Estimating Velocity and Spin of Spherical Objects from Rolling-Shutter Image(s)

Wenjie Xue[1], Jun Yang[1], Jingmin Wang[1,2], and Limin Shang[1]

[1] Epson Canada Ltd, Toronto, ON L3R 6G3, Canada
{Mark.Xue,Jun.Yang,Limin.Shang}@ea.epson.com
[2] University of Toronto, Toronto, ON M5S 1A1, Canada
jingmin.wang@mail.utoronto.ca

Abstract. Rolling-shutter cameras introduce characteristic distortions when imaging fast moving objects, and these effects are typically treated as artifacts to be corrected. In this work, we instead leverage rolling-shutter distortions as a valuable source of temporal information to estimate the 3D translational and angular velocities of rapidly moving spherical objects from a single rolling-shutter frame. We design a robust and easily detectable spherical pattern and propose a correspondence-free formulation that recovers motion by enforcing geometric consistency in a back-projection framework. By exploiting the geometry of the sphere, translational and rotational motions are decoupled and estimated through a two-stage optimization process, enabling reliable velocity recovery even for textureless objects. Extensive experiments on both synthetic and real datasets demonstrate accurate and robust estimation of motion parameters under challenging high-speed conditions.

Keywords: Rolling Shutter · Single-Image Motion Estimation · Spherical Motion Reconstruction

1 Introduction

Estimating the 3D translational velocity and spin of a fast-moving ball is central to sports and industrial applications. Currently, reliable motion capture typically relies on specialized high-speed global-shutter cameras, event sensors, or radar systems. By contrast, most commodity cameras use CMOS rolling-shutter sensors, whose sequential scanline exposure produces the familiar "wobble" artifacts under rapid motion (see Fig. 1). Existing rolling-shutter motion estimation methods, however, are a poor fit for spherical objects. Many approaches require explicit 3D–2D correspondences [1,17], which are typically difficult to acquire in real-world scenarios. Other methods rely on straight-line features with additional parameters and assumptions [2], limiting their applicability to spheres, which often lack such straight-line structures. Moreover, these formulations jointly optimize a large number of coupled variables, leading to highly non-convex problems that are sensitive to initialization and prone to instability.

In this work, we present a formulation that eliminates the need for explicit 2D–3D correspondences and leverages rolling-shutter distortions as motion cues

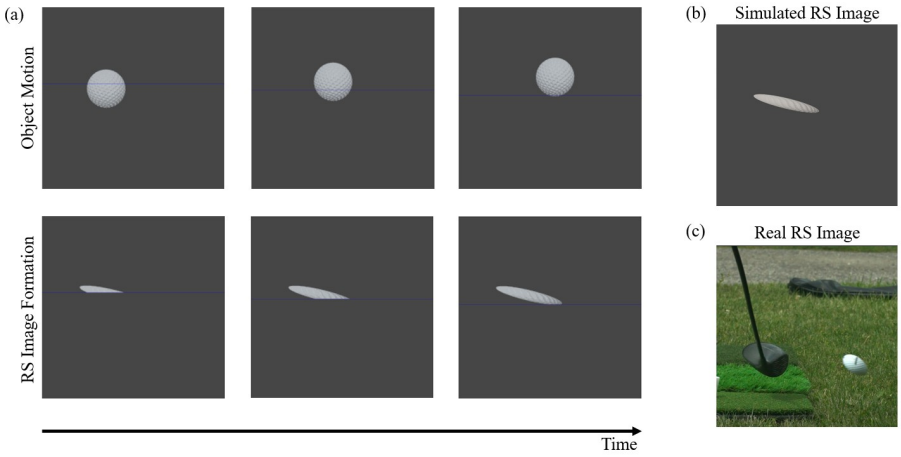


Fig. 1: (a) Visualization of rolling-shutter distortion for a sphere moving rightward. Top: Scene motion during image acquisition. Bottom: Corresponding rolling-shutter image formation. The blue line indicates the active scanline. (b) Rolling-shutter image rendered by our simulator. (c) Rolling-shutter image captured in the real world.

for spherical objects. Our key idea is to back-project observed image features into 3D rays, transform them to a common temporal reference frame, and recover object motion through geometric consistency constraints. To achieve this, we first design a robust, easily detectable pattern for the sphere and formulate 6D sphere motion estimation as a two-stage optimization problem. In the first stage, we estimate the 3D translational motion by exploiting the sphere's geometric properties, making the method applicable to textureless objects. In the second stage, we leverage the designed pattern to estimate the sphere's 3D rotational motion through a separate optimization. The resulting optimization is low-dimensional, computationally efficient, and stable in practice, while naturally extending to multi-image and multi-camera settings. Together, these design choices yield a robust and efficient solution for spherical motion recovery.

In summary, we make the following key contributions:

1. A back-projection-based algorithm that estimates 3D translational and angular velocity from a single rolling-shutter image without requiring explicit 3D–2D correspondences.
2. A geometry-driven two-stage optimization framework that decouples translational and angular motion, yielding stable convergence and reliable translation estimates for textureless spheres.
3. Extensions to multi-image and multi-camera configurations, validated in both simulation and a real-world golf launch-monitor prototype.

2 Related Work

2.1 Rolling-Shutter Cameras

Rolling-shutter cameras capture image rows sequentially, producing geometric distortions in dynamic scenes [19]. Most prior work treats these distortions as artifacts to be removed [5,12]. Other approaches explicitly model rolling-shutter geometry within pose estimation and bundle adjustment frameworks [3,4,11,13, 15,22]. More recently, learning-based methods have been proposed to estimate camera motion and synthesize undistorted images from one or more rolling-shutter frames [16,21].

Beyond correction, a complementary line of research exploits rolling-shutter distortions as motion cues. Ait-Aider et al. [1] showed that pose and velocity can be recovered from a single RS image with 2D–3D correspondences. Later works extended this idea to more complex motion models, including piecewise global-shutter approximations for non-uniform motion [18], Taylor expansions for small angular motion [17], and geometric primitives such as line correspondences for kinematic estimation [2].

Despite these advances, the above methods rely on accurate feature correspondences, which are difficult to obtain for fast-moving spherical objects that are often textureless or exhibit repetitive patterns. Moreover, in industrial and sports settings, rapid object motion violates the small-motion and global-shutter assumptions of prior methods, degrading feature detection and matching. As a result, no dedicated formulation robustly recovers spherical object motion directly from rolling-shutter imagery.

2.2 Motion Estimation for Spherical Objects

Estimating the velocity of a moving sphere has been widely studied in computer vision and robotics, typically using multi-frame or multi-camera setups with high-speed global-shutter cameras to avoid rolling-shutter distortions [8,24, 25]. In parallel, event-camera-based approaches have emerged as an alternative that bypasses these limitations, enabling accurate recovery of spherical motion through spatiotemporal optimization of asynchronous events [20]. More recently, learning-based methods have leveraged video sequences and known scene geometry to infer 3D trajectory and spin directly from data [14]. Visual servoing–based approaches [9] have also been explored in robotic settings, where image feedback is used to estimate and track sphere motion online in a closed-loop manner [10].

In contrast, our work does not require costly robotic setups and demonstrates that sphere velocity can be recovered using a low-cost rolling-shutter camera.

3 Problem Formulation

For a fast-moving spherical object, we aim to estimate its translational velocity, $\mathbf{v}_{co} \in \mathbb{R}^3$, and angular velocity, $\boldsymbol{\omega}_{co} \in \mathbb{R}^3$, from measurements captured by

a rolling-shutter camera. For global-shutter cameras, which capture the entire frame instantaneously, the estimation of these velocities can be formulated as a simple nonlinear optimization problem using 2D–3D correspondences.

Specifically, for a 3D object point, $\mathbf{P}_o \in \mathbb{R}^3$, defined in the object frame $\mathcal{F}_o$, its position in the camera frame, $\mathcal{F}_c$, at timestamp k is defined as:

$$\mathbf{P}_{c,k} = \mathbf{R}_{co,k}\mathbf{P}_o + \mathbf{t}_{co,k}, \tag{1}$$

where $\mathbf{R}_{co,k} \in \mathbb{SO}(3)$ and $\mathbf{t}_{co,k} \in \mathbb{R}^3$ denote the rotation and translation of the object with respect to the camera frame at timestamp k, respectively. For brevity, we will omit the subscript "co" in the remainder of this section. Due to the short exposure time relative to the high-speed sphere motion, we assume constant velocity over a short time interval Δt_g from timestamp 0 to k. Under this assumption, the pose $(\mathbf{R}_k, \mathbf{t}_k)$ evolve as follows:

$$\mathbf{R}_k = \exp(\boldsymbol{\omega}_{co}^{\wedge}\Delta t_g)\,\mathbf{R}_0, \tag{2}$$

$$\mathbf{t}_k = \mathbf{t}_0 + \mathbf{v}_{co}\Delta t_g, \tag{3}$$

where $\mathbf{R}_0$ and $\mathbf{t}_0$ denote the rotation and translation of the sphere at timestamp 0. The operator $\exp(\cdot)$ denotes the exponential map from the Lie algebra $\mathfrak{so}(3)$ to the Lie group $\mathbb{SO}(3)$, and $(\cdot)^{\wedge}$ is the skew-symmetric operator that converts a vector in $\mathbb{R}^3$ into a matrix in $\mathfrak{so}(3)$. Given the corresponding 2D image measurement, $\mathbf{u} = [u_x, u_y]^{\top}$, of the 3D point $\mathbf{P}_o$, we can construct the following nonlinear optimization problem to estimate the velocities:

$$\mathbf{v}_{co}, \boldsymbol{\omega}_{co} = \arg\min_{\mathbf{v}_{co},\boldsymbol{\omega}_{co}} \sum_{i=0}^{N-1} \left\| \pi\big(\mathbf{R}_k\mathbf{P}_o^i + \mathbf{t}_k\big) - \mathbf{u}^i \right\|^2, \tag{4}$$

with $i \in [0, N-1]$ indexing the 3D points on the object, and $\pi(\cdot)$ denotes the pinhole camera projection of a 3D point using the camera intrinsic matrix $\mathbf{K}$. This optimization, for a global-shutter camera, minimizes the reprojection error to estimate the sphere's translational velocity $\mathbf{v}_{co}$ and angular velocity $\boldsymbol{\omega}_{co}$, assuming all pixels are captured simultaneously.

In contrast, a rolling-shutter camera acquires image rows sequentially, meaning that different rows correspond to different time instants. As a result, the image contains motion-induced distortion that must be modeled to accurately estimate $\mathbf{v}_{co}$ and $\boldsymbol{\omega}_{co}$. To this end, we introduce the parameter τ, representing the readout time between consecutive scanlines, which can be calibrated using a rapidly flashing LED. Unlike a global shutter, where the entire image is captured after a fixed interval Δt_g, the effective capture time of a specific row $\Delta t_r(\mathbf{u})$ in a rolling-shutter image becomes:

$$\Delta t_r(\mathbf{u}) = \Delta t_g + \tau\,(u_y - u_y(0)) \tag{5}$$

$$= \Delta t_g + \tau u_y, \tag{6}$$

where $u_y - u_y(0)$ denotes the row offset from the first scanline, i.e., the vertical pixel distance from the first image row to the current row. Substituting Eq. (5)

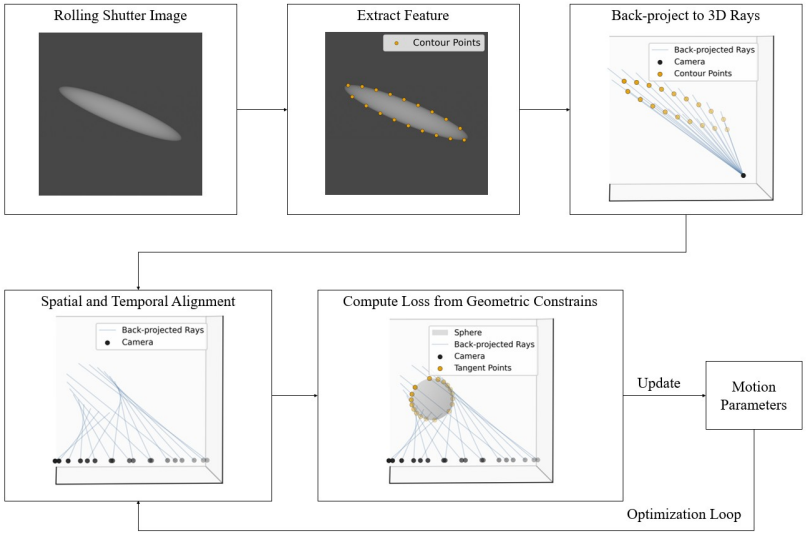


Fig. 2: Illustration of our pipeline for estimating the sphere's translational velocity. Detected image keypoints are first back-projected to 3D rays in object coordinates. The rays are then temporally and spatially aligned under the current motion parameters, and a loss is computed from the aligned rays to update the parameters. Spin estimation follows the same procedure.

into Eq. (2) and Eq. (3) extends the optimization in Eq. (4) to account for rolling-shutter effects.

However, solving such optimization in practice is challenging because the capture time $\Delta t(\mathbf{u})$ depends on the unknown pixel coordinate u_y, which in turn depends on the 3D-to-2D projection. This coupling creates an implicit nonlinear problem for each feature point, making direct optimization computationally difficult. To address this, we employ a backward-projection formulation, as detailed in the next section.

4 Methodology

In this section, we present our approach for estimating spherical motion under rolling-shutter image formation. To account for row-dependent capture timing, we adopt a back-projection formulation and decompose the problem into two decoupled optimizations for translational and angular velocities. An overview of the proposed pipeline is shown in Fig. 2. In the following sections, we first describe the back-projection formulation and coordinate representation in Sec. 4.1. We then detail the estimation of translational and angular velocities in Sec. 4.2 and Sec. 4.3, respectively. Finally, we extend our framework to the multi-camera and multi-frame setting in Sec. 4.4.

4.1 Back-Projection Formulation

By the principle of relative motion, observing a moving sphere with a static camera is equivalent to observing a static sphere with a moving camera. For consistency in our formulation and to simplify subsequent motion estimation, we express all quantities in the object coordinate frame $\mathcal{F}_o$.

For a 2D image measurement, $\mathbf{u} \in \mathbb{R}^2$, captured by a rolling-shutter camera, the pixel defines a back-projection ray originating from the camera optical center $\mathbf{x} \in \mathbb{R}^3$ and passing through the image plane. Expressed in the object frame $\mathcal{F}_o$, the ray is parameterized as:

$$\mathbf{r}_o(a, \mathbf{u}) = \mathbf{x}_o + a\,\mathbf{d}_o(\mathbf{u}), \quad a > 0, \tag{7}$$

where a denotes the depth along the ray, $\mathbf{x}_o \in \mathbb{R}^3$ is the camera center, and $\mathbf{d}_o \in \mathbb{R}^3$ is the back-projected viewing direction, both expressed in the object frame. The camera center and viewing direction are given by:

$$\mathbf{x}_o = \mathbf{t}_{oc} = -\mathbf{R}_{co}^\top \mathbf{t}_{co} \tag{8}$$

$$\mathbf{d}_o(\mathbf{u}) \propto \mathbf{R}_{oc}\mathbf{K}^{-1} \begin{bmatrix} u_x \\ u_y \\ 1 \end{bmatrix}, \ \|\mathbf{d}_o(\mathbf{u})\| = 1 \tag{9}$$

where $\mathbf{K}$ denotes the camera intrinsic matrix, and $\mathbf{t}_{oc}$ and $\mathbf{R}_{oc}$ represent the translation and rotation from the camera frame to the object frame, respectively.

For a moving camera that starts motion at a reference timestamp k_0, the elapsed time for a pixel measurement $\mathbf{u} = [u_x, u_y]^\top$ captured at timestamp k by a rolling-shutter camera is:

$$\Delta t\,(\mathbf{u}) = k - k_0 \tag{10}$$

$$= (k_s + \tau u_y) - k_0, \tag{11}$$

where k_s is the timestamp of the first scanline and τ is the readout time per row. Due to the rolling-shutter effect, each image row has a slightly different capture time, distorting rigid shapes. To preserve the geometry under rigid-body motion, we align all rays from time k to the reference timestamp k_0.

Let $\mathbf{v}_{oc}$ and $\boldsymbol{\omega}_{oc}$ denote the camera's translational and rotational velocities, related to the object's motion by $\mathbf{v}_{oc} = -\mathbf{R}_{co}^\top \mathbf{v}_{co}$ and $\boldsymbol{\omega}_{oc} = -\mathbf{R}_{co}^\top \boldsymbol{\omega}_{co}$. The aligned ray position and direction are then given by:

$$\mathbf{x}_{o,k_0}(\mathbf{u}, \mathbf{t}_{co}, \mathbf{v}_{co}, \boldsymbol{\omega}_{co}) = \exp(\Delta t(\mathbf{u})\boldsymbol{\omega}_{oc}^\wedge)\,(\mathbf{x}_o + \Delta t(\mathbf{u})\mathbf{v}_{oc}) \tag{12}$$

$$= \exp\big(-\Delta t(\mathbf{u})(\mathbf{R}_{co}^\top \boldsymbol{\omega}_{co})^\wedge\big)\,\big(\mathbf{x}_o - \Delta t(\mathbf{u})\mathbf{R}_{co}^\top \mathbf{v}_{co}\big)\,. \tag{13}$$

$$\mathbf{d}_{o,k_0}(\mathbf{u}, \mathbf{t}_{co}, \mathbf{v}_{co}, \boldsymbol{\omega}_{co}) = \exp(\Delta t(\mathbf{u})\boldsymbol{\omega}_{oc}^\wedge)\,\mathbf{d}_o(\mathbf{u}) \tag{14}$$

$$= \exp\big(-\Delta t(\mathbf{u})(\mathbf{R}_{co}^\top \boldsymbol{\omega}_{co})^\wedge\big)\,\mathbf{d}_o(\mathbf{u}), \tag{15}$$

For notational clarity, we omit the motion parameters $\mathbf{t}_{co}, \mathbf{v}_{co}, \boldsymbol{\omega}_{co}$ in the remainder of this paper. For pixel measurement $\mathbf{u} = [u_x, u_y]^\top$, the back-projection ray at the reference timestamp k_0 is finally represented as:

$$\mathbf{r}_{o,k_0}\,(a; \mathbf{u}) = \mathbf{x}_{o,k_0}\,(\mathbf{u}) + a\mathbf{d}_{o,k_0}\,(\mathbf{u})\,, \quad a > 0, \tag{16}$$

This backward projection formulation enables the estimation of both translational and rotational motion without requiring explicit 3D–2D correspondences [1, 17], as will be described in Secs. 4.2 and 4.3.

4.2 Translational Velocity Estimation

To estimate the translational velocity, as illustrated in Fig. 2, we extract 2D contour points of the observed sphere. Since the sphere's silhouette is invariant to rotation, the resulting contour constraints depend only on the sphere's position $\mathbf{t}_{co}$ and translational velocity $\mathbf{v}_{co}$. This enables a two-stage estimation procedure: we first determine the object position and translational velocity, then recover the angular velocity. This decomposition reduces the number of jointly optimized parameters from 12 to 6, improving optimization efficiency.

As illustrated in Fig. 2, let $\mathbf{z} = [\mathbf{u}_0^\top, \ldots, \mathbf{u}_{N-1}^\top]^\top$ be the set of detected 2D contour points, where $\mathbf{u}_i$ denotes the i-th measured point along the contour. For each contour measurement, we construct the projection ray using Eq. (7)–Eq. (16) under a motion hypothesis with zero angular velocity, yielding:

$$\mathbf{r}_{o,k_0}(a; \mathbf{u}_i) = \mathbf{x}_{o,k_0}(\mathbf{u}_i) + a\mathbf{d}_{o,k_0}(\mathbf{u}_i), \quad a > 0, \tag{17}$$

where $\mathbf{r}_{o,k_0}(a; \mathbf{u}_i)$ represents the ray associated with the i-th contour point. For brevity, we will omit the subscript "o, k_0" and use subscript "i" to denote correspondence to $\mathbf{u}_i$ in the following sections.

Along this ray, the 3D point that minimizes the Euclidean distance to the sphere center can be computed as:

$$\hat{a}_i = \arg\min_{a \in \mathbb{R}^+} \|\mathbf{x}_i + a\,\mathbf{d}_i\|, \tag{18}$$

which has the closed-form solution:

$$\hat{a}_i = \max\left(0, -\mathbf{d}_i^\top \mathbf{x}_i\right), \ \mathbf{P}_i = \mathbf{x}_i + \hat{a}_i\,\mathbf{d}_i. \tag{19}$$

where $\mathbf{P}_i$ is the obtained 3D point on the ray.

In the ideal case, when the sphere's initial position $\mathbf{t}_{co}$ and translational velocity, $\mathbf{v}_{co}$ are correct, the resulting 3D point $\mathbf{P}_i$ from the previous equation lies on the sphere surface, satisfying:

$$\|\mathbf{P}_i\| = r, \tag{20}$$

where r is the known sphere radius. To enforce this, we define the velocity-consistency loss:

$$\mathcal{L}_{\mathbf{v}}(\mathbf{t}_{co}, \mathbf{v}_{co}) = \frac{1}{N}\sum_{i=0}^{N-1}\left(\|\mathbf{P}_i\| - r\right)^2. \tag{21}$$

However, relying solely on this velocity loss may lead to a degenerate solution, where all rays collapse into a small region on the sphere surface. To prevent this,

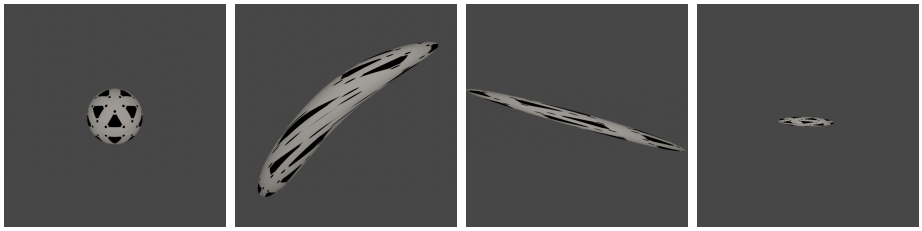


Fig. 3: Our surface pattern design for the sphere. The leftmost image shows the stationary sphere, while the remaining three are representative rolling-shutter images.

we introduce a regularization term that enforces a sufficient spatial spread of the reconstructed points:

$$\mathcal{R}_{\mathbf{v}}(\mathbf{t}_{co}, \mathbf{v}_{co}) = \max\left(0,\ \sigma^2 r^2 - \frac{1}{N}\sum_{i=0}^{N-1} \left\|\mathbf{P}_i - \bar{\mathbf{P}}\right\|^2\right), \tag{22}$$

where $\bar{\mathbf{P}}$ is the mean of the reconstructed 3D points, and σ determines the minimum allowable spread of the points, set empirically to $\sigma = \frac{1}{2}$. Our final objective is to minimize the following loss using the L-BFGS-B algorithm [7]:

$$\mathcal{J}_{\mathbf{v}}(\mathbf{t}_{co}, \mathbf{v}_{co}) = \mathcal{L}_{\mathbf{v}}(\mathbf{t}_{co}, \mathbf{v}_{co}) + \mathcal{R}_{\mathbf{v}}(\mathbf{t}_{co}, \mathbf{v}_{co}). \tag{23}$$

4.3 Angular Velocity Estimation

To recover the angular velocity $\boldsymbol{\omega}_{co}$, we design a structured surface pattern on the sphere (Fig. 3) and estimate it by enforcing relative surface feature constraints. Specifically, we assume a set of salient surface features arranged into point pairs with equal distances. Under the rolling-shutter motion model, these pairwise distances are constrained to remain constant, enabling the recovery of $\boldsymbol{\omega}_{co}$.

Let $z = [\mathbf{u}_0^\top, ..., \mathbf{u}_{N-1}^\top]^\top$ denote the set of detected surface feature points and $\mathcal{E} \subset \{0, ..., N-1\}^2$ denote the set of equidistant point pairs. Similar to Sec. 4.2, we back-project feature point $\mathbf{u}_i$ to 3D ray $\mathbf{r}_i$ under a motion hypothesis in which $\mathbf{t}_{co}$ and $\mathbf{v}_{co}$ are determined from the previous step. The first intersection between $\mathbf{r}_i$ and the sphere of radius r is obtained by solving

$$\|\mathbf{x}_i + \hat{a}_i\, \mathbf{d}_i\| = r, \tag{24}$$

which yields the following closed-form expression for the smaller root

$$\hat{a}_i = -\mathbf{d}_i^\top \mathbf{x}_i - \sqrt{\left(\mathbf{d}_i^\top \mathbf{x}_i\right)^2 - \left(\|\mathbf{x}_i\|^2 - r^2\right)}\ ,\ \mathbf{P}_i = \mathbf{x}_i + \hat{a}_i \mathbf{d}_i, \tag{25}$$

where $\mathbf{P}_i$ is the obtained surface point. We then construct the objective as the variance of the point-pair distances:

$$\mathcal{J}_{\omega}(\boldsymbol{\omega}_{co}) = \frac{1}{|\mathcal{E}|}\sum_{(i,j)\in\mathcal{E}} \|\mathbf{P}_i - \mathbf{P}_j\|^2 - \left(\frac{1}{|\mathcal{E}|}\sum_{(i,j)\in\mathcal{E}} \|\mathbf{P}_i - \mathbf{P}_j\|\right)^2. \tag{26}$$

4.4 Extension to Multi-Camera and Multi-Frame

Our formulation extends straightforwardly to multiple images, either captured by a single or multiple rolling-shutter cameras. Let $\{I_{c_0}, ..., I_{c_{N-1}}\}$ be the images captured by cameras $c_0, ..., c_{N-1}$. The relative rotation, $R_{c_0c_i}$, and relative translation, $\mathbf{t}_{c_0c_i}$, of camera c_i with respect to c_0 can be obtained through calibration. To align back-projected rays to object frame $\mathcal{F}_o$, we modify Eqs. (8) and (9) to

$$\mathbf{x}_o = \mathbf{R}_{c_0o}^\top(\mathbf{t}_{c_0c_i} - \mathbf{t}_{c_0o}) \tag{27}$$

$$\mathbf{d}_o(\mathbf{u}) \propto \mathbf{R}_{c_0o}^\top \mathbf{R}_{c_0c_i} \mathbf{K}_{c_i}^{-1} \begin{bmatrix} u_x \\ u_y \\ 1 \end{bmatrix} \;,\; \|\mathbf{d}_o(\mathbf{u})\| = 1 \tag{28}$$

where $\mathbf{K}_{c_i}$ is the intrinsic matrix of camera c_i. The subsequent derivation of back-projection follows identically to the single-camera case.

To solve for translational velocity under a multi-image scenario, we calculate objective $\mathcal{J}_{\mathbf{v}}^{(i)}(\mathbf{t}_{c_0o}, \mathbf{v}_{c_0o})$ for each image I_i and jointly optimize the objective as:

$$\mathcal{J}_{\mathbf{v}}(\mathbf{t}_{c_0o}, \mathbf{v}_{c_0o}) = \sum_{i=0}^{N-1} \mathcal{J}_{\mathbf{v}}^{(i)}(\mathbf{t}_{c_0o}, \mathbf{v}_{c_0o}). \tag{29}$$

Based on the previously estimated motion parameters, we subsequently optimize the angular velocity in a similar manner using the L-BFGS-B solver [7]:

$$\mathcal{J}_{\boldsymbol{\omega},1}(\boldsymbol{\omega}_{c_0o}) = \sum_{i=0}^{N-1} \mathcal{J}_{\boldsymbol{\omega}}^{(i)}(\boldsymbol{\omega}_{c_0o}). \tag{30}$$

When the images are captured with an inter-frame delay, we further refine the estimate by exploiting the larger temporal baseline. Let $\mathcal{P}_i$ denote the projected surface points corresponding to image I_i. We define an inter-frame objective based on the truncated Chamfer distance. The one-way truncated squared Chamfer distance from the set $\mathcal{A}$ to the set $\mathcal{B}$ is given by

$$d_{\mathrm{ch}}(\mathcal{A}, \mathcal{B}) = \frac{1}{|\mathcal{A}|} \sum_{\boldsymbol{a} \in \mathcal{A}} \min\left(D,\ \min_{\boldsymbol{b} \in \mathcal{B}} \|\boldsymbol{a} - \boldsymbol{b}\| \right)^2, \tag{31}$$

where D is a truncation hyper-parameter that limits the pairwise matching distance to mitigate the effect of incomplete correspondences between frames. Using this metric, we introduce the refinement loss:

$$\mathcal{J}_{\boldsymbol{\omega},2}(\boldsymbol{\omega}_{c_0o}) = \sum_{i=0}^{N-1} \sum_{j=0}^{N-1} d_{\mathrm{ch}}(\mathcal{P}_i, \mathcal{P}_j)\,. \tag{32}$$

The result of Eq. (30) serves as the initialization for this refinement step. The objective $\mathcal{J}_{\boldsymbol{\omega},2}$ enforces temporal consistency by aligning the back-projected surface points across multiple frames. We illustrate this effect in Fig. 4.

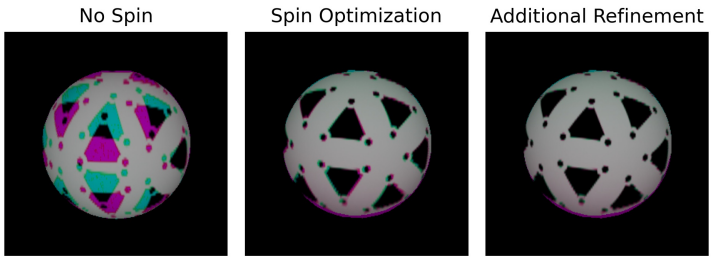


Fig. 4: Proposed spin refinement step. Left: overlay of two frames captured 2 ms apart. Middle: alignment after unwarping using spin estimated from geometric constraints only, with minor residual misalignment remaining. Right: alignment after the refinement step, resulting in near-perfect pattern overlap.

5 Experimental Setup

We describe the experimental setup used for evaluating the proposed method, first in simulation and then in real-world scenarios.

5.1 Pattern Design

As discussed in Sec. 1 and Sec. 4.3, recovering rotational velocity requires surface features on the sphere. We therefore design a robust, easily detectable surface pattern, illustrated in Fig. 3. The pattern is derived from an icosahedron and consists of 20 groups of dots, each forming an equilateral triangle with a shaded interior for robust grouping. The shaded triangles provide a robust cue for grouping the dots, enabling reliable detection of equidistant point pairs by the feature extraction pipeline shown in Fig. 5.

The icosahedron-inspired layout ensures that multiple triangles remain fully visible from most viewpoints, providing stable geometric constraints for spin estimation. In rare edge cases, a triangle vertex may lie near the silhouette and become difficult to detect; therefore, we slightly shrink the triangles to increase the number of reliably visible points.

5.2 Rolling-Shutter Simulation and Baseline

We implement a rolling-shutter simulator in Blender [6]. For each trial, a patterned sphere is initialized with a random pose, and its motion is simulated under constant translational and angular velocities. A sequence of intermediate global-shutter frames is rendered, from which a single rolling-shutter image is synthesized by selecting the corresponding scanlines. Representative simulated rolling-shutter images are shown in Fig. 3. In our simulation, we set the camera resolution to 2048×2048, the focal length to 3000 pixels, and the readout time to $10\,\mu$s. The camera is positioned 30 cm from the sphere. We sample translational and angular velocity directions uniformly on the unit sphere, with magnitudes drawn uniformly from 0–50 m/s and 0–5000 rpm, respectively.

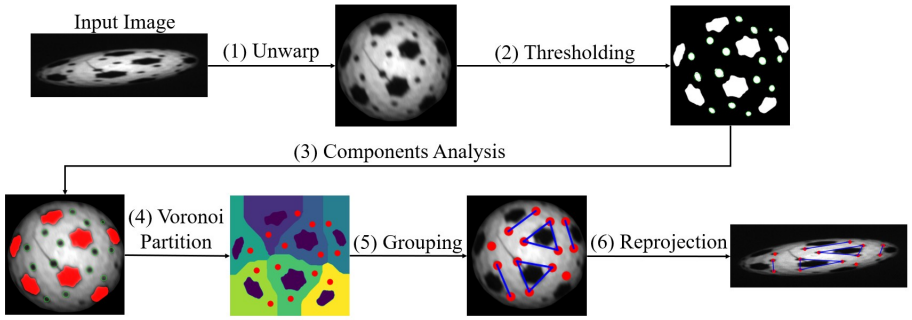


Fig. 5: Feature-extraction pipeline for the designed pattern. (1) Unwrap the RS image using the predicted velocity. (2) Apply adaptive thresholding to segment the pattern. (3) Classify connected components as dots or markers. (4) Partition the image into Voronoi cells induced by the markers. (5) Group dots into edges according to the Voronoi cells. (6) Reproject the detected feature coordinates into the original image.

We apply our feature-extraction pipeline, followed by the proposed motion estimation algorithm, to one or more simulated images, and compare the estimated velocities with the ground-truth simulation parameters. For translational velocity, we perform a coarse grid search over initial hypotheses {-25, 0, 25} m/s along each axis. The spin optimization is initialized at $\boldsymbol{\omega}_{co} = [0, 0, 0]$.

We evaluate four scenarios: (1) a single image; (2) two images captured simultaneously by a stereo pair with a 10 cm baseline; (3) two images captured by the same camera with a 2 ms inter-frame delay; and (4) two images captured by the same stereo pair but with 2 ms inter-frame delay. All scenarios are evaluated on the same set of 100 simulation trials. We express all quantities in the camera coordinate frame, where the x-axis corresponds to the horizontal image direction, the y-axis to the vertical image direction, and the z-axis to the camera viewing direction.

Since existing rolling-shutter methods do not estimate 3D translational and angular velocities without explicit 3D–2D correspondences, we compare our approach with the nonlinear formulation of Ait-Aider et al. [1], which provides a solution for general objects. We provide the baseline method with ground-truth 3D–2D correspondences, while our method operates without correspondence information. To ensure comparable computational budget, both methods are initialized using the same coarse grid search strategy over motion hypotheses. In addition, we include results from directly applying a global-shutter perspective-n-point (PnP) method as a reference baseline.

5.3 Real-World Setup and Baseline

We evaluate our method in a challenging real-world application: a vision-based golf launch monitor. A well-struck golf ball can reach speeds of up to 70 m/s and spin rates of up to 10,000 rpm. Our overall device design is illustrated in Fig. 7a.

	1. Single			2. Stereo			3. Multi-Frame			4. Stereo MF		
	x	y	z	x	y	z	x	y	z	x	y	z
Global Shutter	-	-	-	-	-	-	11.57	8.03	30.58	38.42	8.55	39.21
Ait-Aider*et al.* [1]	3.24	10.14	14.33	3.25	7.80	10.87	1.44	1.12	12.40	2.76	1.16	9.84
Ours	0.23	4.61	2.95	0.12	0.23	0.19	0.16	0.12	0.36	0.19	0.10	0.26

Table 1: MAE of the translational velocity components for Scenarios 1–4 (m/s).

	1. Single			2. Stereo			3. Multi-Frames			4. Stereo MF		
	x	y	z	x	y	z	x	y	z	x	y	z
Global Shutter	-	-	-	-	-	-	131.00	126.50	120.88	132.52	123.50	119.31
Ait-Aider*et al.* [1]	164.16	172.60	138.25	163.96	188.64	137.85	142.31	133.22	81.99	153.16	151.77	114.63
Ours	49.82	17.18	27.89	33.11	20.60	10.13	8.48	7.22	7.51	13.58	23.98	6.68
Ours + Ref.	-	-	-	-	-	-	4.58	2.41	1.10	10.39	3.93	2.68

Table 2: MAE of the angular velocity components for Scenarios 1–4 (rad/s).

We build the prototype using two 6.5 MP rolling-shutter cameras with a fixed baseline. The cameras are oriented with a 90-degree in-plane rotation so that the scan direction opposes the dominant ball motion, relaxing trigger-timing constraints. Otherwise, the ball may leave the field of view before the frame readout is completed. Since the dominant velocity direction is approximately known in this setup, we directly initialize the optimization, omitting the coarse grid search used in simulation.

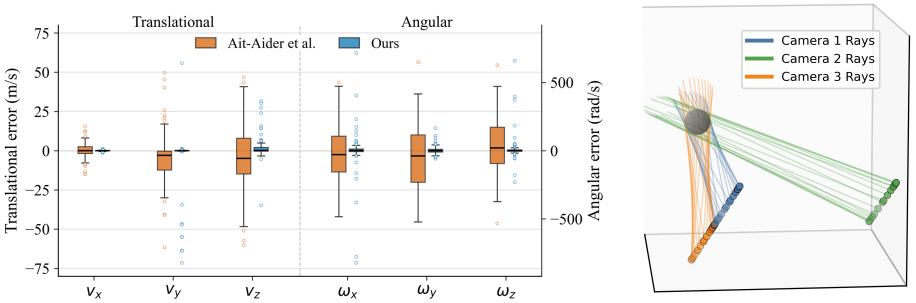
We evaluate accuracy by placing our device alongside a commercially available launch monitor (SkyTrak [23]) and comparing estimated velocity over the same set of shots. Note that SkyTrak is used as an external baseline rather than ground truth. According to its published specifications, the stated measurement tolerances are ± 1 mph for ball speed, $\pm 1^\circ$ for launch angle, $\pm 2^\circ$ for side angle, and ± 250 rpm for both backspin and sidespin.

6 Results

In this section, we first evaluate the proposed method on synthetic data, followed by experiments on real-world measurements where we compare our results with an external baseline.

6.1 Synthetic Evaluation

Following our two-step optimization strategy, we first estimate translational velocity and report the mean absolute errors (MAE) in Tab. 1. We then fix the estimated velocities and optimize for angular velocity. The MAE results for angular velocity are reported in Tab. 2. The results demonstrate that, across all



(a) Error distribution with single frame. **(b)** Multi-view constraints.

Fig. 6: (a) Boxplots of the error distribution obtained with the single-frame, single-camera setup. We report the mean and standard deviation of velocity errors for both the baseline and our method. (b) Multi-view constraints for velocity estimation. Motion primarily along the camera depth axis is ambiguous in a single rolling-shutter image (Camera 1). A second frame (Camera 2) or an additional viewpoint (Camera 3) provides complementary constraints that help resolve this ambiguity.

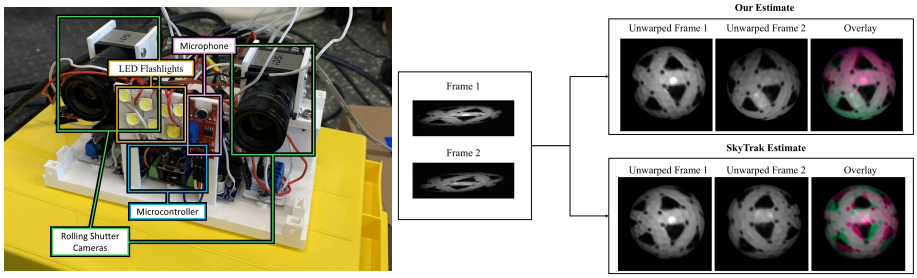
configurations, our proposed system consistently outperforms both Ait-Aider et al. [1] and the global-shutter baselines by a substantial margin in both translational and angular velocity estimation. Under the same rolling-shutter setup, the method of Ait-Aider et al. [1] exhibits large errors even when provided with ground-truth correspondences, due to its sensitivity to initialization under high translational and angular velocities.

We also observe that our algorithm exhibits larger mean errors and variance along the y and z axes in the single-camera setting, as illustrated in Fig. 6a. This behavior is expected because motion along the y and z axes can produce similar rolling-shutter distortions, leading to ambiguities when observed from a single viewpoint. To resolve this ambiguity, introducing a second view or an extra frame provides additional constraints, as shown in Fig. 6b, resulting in more reliable velocity estimates. The improvement from the two-camera configuration is more noticeable for rotational velocity, as illustrated in Tab. 2, which benefits from the additional spin-refinement step (Fig. 4).

Additionally, we conduct an ablation study on the robustness of our approach with respect to different sphere radii. We evaluate settings with varying radii (80%, 100%, and 120% of the default radius) in the single-image setting. As shown in Tab. 3, a larger radius generally yields smaller errors due to higher

	$\mathbf{e}_{\mathbf{v}_x}$	$\mathbf{e}_{\mathbf{v}_y}$	$\mathbf{e}_{\mathbf{v}_z}$	$\mathbf{e}_{\boldsymbol{\omega}_x}$	$\mathbf{e}_{\boldsymbol{\omega}_y}$	$\mathbf{e}_{\boldsymbol{\omega}_z}$
80% radius	0.32	3.46	4.90	45.32	31.54	44.02
100% radius	0.23	4.61	2.95	49.82	17.18	27.89
120% radius	0.19	3.50	3.63	38.14	14.86	22.28

Table 3: Effect of radius on $\mathbf{v}(m/s)$ and $\boldsymbol{\omega}(rad/s)$ error in single-image setting.



(a) (b)

Fig. 7: Our prototype setup and qualitative comparison for an outlier case. (a) Our prototype hardware setup. The flash and cameras are triggered by the impact sound recorded by a microphone. The two rolling-shutter cameras are vertically mounted and captured with a 2 ms inter-frame delay. (b) We validate the estimated spin by overlaying the unwarped images (red and green) after warping them using the estimated angular velocity. Accurate estimates result in strong alignment of the pattern.

image resolution of the ball. However, for the largest radii, slight increases in the errors of $\mathbf{v}_y$, $\mathbf{v}_z$, and $\boldsymbol{\omega}_x$ are observed due to increased variance (Fig. 6a).

6.2 Real-World Application Evaluation

For the real-world experiments, we recorded 10 different golf shots using both our system and a commercially available launch monitor (SkyTrak [23]). The estimated translational velocity is converted to the speed–launch/side–angle representation reported by SkyTrak. The third spin component is omitted, as it is not provided by the device.

We present the full comparison in Tab. 4. Across all 10 shots, our method produces speed and launch/side-angle estimates that closely match those reported by SkyTrak, demonstrating strong overall consistency across real-world trials. Spin estimates also show good agreement for 9 out of 10 shots, indicating reliable recovery of rotational dynamics under most conditions. However, shot 10 exhibits a noticeable discrepancy in spin, suggesting a potential inconsistency between the two systems.

To further examine the abnormal result from shot 10, we unwarp the captured images using the estimated velocities and qualitatively inspect the resulting reconstructions. As shown in Fig. 7b, the reconstruction produced by our method exhibits stronger temporal consistency. Since SkyTrak estimates spin from consecutive global-shutter frames, differences in texture alignment may contribute to the observed discrepancy. In contrast, our approach leverages rolling-shutter distortions as additional constraints, which can help resolve such ambiguities.

Data	Speed (mph)			Launch Angle (deg)			Side Angle (deg)			Backspin (rpm)			Sidespin (rpm)		
	Ours	SkyTrak	Diff	Ours	SkyTrak	Diff	Ours	SkyTrak	Diff	Ours	SkyTrak	Diff	Ours	SkyTrak	Diff
1	107.4	106	1.4	25.6	25	0.6	0.9	-3	3.9	3856	3583	273	698	780	-82
2	97.2	96	1.2	25.6	25	0.6	-1.3	-6	4.7	3731	3794	-63	530	0	530
3	122.2	120	2.2	26.2	25	1.2	-1.4	5	-6.4	4320	4150	170	732	237	495
4	113.8	107	6.8	27.3	27	0.3	2.8	3	-0.2	4255	4159	96	482	0	482
5	118.2	116	2.2	25.6	24	1.6	3.7	0	3.7	4408	4511	-103	495	453	42
6	55.1	54	1.1	26.6	26	0.6	2.9	0	2.9	2741	2664	77	602	383	219
7	99.3	95	4.3	24.2	23	1.2	0.6	-3	3.6	4365	4249	116	858	798	60
8	69.7	70	-0.3	27.8	26	1.8	-7.4	-4	-3.4	3269	3208	61	468	368	100
9	100	98	2	25.3	24	1.3	1.5	4	-2.5	4215	4218	-3	518	302	216
10	86.3	89	-2.7	25.5	25	0.5	0.2	0	0.2	3805	11254	-7449	667	-1129	1796
MAE			2.4			1.0			3.2			841			402
MAE w/o #10			2.4			1.0			3.5			107			247

Table 4: Comparison between our device and SkyTrak [23] on 10 golf shots. Differences are computed as (Ours – SkyTrak). Data 10 is treated as an outlier, and its anomalous measurements are highlighted in red.

7 Conclusion

In this work, we present a correspondence-free formulation for estimating the 3D translational and angular velocity of spherical objects from rolling-shutter imagery. Rather than correcting rolling-shutter distortions, we exploit them as a rich source of temporal information for velocity estimation. We design a robust surface pattern on the sphere and formulate the estimation as a backward-projection optimization. This avoids explicit 3D–2D correspondences on repetitive spherical textures while still exploiting the structured distortions induced by rolling-shutter readout. We adopt a two-stage optimization strategy that explicitly decouples translational and rotational estimation, improving robustness while reducing optimization complexity. The formulation extends naturally from single-image to multi-image and multi-camera settings, where increased temporal baselines and additional viewpoints provide stronger geometric constraints. Experiments in both simulation and real-world scenarios demonstrate accurate motion recovery across a wide range of configurations.

There are several limitations of our proposed approach. First, although it does not require explicit correspondences, the system still depends on additional known parameters, such as the sphere radius, and on a carefully designed surface pattern. Second, our method is currently restricted to spherical objects, rather than other non-spherical geometries. Third, we assume high-speed motion and model the sphere's motion with a constant-velocity assumption, which may limit its applicability to more complex, time-varying motion.

In future work, we will relax the assumption of known parameters. Moreover, we will extend our method beyond spherical objects to more general rigid bodies, including ellipsoids and other non-spherical shapes.

References

1. Ait-Aider, O., Andreff, N., Lavest, J., Martinet, P.: Simultaneous object pose and velocity computation using a single view from a rolling shutter camera. In: Computer Vision – ECCV 2006. Lecture Notes in Computer Science, vol. 3952, pp. 56–68. Springer (2006)
2. Ait-Aider, O., Bartoli, A., Andreff, N.: Kinematics from lines in a single rolling shutter image. In: IEEE Conf. Comput. Vis. Pattern Recog. pp. 1–6 (2007)
3. Albl, C., Kukelova, Z., Larsson, V., Pajdla, T.: Rolling shutter camera absolute pose. IEEE Trans. Pattern Anal. Mach. Intell. **42**(6), 1439–1452 (2020)
4. Albl, C., Kukelova, Z., Pajdla, T.: R6P - rolling shutter absolute camera pose. In: IEEE Conf. Comput. Vis. Pattern Recog. pp. 2292–2300 (2015)
5. Baker, S., Bennett, E., Kang, S.B., Szeliski, R.: Removing rolling shutter wobble. In: IEEE Conf. Comput. Vis. Pattern Recog. pp. 2392–2399. IEEE (2010)
6. Blender Foundation: Blender: A 3D modeling and rendering package. `https://www.blender.org/` (2026), accessed 2026
7. Byrd, R.H., Lu, P., Nocedal, J., Zhu, C.: A limited memory algorithm for bound constrained optimization. SIAM J. Sci. Comput. **16**(5), 1190–1208 (1995)
8. Cant, O.: Exploring the effects of ball speed and spin in grand slam tennis matchplay. Victoria University (2020)
9. Chaumette, F., Hutchinson, S.: Visual servo control. I. basic approaches. IEEE Robot. Autom. Mag. **13**(4), 82–90 (2006)
10. Dahmouche, R., Andreff, N., Mezouar, Y., Ait-Aider, O., Martinet, P.: Dynamic visual servoing from sequential regions of interest acquisition. Int. J. Robotics Res. **31**(4), 520–537 (2012)
11. Dai, Y., Li, H., Kneip, L.: Rolling shutter camera relative pose: Generalized epipolar geometry. In: IEEE Conf. Comput. Vis. Pattern Recog. pp. 4132–4140 (2016)
12. Grundmann, M., Kwatra, V., Castro, D., Essa, I.: Calibration-free rolling shutter removal. In: IEEE Int. Conf. Comput. Photography. pp. 1–8 (2012)
13. Hedborg, J., Forssén, P.E., Felsberg, M., Ringaby, E.: Rolling shutter bundle adjustment. In: IEEE Conf. Comput. Vis. Pattern Recog. pp. 1434–1441 (2012)
14. Kienzle, D., Schön, R., Lienhart, R., Satoh, S.: Towards ball spin and trajectory analysis in table tennis broadcast videos via physically grounded synthetic-to-real transfer. In: IEEE Conf. Comput. Vis. Pattern Recog. Worksh. pp. 5888–5897 (2025)
15. Liao, B., Qu, D., Xue, Y., Zhang, H., Lao, Y.: Revisiting rolling shutter bundle adjustment: Toward accurate and fast solution. In: IEEE Conf. Comput. Vis. Pattern Recog. pp. 4863–4871 (2023)
16. Liu, P., Cui, Z., Larsson, V., Pollefeys, M.: Deep shutter unrolling network. In: IEEE Conf. Comput. Vis. Pattern Recog. pp. 5941–5949 (2020)
17. Magerand, L., Bartoli, A., André, V., Pizarro, D.: Global optimization of object pose and motion from a single rolling shutter image with automatic 2D-3D matching. In: Computer Vision – ECCV 2012. Lecture Notes in Computer Science, vol. 7572, pp. 456–469. Springer (2012)
18. Magerand, L., Bartoli, A.: A generic rolling shutter camera model and its application to dynamic pose estimation. In: Int. Symp. 3D Data Process. Vis. Trans. (2010)
19. Meingast, M., Geyer, C., Sastry, S.S.: Geometric models of rolling-shutter cameras. arXiv:cs/0503076 (2005)

20. Nakabayashi, T., Higa, K., Yamaguchi, M., Fujiwara, R., Saito, H.: Event-based ball spin estimation in sports. In: IEEE Conf. Comput. Vis. Pattern Recog. Worksh. pp. 3367–3375 (2024)
21. Rengarajan, V., Balaji, Y., Rajagopalan, A.N.: Unrolling the shutter: CNN to correct motion distortions. In: IEEE Conf. Comput. Vis. Pattern Recog. pp. 2345–2353 (2017)
22. Saurer, O., Pollefeys, M., Lee, G.H.: A minimal solution to the rolling shutter pose estimation problem. In: IEEE/RSJ Int. Conf. Intell. Robots Syst. pp. 1328–1334 (2015)
23. SkyTrak: Skytrak launch monitor. `https://skytrakgolf.com` (2026), accessed 2026
24. Tamaki, T., Sugino, T., Yamamoto, M.: Measuring ball spin by image registration. In: Korea–Japan Joint Workshop Frontiers Comput. Vis. pp. 269–274. Frontiers of Computer Vision (2004)
25. Tamaki, T., Wang, H., Raytchev, B., Kaneda, K., Ushiyama, Y.: Estimating the spin of a table tennis ball using inverse compositional image alignment. In: ICASSP. pp. 1457–1460 (2012)