CoRE: Non-Linear 3D Sampling for Robust 360° Video Streaming

Mijanur R. Palash
Purdue University
mpalash@purdue.edu

Voicu Popescu
Purdue University
popescu@purdue.edu

Amit Sheoran
Purdue University
asheoran@purdue.edu

Sonia Fahmy
Purdue University
fahmy@cs.purdue.edu

***Abstract*—CoRE is an approach for streaming 360° videos based on a non-linear sampling of the equirectangular video cube. CoRE is robust to view prediction errors.**

***Index Terms*—360° video streaming.**

I. Introduction

360° videos are gaining popularity as they offer an immersive experience. Content providers like Facebook and YouTube provide 360° videos for clients to stream, but this requires high network bandwidth and client processing power. The transfer and processing requirements can be reduced by only sending to the client the part of the video viewed by the user. However, sending each frame incurs the prohibitive latency of traversing the network twice per frame, to request and to receive the frame. A promising approach is to transfer only and all the 360° video data required for the next sequence of frames, which is then rendered at the client, bypassing network latency. Two problems must be overcome. The first problem is view direction prediction, to determine which part of the 360° video is needed for the next sequence, e.g., via machine learning; to mitigate missing data artifacts when prediction fails, Flare [1] pads the prediction, whereas Rubiks [2] transfers an additional complete 360° video at a lower resolution. The second problem is 360° video partitioning, to allow extracting the needed part. Partitioning typically uses tiling [1]–[3].

Tiling brings flexibility to carving out the needed part of a 360° video, and a tile is essentially a small video that can be handled by existing video codec (e.g., H264) and transfer (e.g., DASH) infrastructure. However, tiling has several disadvantages. *Decoding overhead*: the client has to decode tens of tiles, which, even though a tile is small, implies computational and power consumption overhead. *Compression inefficiency*: compressing individual tiles fails to take advantage of data redundancy across tiles, increasing the required bandwidth. *Partitioning coarseness*: increasing tile size reduces the number of tiles which alleviates some tiling disadvantages, but large tiles preclude a high-fidelity approximation of the predicted part of the 360° video, which results in transferring unnecessary data. *Discontinuous frame quality*: rendering the frame at the client from tiles of various resolutions creates objectionable abrupt changes in image quality. *Transfer scheduling complexity*: managing the set of tiles to be transferred to keep up with changes in view direction prediction is challenging. *Non-uniform partitioning*: a uniform tiling of an equirectangular 360° video translates to a non-uniform partitioning of the unit sphere, which increases the number of tiles needed to cover tilted view directions, and decreases quality for horizontal view directions.

This work has been funded in part by NSF grant CNS-1717493.

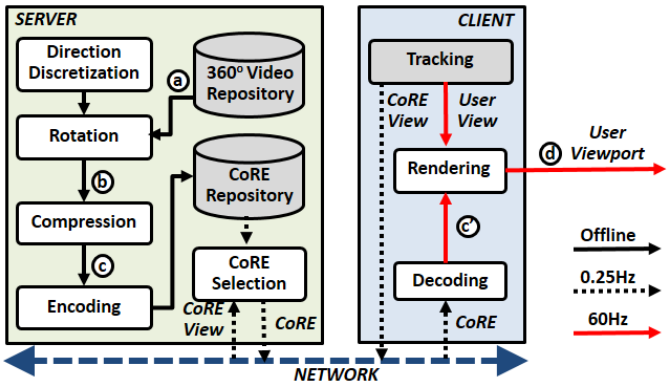
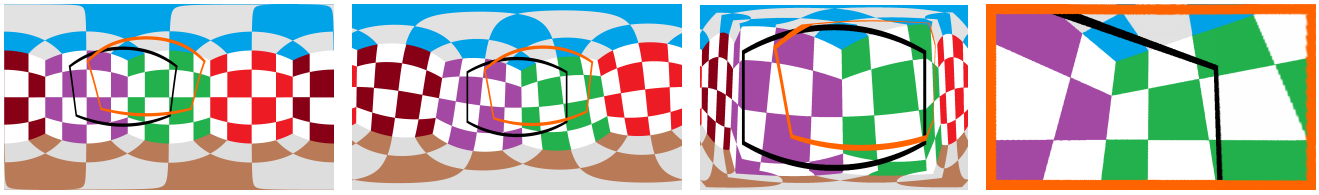


Fig. 1: CoRE architecture.

In this poster, we introduce CoRE, a non-linear 3D video sampling method for robust 360° video streaming, inspired by earlier work on non-uniform sampling of 2D images [4]. Given a 360° video V, a range of view direction pan and tilt angles $[\theta_0, \theta_1] \times [\phi_0, \phi_1]$, and a time interval $[t_0, t_1]$, we build a non-linearly sampled CoRE video that covers the 3D subvolume S=$[\theta_0, \theta_1]$×$[\phi_0, \phi_1]$×$[t_0, t_1]$ at full quality, but also covers $V-S$, at lower quality. Quality is uniformly high inside S, and decreases gradually with distance from S, which avoids the frame quality discontinuity of tiling. A CoRE video covers all view directions, albeit at a lower spatial sampling rate, which brings robustness to view direction prediction errors. A CoRE video covers time points beyond t_1, albeit at a lower frame rate, which avoids stalls when network conditions delay receiving data for the next user frame sequence. Finally, CoRE transfers a single file for each time interval, avoiding the decoding overhead, compression inefficiency, and scheduling complexity of tiling, at the expense of storage at the server.

II. Approach

As depicted in Fig. 1, the server builds a set of CoRE videos for each 360° video, offline (1). Then, once the client connects, the server provides every few seconds a CoRE video for rendering an entire sequence of user frames at 60Hz (2).



(a) Original 360° video frame (b) Rotated 360° video frame (c) CoRE frame (magnif.) (d) Output frame

Fig. 2: Sample output for the stages of the CoRE pipeline from Fig. 1 on a checkered cube scene. The black outline shows the viewport for which the CoRE was constructed, and the orange outline shows the viewport of the output frame.

(1) Given a 360° video, the server builds one CoRE video for each direction and each time interval. For example, for a 400s 360° video, with a $20° \times 20°$ direction discretization and a 4s time interval, $18 \times 9 \times 100$ CoRE videos are built. A CoRE video is built one frame at the time. Given a direction (θ, ϕ), a time step t, and the horizontal and vertical fields of view (h, v) of the viewport, a **Co**mpressed **R**otated **E**quirectangular (**CoRE**) frame is built from the 360° video frame t ((a) in Fig. 1 and 2) by rotating it to center (θ, ϕ) ((b) in the figures), and by compressing the parts outside (h, v) non-linearly (c). The CoRE video is encoded conventionally (e.g., H264). The video has a variable frame rate, with full frame rate for its time interval and decreasing frame rate beyond.

(2) CoRE works with view direction prediction, when the client requests a CoRE video for the predicted view, or without prediction, when the client requests a CoRE video for the current view. The server sends the best matching CoRE video. The client renders the output frame (Fig. 2d) on the CPU, by resampling the CoRE frame, or on the GPU, by rendering a sphere mapped with a CoRE video texture.

III. Preliminary Results

We evaluated CoRE via simulations on 4k equirectangular 360° videos with pre-recorded HMD data traces. The output frame had a 110° horizontal field of view and a 16:9 aspect ratio. Compression reduces the $3,840 \times 2,160$ input resolution to a CoRE resolution of $1,706 \times 1,175$, by keeping the construction viewport at the original resolution, and by shrinking the outer parts non-linearly with an application-chosen average compression factor, here $5\times$. Fig. 2c magnifies the CoRE frame for illustration purposes.

Fig. 3 illustrates the CoRE non-linear sampling. One graph (left y axis) shows the mapping from the CoRE frame to the uncompressed frame, which changes quadratically over the compressed left and right boundary regions, and linearly over the uncompressed central region. The second graph (right y axis) shows the derivative of the first graph, and corresponds to the sampling step in the uncompressed image. As expected, the sampling step decreases (increases) linearly over the left (right) boundary regions, and it is exactly one for the uncompressed central region. The maximum (average) sampling step is 9 (5). In addition to avoiding quality discontinuities, the CoRE non-linear sampling yields higher quality close to the inner edge of a boundary region, compared to its outer edge. This translates to higher quality output frames when the output viewport extends slightly beyond the CoRE construction viewport. In other words, the quality penalty for an incorrect view direction prediction is often small in CoRE, and always commensurate to the error magnitude.

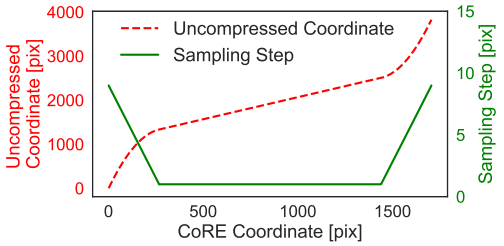


Fig. 3: Sampling over the equator of a CoRE frame.

Fig. 4 shows that CoRE transfers less data, even compared to 10×10 tiling. In practice, decoding tens of tiles on the client is prohibitively expensive, and applications resort to fewer, larger tiles. Flare [1] employs 4×6 1s tiling, which transfers 60% more than CoRE, even without including the padding needed by tiling to compensate for prediction inaccuracy.

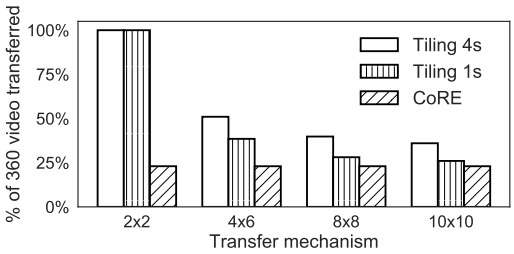


Fig. 4: Comparison of transferred data amount.

Our future work plans include completing the CoRE system for hand-held and head-mounted display interactive viewing.

References

[1] F. Qian, B. Han, Q. Xiao, and V. Gopalakrishnan, "Flare: Practical viewport-adaptive 360-degree video streaming for mobile devicesdownload," in *Proceedings of MOBICOM*, 2018.

[2] J. He, M. A. Qureshi, L. Qiu, J. Li, F. Li, and L. Han, "Rubiks: Practical 360-degree streaming for smartphones," in *Proceedings of MobiSys*, 2018.

[3] Y. Guan, C. Zheng, X. Zhang, Z. Guo, and J. Jiang, "Pano: Optimizing 360 video streaming with a better understanding of quality perception," in *Proceedings of SIGCOMM*, 2019.

[4] V. Popescu, P. Rosen, L. Arns, X. Tricoche, C. Wyman, and C. M. Hoffmann, "The general pinhole camera: Effective and efficient nonuniform sampling for visualization," *IEEE transactions on visualization and computer graphics*, vol. 16, no. 5, pp. 777–790, 2010.