture-based interaction for omni-directional video. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. 4077–4086.

[36] Klaus Schoeffmann, Mario Taschwer, and Laszlo Boeszoermenyi. 2010. The video explorer: a tool for navigation and searching within a single video based on fast content analysis. In *Proceedings of the first annual ACM SIGMM conference on Multimedia systems*. 247–258.

[37] Vincent Sitzmann, Ana Serrano, Amy Pavel, Maneesh Agrawala, Diego Gutierrez, Belen Masia, and Gordon Wetzstein. 2018. Saliency in VR: How do people explore virtual environments? *IEEE transactions on visualization and computer graphics* 24, 4 (2018), 1633–1642.

[38] Yu-Chuan Su and Kristen Grauman. 2017. Making 360 video watchable in 2d: Learning videography for click free viewing. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, 1368–1376.

[39] Yu-Chuan Su, Dinesh Jayaraman, and Kristen Grauman. 2016. Pano2Vid: Automatic Cinematography for Watching 360 Videos. In *Asian Conference on Computer Vision*. Springer, 154–171.

[40] Anthony Tang, Saul Greenberg, and Sidney Fels. 2008. Exploring video streams using slit-tear visualizations. In *Proceedings of the working conference on Advanced visual interfaces*. 191–198.

[41] Yukinobu Taniguchi, Akihito Akutsu, and Yoshinobu Tonomura. 1997. PanoramaExcerpts: extracting and packing panoramas for video browsing. In *Proceedings of the fifth ACM international conference on Multimedia*. 427–436.

[42] Anh Truong and Maneesh Agrawala. 2019. A Tool for Navigating and Editing 360 Video of Social Conversations into Shareable Highlights.. In *Graphics Interface*. 14–1.

[43] Anh Truong, Sara Chen, Ersin Yumer, David Salesin, and Wilmot Li. 2018. Extracting regular fov shots from 360 event footage. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems*. 1–11.

[44] Fu-En Wang, Hou-Ning Hu, Hsien-Tzu Cheng, Juan-Ting Lin, Shang-Ta Yang, Meng-Li Shih, Hung-Kuo Chu, and Min Sun. 2018. Self-supervised Learning of Depth and Camera Motion from 360 Videos. In *Asian Conference on Computer Vision*. Springer, 53–68.

[45] Jiang Yu Zheng. 2003. Digital route panoramas. *IEEE MultiMedia* 10, 3 (2003), 57–67.