Rolling Shutter and Radial Distortion are Features for High Frame Rate Multi-camera Tracking

Akash Bapat, True Price, Jan-Michael Frahm
Department of Computer Science, The University of North Carolina at Chapel Hill
{akash,jtprice,jmf}@cs.unc.edu

Abstract

Traditionally, camera-based tracking approaches have treated rolling shutter and radial distortion as imaging artifacts that have to be overcome and corrected for in order to apply standard camera models and scene reconstruction methods. In this paper, we introduce a novel multi-camera tracking approach that for the first time jointly leverages the information introduced by rolling shutter and radial distortion as a feature to achieve superior performance with respect to high-frequency camera pose estimation. In particular, our system is capable of attaining high tracking rates that were previously unachievable. Our approach explicitly leverages rolling shutter capture and radial distortion to process individual rows, rather than entire image frames, for accurate camera motion estimation. We estimate a per-row 6 DoF pose of a rolling shutter camera by tracking multiple points on a radially distorted row whose rays span a curved surface in 3D space. Although tracking systems for rolling shutter cameras exist, we are the first to leverage radial distortion to measure a per-row pose – enabling us to use less than half the number of cameras required by the previous state of the art. We validate our system on both synthetic and real imagery.

1. Introduction

The recent resurgence of Augmented Reality and Virtual Reality (AR/VR) has provided challenging computer vision problems of drastically higher requirements. For example, recent state-of-the-art low-latency rendering techniques [38, 24] assume that very high-frequency (>30 kHz) tracking is available. However, commercial AR/VR systems like Google's Daydream and Cardboard products and Samsung VR, provide only rotational tracking, which limits the immersive experience. The Oculus Rift and the HTC Vive provide full 6-DoF (*i.e.* rotation and translation) tracking at ∼1 kHz frequencies. This full 6-DoF tracking comes at a cost by requiring external lighthouses or IR cameras.

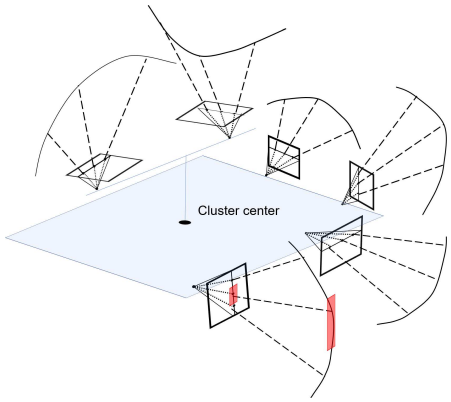


Figure 1: Due to radial distortion, rows of rolling shutter images correspond to rays which span a curve in space. Rolling shutter further provides a dense sampling of the scene in time. This combination of high frequency and curved sampling of the scene provides us more information than traditional pinhole cameras. The highlighted region reflects per-row image region corresponding to the curve.

Microsoft's HoloLens provides inside-out tracking without requiring any external apparatus, but still incurs tracking errors; see Sec. 4.3 for a more detailed analysis. It is well understood that high tracking rates are critical for immersive experiences in AR/VR systems [31]. We present a full 6-DoF tracking system using multiple cameras that can track at extremely high frequencies up to 86.4kHz using commodity rolling shutter cameras.

Many commercial AR/VR systems rely on camera-based tracking, but for cost reasons use standard cameras with frames rates of a few tens of frames per second: *e.g.*, the Oculus Rift uses a 60 Hz external IR camera. In order to achieve the higher frame rates required for tracking, they rely on gyroscope data to improve the tracking frame rate. However, positional tracking still remains difficult to achieve, as IMUs drift [23]. Even using high frame-rate global shutter cameras is not the solution, as the high frame rates required lead to a decrease in the maximum possi-

ble exposure time (equal to the inverse of the kHz frame rate), making capturing sufficient light impractical, especially indoors [17], where AR/VR systems are most frequently used. Also, most of these systems are head-worn and require small-form-factor cameras, further limiting the amount of light captured even.

We present a method for drastically improving the 6-DoF tracking frequency in AR/VR devices while maintaining comparable tracking accuracy w.r.t the current state of the art [5] and currently available commercial systems such as the Microsoft HoloLens. Our camera-based approach uses standard low-cost commodity cameras and leverages two unlikely camera '*features*': rolling shutter and radial distortion, both of which are usually considered to be nuisances in computer vision applications (Fig. 1). Bapat *et al*. [5] demonstrated a proof-of-concept inside-out tracking system using a cluster of 10 rolling shutter cameras that operated at frequencies as high as 80kHz. We extend their system and overcome the limitations of the large number of required cameras as well as the need for stereo rectification and radial undistortion. We achieve comparable tracking stability while (only requiring *four* cameras instead of ten) making our system significantly more practical for use in a headset. As a result, our system is simpler and more flexible in terms of camera placement in the multi-camera rig. Our novel insight is that radial distortion in the camera lens induces constraints on the tracking system, which in turn reduces the number of cameras required. We forgo compensating the image for radial undistortion and directly use the radially distorted rolling shutter image as captured by the camera yielding more efficient computation, again of significant importance for high-frequency tracking.

2. Related Work

The ubiquitous presence of rolling shutter (RS) cameras in almost every cell-phone and low-cost camera has driven much research to formulate its imaging model [15, 1, 2, 7]. In rolling shutter capture, each row (or column) is captured at a slightly different time. Researchers have long strived to remove rolling shutter and the artifacts resulting from camera motion and the different exposure start times of the various rows of the camera have generally been regarded as nuisances of the imaging process [13, 16, 30].

There have also been many efforts in the past decade to incorporate rolling shutter capture models into traditional algorithms in computer vision. Traditional geometric problems were explored while accounting for RS effects. For example, Albl *et al*. [3] studied the RS PnP problem and formulated RS P6P to estimate a moving camera's pose from six 2D-to-3D correspondences. Saurer *et al*. [33] developed a minimal absolute pose estimation solver based on Gröbner basis functions. Magerand *et al*. [26] used constrained global optimization to estimate uniform motion of a known object captured using a RS camera. Similar to [1, 2], they treated the RS camera as a highly sensitive motion sensor given the template of the known object in the scene and 2D-3D point correspondences. Dai *et al*. [9] studied epipolar constraints for RS cameras and showed that epipolar constraints do not define a line for RS cameras but rather define epipolar curves. RS in multi-view stereo was addressed by Saurer *et al*. [32], in which the authors adapted the plane sweeping stereo algorithm for RS and showed that the combination of RS artifacts and lens distortion leads to biased 3D reconstruction if global shutter is naïvely assumed. RS calibration was also examined in [28, 15], where the goal was to measure the line delay, *i.e*., the time delay between consecutive rows from imagery. Structure from Motion (SfM) traditionally assumes global shutter and was thus shown to be unreliable when used with images with large RS distortion [25, 19]. To resolve this, Hedborg *et al*. [18] proposed a RS aware Bundle Adjustment (BA) method to account for the RS effect. This effort improved the SfM pipeline, which had previously not worked well for RS images with significant motion [25]. Later, Saurer *et al*. [34] developed a full-fledged SfM pipeline in which they assumed constant intra-frame rotational and translational velocity similar to [3] during BA and enforced temporal smoothness across frames using GPS readings. Recently, Albl *et al*. [4] showed that RS cameras exhibit more degenerate configurations in SfM as compared to the traditional pinhole global shutter model. In such cases, BA prefers a trivial case where the scene is estimated to be planar.

Kim *et al*. [21] introduced a monocular Simultaneous Localization and Mapping (SLAM) system specifically for RS cameras by adapting LSD–SLAM [12]. They parametrized the intra-frame motion via a k-control point B-spline. In contrast to our method, they assumed no radial distortion and estimate single pose for the entire frame. More recently, they incorporated radial distortion in [22] by using generalized epipolar curves instead of epipolar lines. We introduce a tracking system (without mapping) which essentially turns the rolling shutter camera into a *computational sensor*. We explicitly estimate intra-frame motion while leveraging the radial distortion as a feature, and hence methods like [21, 20, 29] which parametrize the intra-frame motion using splines can benefit from our method.

3. Our Approach

In many computer vision applications, rolling shutter artifacts and radial distortion are considered negative aspects of the capture process that must be corrected for, after image formation. However, just as Bapat *et al*. [5] previously converted rolling shutter "distortion" into an enabling feature for high frequency tracking, we argue that radial distortion can serve to greatly reduce a tracking system's complexity.

(a) (b) (c) (d) (e)

Figure 2: Under radial distortion, an input row sample represents a curve in the undistorted image space. (a) Radially distorted image with one row highlighted by a dashed green line; (b) the same row in undistorted image space forms a curve. By approximating this curve with (for example) three linear line segments as shown in (c-e), we obtain multiple virtual cameras that each provide an independent linear constraint. In this sense, a single radially distorted line-image approximates the multiple pinhole cameras used in [5]. For purposes of visualization, the distortion in (a) and (b) has been exaggerated; in practice, we use a larger number of local linear approximations that better fit the undistorted curve.

It is in fact a feature, not an artifact.

We extend the high frequency 6-DoF tracking method introduced by Bapat *et al*. [5] that used a cluster of RS cameras arranged in stereo-pairs. In their system, each row of the RS camera is treated as a line-camera capturing the scene at high frequency. They process each row to measure pixel disparities for their stereo camera to obtain depth of a single point per camera. They also measure the pixel shift of this point across time to obtain a constraint on the cluster motion. Such a formulation provides a single constraint per stereo-pair in a system of linear equations, requiring at least 12 cameras. In their simulation experiments, they use 10 stereo-pairs that are exactly in-plane. They also enforce that the left-right rows expose at the same time. This is hard to achieve in reality at kHz capture rates, particularly as their 5 GoPro stereo-pairs are not genlocked. This leads to staggering of row exposures in time, which allows them to track points with different y coordinates reducing the number of cameras from 6 to 5 stereo-pairs. We use the spirit of this earlier work to exploit the different rows of a RS camera as independent high-frequency line cameras. Given this high-frequency sampling, we propose a novel system that leverages the naturally occurring radial distortion of the lens as a feature for motion estimation to obtain a complementary set of constraints, without the need of row-level in-step capture. Our tracking also benefits from a wider field of view where radial distortion is more pronounced. Such cameras are commonly used in tracking research, *e.g*., using a 130^o FoV camera is recommended for LSD-SLAM [11].

Ideal pinhole rolling shutter cameras only allow for a single linear constraint per row-image. Our key insight is that the process of undistorting a radially distorted row yields a curve in the image space that can be approximated by multiple line segments, as shown in Fig. 2. Linear shifts can thus be computed for each segment independently, affording us a linear constraint per segment, substantially reducing the number of cameras by 60%: from 10 cameras in Bapat *et al*. [5] to 4 cameras in our system. In tetherless AR/VR devices, reducing system complexity is critical, and our approach substantially lowers the video bandwidth and headset weight. Our proposed extension has the following advantages over the system of Bapat *et al*. [5] while maintaining its benefits of kHz tracking frequencies: 1.) Instead of 10 cameras, we show high-frequency tracking can be performed using just 4 cameras. They are arranged so that every camera has overlap with at least one other camera. 2.) Our proposed approach does not need guarantees of physically in-plane stereo camera sensors for depth estimation, as we propagate a depthmap for each camera across time. Rather, we benefit from the rotation between cameras, as each view provides additional scene coverage. 3.) We exploit the lens' radial distortion and use it to track multiple points per row; see Sec. 3.2 for a detailed description.

3.1. Preliminaries

We first introduce the notation used in this paper while explaining the tracking constraints of [5]. The time step Δt is defined from t_1 to t_2, where $t_2 > t_1$, and magnitude of $t_2 - t_1$ may vary. Bold capital letters define a matrix, *e.g*. $\mathbf{M}$. We define a relative transformation from space w to space c, as ${}^{c}\mathbf{T}_{w}$, which transforms points in space w to space c, and a transform within a space w to be $\mathbf{T}_{w}$.

The tracking method of [5] uses a cluster of cameras to track the head-pose starting from the identity pose. There are N cameras in the cluster, and the relative pose of camera $n \in N$ in the cluster space is denoted by ${}^{n}\mathbf{T}_{cl}$, which can be estimated by performing an extrinsic calibration.

The 6 DoF head-pose tracking is performed by measuring small pixel shifts between the projection of a 3D world point $X_w = [x_w\ y_w\ z_w\ 1]^T$ in camera n at time instants t_1 and t_2, and transforming the shifts into the 3D space using the depth of the point. These operations can be expressed

for a camera n as follows:

$$\begin{aligned} X(n,t_1) &= {}^{n}\mathbf{T}_{cl}\,{}^{cl}\mathbf{T}_{w}(t_1)X_w \\ X(n,t_2) &= {}^{n}\mathbf{T}_{cl}\,{}^{cl}\mathbf{T}_{w}(t_2)X_w \end{aligned} \tag{1}$$

where $X(n,t_1) = [x_1\ y_1\ z_1\ 1]^T$ and $X(n,t_2) = [x_2\ y_2\ z_2\ 1]^T$ correspond to the same 3D world point X_w, and are expressed relative to camera n at time t_1 and t_2 respectively. The transformation ${}^{cl}\mathbf{T}_w$ transforms the 3D point X_w from world space to the cluster space.

The 3D points $X(n,t_1)$ and $X(n,t_2)$ are related to the corresponding pixels $\boldsymbol{x}(t_1)$ and $\boldsymbol{x}(t_2)$, $\boldsymbol{x}(t_1) = [p_x(t_1)\ p_y(t_1)\ 1]^T$, via the intrinsic matrix K_n. Depths $z(t_1)$, $z(t_2)$ as defined by the following relations:

$$\begin{aligned} X(n,t_1) &= z(t_1)\ \boldsymbol{K}_n^{-1}\boldsymbol{x}(t_1) \\ X(n,t_2) &= z(t_2)\ \boldsymbol{K}_n^{-1}\boldsymbol{x}(t_2) \end{aligned} \tag{2}$$

Tracking at high frequencies allows them to assume small relative motion at each time step. We denote the small motion approximation for the combined transform ${}^{cl}\mathbf{T}_w(t_2){}^{cl}\mathbf{T}_w^{-1}(t_1)$ by $\delta\boldsymbol{M}_{cl}$. The cluster's approximate small motion $\delta\boldsymbol{M}_{cl}$ can be expressed in matrix form as

$$\delta\mathbf{M}_{cl} = \begin{bmatrix} 1 & \theta_z & -\theta_y & -t_x \\ -\theta_z & 1 & \theta_x & -t_y \\ \theta_y & -\theta_x & 1 & -t_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{3}$$

Using the small motion assumption, Eq.(1) can be simplified to

$$X(n,t_2) = {}^{n}\mathbf{T}_{cl}\ \delta\ \mathbf{M}_{cl}\ {}^{cl}\mathbf{T}_{n}\ X(n,t_1) \tag{4}$$

The depth $z(t_1)$ is estimated using the rows captured at time t_1 from the left and right cameras in the stereo-pair by measuring pixel disparity. To estimate motion, temporal stereo is used to measure the shift in pixels $p_x(t_2) - p_x(t_1)$ across the time step Δt.

Using Eqn.(2), authors of [5] substitute image points for 3D points because the measurements are in terms of pixels. Subsequently, a linear system is formulated by rearranging Eq.(4) to the form $\mathbf{C}\ \mathbf{A}\ Y = \mathbf{C}\ B$, where $\mathbf{C}$ is a diagonal matrix of weighting factors estimated according to the confidence in the measured shifts and $Y = [\theta_x\ \theta_y\ \theta_z\ t_x\ t_y\ t_z]^T$ represents the unknown 6-DoF motion vector of the camera. Matrix $\mathbf{A}$ captures the cluster configuration and the tracked 3D points, while vector $\mathbf{B}$ captures the shift measurements.

However, the proof-of-concept high frequency tracking method from Bapat *et al*. [5] assumes ideal conditions regarding row-level in-step exposure and exactly in-plane stereo-pairs. Also they assume a perfect lens, *i.e*. they treat radial distortion as an artifact and "correct" for it during preprocessing in their experiments. Due to this, they are forced to perform rectification and undistortion of the images to make the input images more amenable to their model. This pre-processing scheme however blends neighboring rows captured at different times during the radial undistortion and rectification steps and reduces image size. In contrast, our proposed method leverages the full breadth of the original image information to obtain tracking with fewer cameras.

3.2. Radial Distortion

Significant radial distortion is observed in wide angle cameras and activity cameras such as GoPros. Such wide-angular cameras are preferred for more stable tracking. Radial distortion transforms image rows to curves, and hence rays corresponding to each row span a curved surface in 3D space, rather than a plane as assumed in the pinhole model (Fig. 1). We model radial distortion in our system and explicitly leverage the curved mapping introduced by it to measure points at different rows in the undistorted image space. We use the first two parameters, k_1 and k_2, of Brown's radial distortion model [8]. These parameters transform an undistorted normalized pixel $(\widetilde{x}_u,\ \widetilde{y}_u)$ into a distorted normalized pixel $(\widetilde{x}_d,\ \widetilde{y}_d)$ as

$$\begin{aligned} \widetilde{x}_d &= \widetilde{x}_u\ \left(1 + k_1 r^2 + k_2 r^4\right) \\ \widetilde{y}_d &= \widetilde{y}_u\ \left(1 + k_1 r^2 + k_2 r^4\right) \\ r^2 &= \widetilde{x}_u^2 + \widetilde{y}_d^2. \end{aligned} \tag{5}$$

When such a distorted row is captured by a rolling shutter camera, the mapping from the 3D world point to the image is dependent on the time at which a row is exposed, as well as the distortion transformation.

In the rest of the paper, we omit the intrinsic camera parameters for brevity and assume normalized coordinates for the image points. We leverage radial distortion to relax two restrictions: 1) radial distortion bends the row into a curve in the image plane, which means that shifts measured at different $\widetilde{x}_d$ positions at the same row $\widetilde{y}_d$ have different $\widetilde{y}_u$ locations. This has a similar effect as the time staggering observed in real cameras [5]. 2) We track multiple points per row, as opposed to a single point, providing more constraints, enabling our method to track using fewer cameras.

Rolling shutter provides row-wise capture, and our method measures how the current row shifts across time in the horizontal direction. In a given row, we can express the 3D X-coordinate in terms of $\widetilde{x}_u$ pixel positions at two time instants t_1 and t_2 using the pixel shift s_t as follows:

$$x(t_2) = \widetilde{x}_u(t_2)z(t_2) \approx (\widetilde{x}_u(t_1)+s_t)z(t_1) = x(t_1)+s_t z(t_1). \tag{6}$$

3.2.1 Linear independent constraints

We now show the linear independence of the constraints obtained for different points on the row whose rays span a 3D

curve. Consider Eqn.(4), with ${}^{n}\mathbf{T}_{cl}$ as identity for simplicity (corresponding to the anchor camera in the cluster). The horizontal shift s_t corresponds to the first row of this vector equation. Combining Eqn.(4) with Eqn.(6), we obtain

$$\begin{aligned} x(t_2) &= x(t_1) - \theta_z \widetilde{y}_u z(t_1) + \theta_y z(t_1) + t_x \\ &\Rightarrow s_t = -\theta_z \widetilde{y}_u + \theta_y + \frac{t_x}{z(t_1)}. \end{aligned} \tag{7}$$

In Eqn.(7), for a given fixed $\widetilde{y}_u$ and small motion $\delta\mathbf{M}_w$, the only way the observed pixel shift s_t can change is due to the depth of the point $z(t_1)$, which is scene-dependent. That is why Bapat *et al*. [5] had to rely on more cameras to obtain sufficiently many constraints. In contrast, our method overcomes this inherent limitation by recognizing that, due to the radial distortion, each distorted row spans a curve in undistorted space. Hence, we can extract an independent constraint from a piece-wise local linear segment in undistorted space as depicted in Fig. 2 providing us multiple independent constraints per radially distorted row. The additional cameras in the cluster provide even more constraints. This is imperative for reducing the required number of cameras in our system, as each camera in the cluster provides us with multiple constraints at each point in time instead of just one. Similar analysis can be done for ${}^{n}\mathbf{T}_{cl}$ not equal to identity, which is the case for the rest of the cameras in the cluster (see our supplementary material for details). The linear independence of the constraints still persists as the choice of coordinate system to express poses ${}^{n}\mathbf{T}_{cl}$ for the cameras is arbitrary and the constraints are obtained from a single row of each camera.

The stability of such an over-determined linear system can be examined using the condition number [6]. We found that the condition number of our linear system using a 4- or 6-camera cluster is of the same order as that of the 10-camera system of [5]. Note that the number of constraints depends upon the extent of radial distortion present in the image. Higher radial distortion would give more varying $\widetilde{y}_u$ locations for the sampled $\widetilde{y}_d$ locations, making the system more stable. We assume that significant radial distortion exists due to the lens in all of our experiments. Without radial distortion, our linear system reduces to the case of Bapat *et al*. [5]. Small distortion, in the range of one pixel, is also not sufficient, as the constraints obtained from a single row are similar to each other forming an ill-conditioned system.

3.2.2 Homography-based warp

We directly process the radially distorted rolling shutter rows as soon as they become available. To achieve this, for each incoming distorted row, we synthesize a reconstructed row from the older frames for measuring shifts. This reconstructed row is created by sampling from a global shutter undistorted image frame F_{GS} with absolute pose T_{ref}. This frame is synthesized from previous row images. To create F_{GS}, we split the rotational homography warp of $H = KR_{ref}R_{row}^{-1}K^{-1}$, which maps each rolling shutter row to the reference pose by creating two lookup tables: 1) a lookup table to undistort previous frames and adjust their rotation by application of $R_{row}^{-1}K^{-1}$, and 2) a similar table for KR_{ref}. This helps us to avoid redundant computation when the reference pose T_{ref} changes. The reference pose is chosen adaptively such that the motion between the reference and the current row is within a predefined threshold.

3.2.3 Robust shift estimation

As we directly compare distorted rows, small errors in the pixel shift measurements can yield a result with significant errors depending upon their corresponding depth and position in the distorted space. To mitigate this, we use a robust double exponential Holt-Winters smoothing filter to remove outlier shift estimates [14]. For a time series m_t, this filter with trend B_t is defined as follows:

$$\begin{aligned} \tilde{m}_t &= \alpha m_t^* + (1-\alpha)(\tilde{m}_{t-1} + B_{t-1}) \\ \tilde{B}_t &= \beta(\tilde{m}_t - \tilde{m}_{t-1}) + (1-\beta)B_{t-1}, \end{aligned} \tag{8}$$

where α and β are filter parameters and "tilde" denotes the filtered data. m_t^* is a "cleaned" version of m_t using Huber loss Ψ to penalize large differences between noisy estimates m_t and one-step forecasts $m_{t|t-1}$. The cleaned version m_t^* is

$$m_t^* = \Psi\left(\frac{m_t - m_{t|t-1}}{\hat{\sigma}_t}\right)\hat{\sigma}_t + m_{t|t-1}. \tag{9}$$

The scale of this difference is estimated by a slowly varying τ^2-scale estimate $\hat{\sigma}_t$, which is highly robust to outliers. We refer the reader to Gelper *et al*. [14] for more details.

3.3. Cluster configuration

The placement of the cameras in the cluster affects the tracking estimates according to Eqn.(4). We use two cluster configurations in our experiments: 1) a 4-camera configuration, and 2) a 6-camera configuration. The even-numbered cameras are rotated by 90° in the image plane and have translations and small rotations in all three directions w.r.t the odd-numbered cameras. See Fig. 1 for a visualization of our 6-camera cluster. Although using a 2-camera cluster is possible as the problem is well-posed, in our current cluster, the first two cameras have principal axes mostly along the Z direction (forward) creating an *ill-conditioned* linear system. Thus measurements constraining the Z direction are sensitive to noise and the accuracy is unsatisfactory.

The overlap between adjacent cameras can be decreased to extend the overall field of view of the tracking system. We estimate a depthmap per camera, and use the large overlap primarily to bootstrap the depth estimation in the experiments. A possible reduction in overlap can be achieved if

bootstrapping is done using stereo over multiple baselines across time as in LSD-SLAM [12].

4. Experiments

We present experiments using synthetic and real world imagery. For the synthetic experiments, we created a simulator using OpenGL and Qt, which is capable of simulating rolling shutter capture under radial distortion. For the real data experiment, we used tracking information from a 2000Hz Hi-Ball tracking system [36] as ground truth. This system is a low-latency wide-area tracking system operating with absolute errors of less than 0.03° and 0.5mm.

The metric we use for evaluating our approach is the rendering pixel error, that is, the error in the rendering of the virtual scene caused by tracking errors as seen in a VR headset, as proposed by Bapat *et al*. [5]. This is crucial for AR/VR, as the rendering pixel errors directly affects user experience. For this metric, we estimate the rendering pixel error for a point 1m in front of the user for a resolution of 1080×1200 per eye, which is widely used by standalone headgears like the HTC Vive, Oculus Rift and Razer OSVR HDK2. We also present RMS errors of our tracking with respect to the ground truth provided by the Hi-Ball. We compare our results with Bapat *et al*. [5] using synthetic imagery, as it is easy to capture images of the same room along the same motion track using a simulator. We present the tracking results of our approach and compare against the rolling-shutter-aware SLAM method of Kim *et al*. [21] by capturing real imagery using our custom cluster. Additionally, we compare the performance of Microsoft's HoloLens with the Hi-Ball to provide a perspective on how our tracking errors compare against tracking with current commercial AR systems, which work at much lower tracking frequencies. Additional analysis is provided in the supplementary.

4.1. Synthetic data

We now present experiments using synthetic data, which was captured with camera parameters designed to be similar to real cameras. To compare directly with Bapat *et al*. [5], all of the cluster cameras have the same vertical FoV of 60° as in [5] and the images were rendered at 120Hz frame rate with a resolution of 480×640, making the tracking frequency 57.6kHz. As our approach needs radial distortion, we used distortion parameters $k_1 = -0.27$ and $k_2 = 0.11$ to reflect real cameras. We used real human motion data captured using the Hi-Ball to render this synthetic imagery. As the Hi-Ball tracking frequency does not match our tracking frequency, we interpolated it to match our estimation frequency of 57.6kHz. The room we used to simulate motion was captured using a handheld Kinect [10].

The 6-camera case is accurate for real motions, (see Table 1), while the 4-camera case shows more errors in tracking estimates, as it is inherently more sensitive to noise in pixel shifts. The 20-camera system of [5] estimates pose from a single point in a row, and hence is not robust. On the other hand, we leverage multiple points in each row for pose estimation and employ robust filters, leading to 2.55px RMS error for 4-camera case.Fig. 4 shows the rendering pixel error incurred across time for our 4- and 6-camera systems, and also the 20 camera cluster of [5]. The rendering pixel errors for the 20-camera system deviate and start to accumulate quickly, while our system shows gradual drift. See our supplementary material for motion plots depicting pose estimates across time.

4.2. Real data

For our experiments on real data, created a rig of six RS cameras mounted on top of a hardhat. All the cameras were GoPro Hero3+ silver edition and were kept at a narrow FoV, capturing at 720×1280 resolution at 120Hz. The line delay for each camera was calibrated using the method introduced by [28] using a checkerboard pattern. The overlap in the cameras helped us to calibrate adjacent cameras in pairs using Zhang's method [37]. All cameras were synchronized in time using a blinking LED.

Now, we present experiments using captured data at 120Hz, amounting to 86.4kHz tracking frequency. We can track orders of magnitude faster than the current commercial trackers for a fraction of the cost. Our method supports high-frequency tracking using the 4-camera and 6-camera cluster configurations. With the 6-camera setup, we need less smoothing, as the linear system has more redundant constraints; see Fig. 5, as compared to the 4-camera setup, see Fig. 6. Fig. 7 shows the rendering pixel error of our 4- and 6-camera setup against the RS aware SLAM method of Kim *et al*. [21]. As Kim *et al*. [21] estimate a pose per keyframe, we show the pixel error centered at each frame. The rendering pixel errors incurred for our approach are generally around 1 pixel and increase to 3 pixels as the tracking drifts with some outlier spikes in errors. On the other hand, the RS-aware SLAM system shows stronger deviation up to 5 pixels.

Compared to [5], we were able to track for much longer, estimating poses for more than twice the row-samples while still maintaining accuracy. If we compare similar track length tracking estimates as in [5], we incurred only 1 pixel error for real imagery. Table 1 shows the RMS error for translation, rotation, and rendering pixel error. For real imagery, the 4-camera case and 6-camera cases perform equally well, incurring 0.68px and 0.75px rendering error, respectively.

4.3. Comparison with HoloLens

We evaluate how our tracking errors compare to traditional systems like Microsoft's HoloLens for small-scale

E_{RMS}	T_x (cm)	T_y (cm)	T_z (cm)	θ_x (degree)	θ_y (degree)	θ_z (degree)	Rendering error (px)
4-camera real data	0.0648	0.0974	0.1064	0.0541	0.0776	0.0720	0.6804
6-camera real data	0.1783	0.0890	0.1139	0.0326	0.0499	0.0959	0.7507
HoloLens Fig.(8)	0.21	0.04	0.05	0.05	0.03	0.02	0.8249
4-camera synthetic data	0.43	0.18	0.23	0.14	0.23	0.06	2.55
6-camera synthetic data	0.38	0.27	0.22	0.13	0.22	0.08	2.39
20 camera system from [5]	0.67	0.83	1.34	0.21	0.47	0.29	4.63

Table 1: RMS errors for tracking estimates using real imagery for our 4-camera rig, 6-camera rig, and HoloLens pose estimates compared against the ground-truth Hi-Ball tracking. For comparison with [5], we show RMS error computed for the same synthetic scene for our 4 and 6-camera cases, and the 20 camera (10 stereo-pairs) case from [5]. The system of [5] incurs more errors, as it relies on only a single point per row for pose estimation.

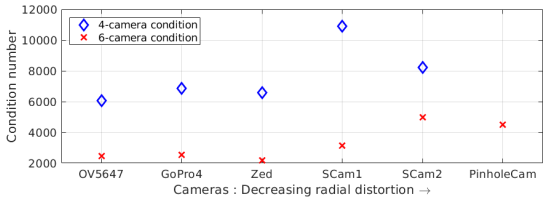


Figure 3: With higher radial distortion, the system stability increases (smaller condition number). Additional views increase stability (4- *vs*. 6-cameras). The plot shows the worst-case condition number (PinholeCam 4-camera result too large to show).

motion. It is impractical to wear the HoloLens, Hi-Ball and our camera cluster simultaneously, but we can attach the Hi-Ball to either HoloLens or our cluster. Hence, we compare the HoloLens with the Hi-Ball and our cluster with the Hi-Ball to provide an indirect comparison. Using the front color camera of the HoloLens, we perform hand-eye calibration using Tsai and Lenz's method [35] to register the HoloLens tracking data with the Hi-Ball. Fig. 8 shows the HoloLens tracking against the Hi-Ball tracking and demonstrates that the HoloLens suffers from translational drift but remains accurate in rotation. The motion range in this plot is similar to the scale of motion in our experiment with real imagery. The HoloLens is particularly good at tracking large translations, and at correcting for drift and incorrect translation estimates, due to its SLAM system. Fig. 8 highlights that the dominant motion in the Y-direction is tracked well by the HoloLens, but the rest of the motion axes have significant errors. Table 1 shows the RMS errors for the HoloLens tracking. HoloLens incurs 0.82 rendering pixel RMS error; on the other hand, for a similar scale of motion, our tracking system incurs 0.68px for our 4-camera cluster. Our approach thus achieves high accuracy in tracking 6 DoF poses, just using commodity rolling shutter cameras.

4.4. System conditioning and extent of distortion

System conditioning depends upon the amount of radial distortion present in the image. We rendered synthetic images for real, widely used cameras like the Pi OV5647 (k_1 = -0.31, k_2 = 0.083), GoPro 4 (-0.27, 0.11), and Zed (-0.17, 0.023). We also rendered images for simulated cameras: SCam1 (0.15, 0.05), SCam2 (0.05, 0.05), and PinholeCam (0, 0). For each of these camera types, we generated synthetic images for our 4- and 6-camera configurations according to the motion patterns in supplementary Sec. 3. We computed the worst-case condition number across all time points and compared this across cameras (Fig. 3). The condition number decreases with higher distortion due to the availability of more constraints per row, and it also decreases when more cameras are present. Average condition numbers (not shown) are substantially lower.

5. Conclusion

We have introduced a simple and flexible high frequency head-pose tracking system that explicitly leverages rolling shutter exposure and radial distortion without using any external markers. Our method achieves tracking frame rates that are orders of magnitude larger than current commercial systems like NDI's Optotrak Certus [27] for a fraction of the cost. In order to achieve this, we process every row of a rolling shutter image to estimate a pose per-row. This high-frequency sampling of rows enables us to assume linear motion. By splitting a radially distorted rolling shutter row into multiple locally linear line segments, we extract multiple linear constraints on 6-DoF head motion and accordingly require only requiring 4-6 cameras for stable 86.4 kHz tracking. Through rigorous experiments, we show the viability of our approach using synthetic data and validate using real imagery captured using our prototype camera cluster headgear. Moreover, our approach allows more flexibility in designing the cluster and simplifies the physical alignment constraints on the system. Most importantly, we have introduced the first approach to leverage both rolling shutter and radial distortion to improve high-frequency tracking.

Acknowledgments This research was partially supported by NSF grant No. CNS-1405847.

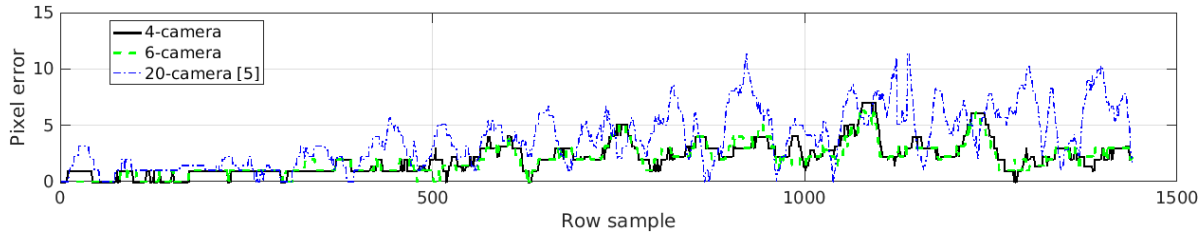
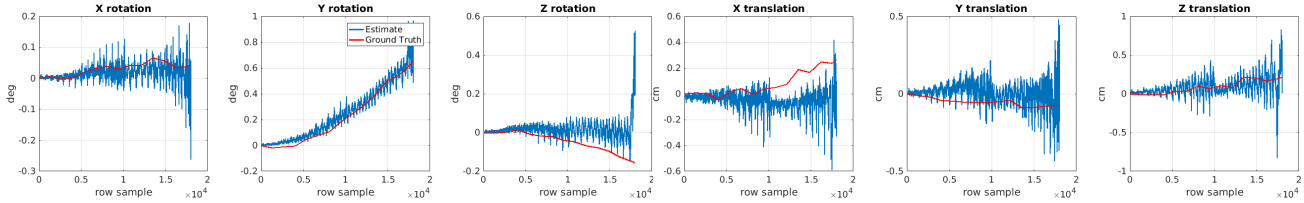
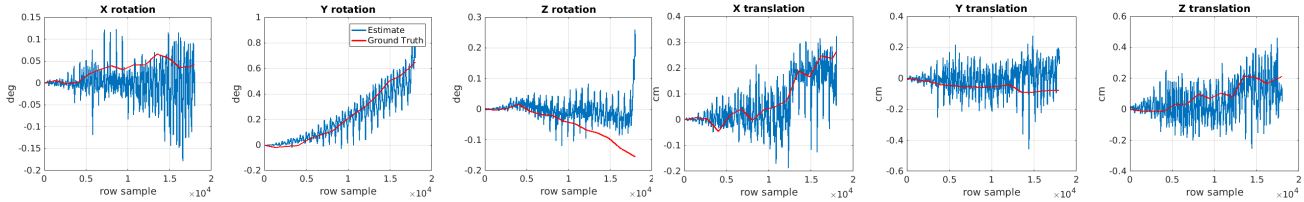


Figure 4: For the same camera motion, our tracking incurs far smaller rendering pixel errors using only 4 cameras as opposed to the 20-camera system of [5] for synthetic data.



(a) (b)

Figure 5: Tracking estimates of the 6-camera configuration using real imagery and Hi-Ball tracking data for ground truth: (a) Rotation estimates in degrees and (b) translation estimates in cm. Note that the scale of the y-axis is different for each figure.



(a) (b)

Figure 6: Tracking estimates of the 4-camera configuration using real imagery and Hi-Ball tracking data for ground truth: (a) Rotation estimates in degrees and (b) translation estimates in cm. Note that the scale of the y-axis is different for each figure.

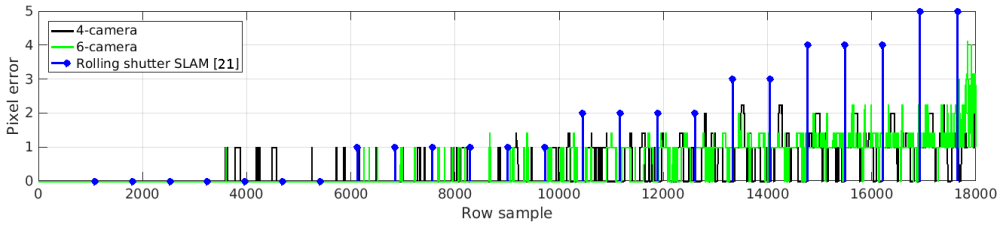
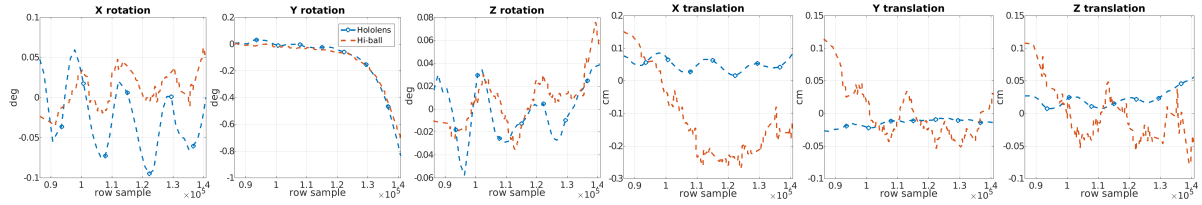


Figure 7: Render pixel estimates: Our 4-camera configuration (black) has higher error than our 6-camera configuration (green). Blue dots show the rendering error obtained by the method of Kim *et al*. [21], which maintains a pose estimate per keyframe, rather than per row.



(a) (b)

Figure 8: Tracking estimates of the HoloLens compared with the Hi-Ball. Note that the vertical axes are not the same across plots. The horizontal axis is expressed in terms of row-samples, instead of time, for easier comparison.

References

[1] O. Ait-Aider, N. Andreff, J.-M. Lavest, and P. Martinet. Exploiting rolling shutter distortions for simultaneous object pose and velocity computation using a single view. In *Computer Vision Systems, 2006 ICVS'06. IEEE International Conference on*, pages 35–35. IEEE, 2006.

[2] O. Ait-Aider, A. Bartoli, and N. Andreff. Kinematics from lines in a single rolling shutter image. In *Computer Vision and Pattern Recognition, 2007. CVPR'07. IEEE Conference on*, pages 1–6. IEEE, 2007.

[3] C. Albl, Z. Kukelova, and T. Pajdla. R6p-rolling shutter absolute camera pose. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2292–2300, 2015.

[4] C. Albl, A. Sugimoto, and T. Pajdla. Degeneracies in rolling shutter sfm. In *European Conference on Computer Vision*, pages 36–51. Springer, 2016.

[5] A. Bapat, E. Dunn, and J.-M. Frahm. Towards kilo-hertz 6-dof visual tracking using an egocentric cluster of rolling shutter cameras. *IEEE Transactions on Visualization and Computer Graphics*, 22(11):2358–2367, 2016.

[6] D. A. Belsley, E. Kuh, and R. E. Welsch. *Regression diagnostics: Identifying influential data and sources of collinearity*, volume 571. John Wiley & Sons, 2005.

[7] D. Bradley, B. Atcheson, I. Ihrke, and W. Heidrich. Synchronization and rolling shutter compensation for consumer video camera arrays. In *Computer Vision and Pattern Recognition Workshops, 2009. CVPR Workshops 2009. IEEE Computer Society Conference on*, pages 1–8. IEEE, 2009.

[8] D. C. Brown. Decentering distortion of lenses. *Photometric Engineering*, 32(3):444–462, 1966.

[9] Y. Dai, H. Li, and L. Kneip. Rolling shutter camera relative pose: Generalized epipolar geometry. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016.

[10] M. Dou, L. Guan, J.-M. Frahm, and H. Fuchs. Exploring high-level plane primitives for indoor 3d reconstruction with a hand-held rgb-d camera. In *Asian Conference on Computer Vision*, pages 94–108. Springer, 2012.

[11] J. Engel and Schöps. LSD-SLAM, General Notes on Good Results. `https://github.com/tum-vision/lsd_slam#316-general-notes-for-good-results`, 2014. [Online; accessed 12-November-2017].

[12] J. Engel, T. Schöps, and D. Cremers. Lsd-slam: Large-scale direct monocular slam. In *European Conference on Computer Vision*, pages 834–849. Springer, 2014.

[13] P.-E. Forssén and E. Ringaby. Rectifying rolling shutter video from hand-held devices. In *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*, pages 507–514. IEEE, 2010.

[14] S. Gelper, R. Fried, and C. Croux. Robust forecasting with exponential and holt–winters smoothing. *Journal of forecasting*, 29(3):285–300, 2010.

[15] C. Geyer, M. Meingast, and S. Sastry. Geometric models of rolling-shutter cameras.

[16] M. Grundmann, V. Kwatra, D. Castro, and I. Essa. Effective calibration free rolling shutter removal. 2012.

[17] A. Handa, R. A. Newcombe, A. Angeli, and A. J. Davison. Real-time camera tracking: When is high frame-rate best? In *European Conference on Computer Vision*, pages 222–235. Springer, 2012.

[18] J. Hedborg, P.-E. Forssén, M. Felsberg, and E. Ringaby. Rolling shutter bundle adjustment. In *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*, pages 1434–1441. IEEE, 2012.

[19] J. Hedborg, E. Ringaby, P.-E. Forssén, and M. Felsberg. Structure and motion estimation from rolling shutter video. In *Computer Vision Workshops (ICCV Workshops), 2011 IEEE International Conference on*, pages 17–23. IEEE, 2011.

[20] C. Kerl, J. Stuckler, and D. Cremers. Dense continuous-time tracking and mapping with rolling shutter rgb-d cameras. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2264–2272, 2015.

[21] J.-H. Kim, C. Cadena, and I. Reid. Direct semi-dense slam for rolling shutter cameras. 2016.

[22] J.-H. Kim, Y. Latif, and I. Reid. Rrd-slam: Radial-distorted rolling-shutter direct slam. In *Robotics and Automation (ICRA), 2017 IEEE International Conference on*, pages 5148–5154. IEEE, 2017.

[23] S. M. LaValle, A. Yershova, M. Katsev, and M. Antonov. Head tracking for the oculus rift. In *Robotics and Automation (ICRA), 2014 IEEE International Conference on*, pages 187–194. IEEE, 2014.

[24] P. Lincoln, A. Blate, M. Singh, T. Whitted, A. State, A. Lastra, and H. Fuchs. From motion to photons in 80 microseconds: Towards minimal latency for virtual and augmented reality. *IEEE Transactions on Visualization and Computer Graphics*, 22(4):1367–1376, 2016.

[25] F. Liu, M. Gleicher, J. Wang, H. Jin, and A. Agarwala. Subspace video stabilization. *ACM Transactions on Graphics (TOG)*, 30(1):4, 2011.

[26] L. Magerand, A. Bartoli, O. Ait-Aider, and D. Pizarro. Global optimization of object pose and motion from a single rolling shutter image with automatic 2d-3d matching. In *European Conference on Computer Vision*, pages 456–469. Springer, 2012.

[27] NDI. Optotrak Certus Technical Specifications. `https://www.ndigital.com/msci/products/optotrak-certus/#optotrak-certus-technical-specifications`. [Online; accessed 28-March-2018].

[28] L. Oth, P. Furgale, L. Kneip, and R. Siegwart. Rolling shutter camera calibration. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1360–1367, 2013.

[29] A. Patron-Perez, S. Lovegrove, and G. Sibley. A spline-based trajectory representation for sensor fusion and rolling shutter cameras. *International Journal of Computer Vision*, 113(3):208–219, 2015.

[30] V. Rengarajan, A. N. Rajagopalan, and R. Aravind. From bows to arrows: Rolling shutter rectification of urban scenes. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2773–2781, 2016.

[31] M. V. Sanchez-Vives and M. Slater. From presence towards consciousness. In *8th Annual Conference for the Scientific Study of Consciousness*, 2004.

[32] O. Saurer, K. Koser, J.-Y. Bouguet, and M. Pollefeys. Rolling shutter stereo. In *Proceedings of the IEEE Intl. Conference on Computer Vision*, pages 465–472, 2013.

[33] O. Saurer, M. Pollefeys, and G. H. Lee. A minimal solution to the rolling shutter pose estimation problem. In *Intelligent Robots and Systems (IROS), 2015 IEEE/RSJ International Conference on*, pages 1328–1334. IEEE, 2015.

[34] O. Saurer, M. Pollefeys, and G. H. Lee. Sparse to dense 3d reconstruction from rolling shutter images. In *IEEE Computer Vision and Pattern Recognition*. IEEE, 2016.

[35] R. Y. Tsai and R. K. Lenz. A new technique for fully autonomous and efficient 3d robotics hand/eye calibration. *IEEE Transactions on robotics and automation*, 5(3):345–358, 1989.

[36] G. Welch, G. Bishop, L. Vicci, S. Brumback, K. Keller, et al. The hiball tracker: High-performance wide-area tracking for virtual and augmented environments. In *Proceedings of the ACM symposium on Virtual reality software and technology*, pages 1–ff. ACM, 1999.

[37] Z. Zhang. A flexible new technique for camera calibration. *IEEE Transactions on pattern analysis and machine intelligence*, 22(11):1330–1334, 2000.

[38] F. Zheng, T. Whitted, A. Lastra, P. Lincoln, A. State, A. Maimone, and H. Fuchs. Minimizing latency for augmented reality displays: Frames considered harmful. In *International Symposium on Mixed and Augmented Reality (ISMAR)*, pages 195–200, 2014.