# EBS-EKF: Accurate and High Frequency Event-based Star Tracking

Supplementary Material

## 6. Overview

We include a video titled *Short video overview.mp4* along with this document, which provides a narrated overview of our paper. Section 7 derives our extended Kalman filter (EKF) for event-based star tracking. Section 8 provides additional details on our tracking algorithm. Section 9 provides implementation details and a discussion on our laboratory setup used to perform centroiding experiments. Section 10 provides more details on our synchronized EVK4-HD event camera and APS star tracker, including their hardware configurations, synchronization scheme, and procedures for obtaining the brightness-dependent offset curve. Section 11 provides additional details on our night sky dataset. Section 12 provides plots for the tracks summarized in Table 2 of the main text.

## 7. State estimation with an Extended Kalman Filter (EKF)

This section provides more details on estimating our state using an EKF. Restated from the main text, our measurements are the set of $N$, positive events $\mathcal{E} = \{\mathbf{x}, t\}_{i=0:N}$, where we denote each event as $\mathbf{e}_i \in \mathcal{E}$.

### 7.1. Bayesian State Estimation with a Kalman Filter

Under a Markov assumption, the probability distribution over all states and measurements is,

$$p(\mathbb{S}_0, ..., \mathbb{S}_N; e_0, ..., e_N) = p(\mathbb{S}_0) \prod_{i=0}^{N} \mathcal{L}(e_i|\mathbb{S}_i) p(\mathbb{S}_i|\mathbb{S}_{i-1}). \tag{9}$$

The Kalman filter uses predict and update phases to recursively estimate the posterior distribution of the states conditioned on measurements up to the current timestep. As shown in [4], the predicted state is the distribution associated with the previous state, over all possible previous states,

$$p(\mathbb{S}_i|\mathcal{E}_{i-1}) = \int p(\mathbb{S}_i|\mathbb{S}_{i-1}) p(\mathbb{S}_{i-1}|\mathcal{E}_{i-1}) d\mathbb{S}_{i-1} \tag{10}$$

where the measurements up to the current timestep $t$ are $\mathcal{E}_t = \{e_1, ..., e_t\}$. The update step is the product of the measurement likelihood and the predicted state,

$$p(\mathbb{S}_i|\mathcal{E}_{i-1}) = \frac{\int p(e_i|\mathbb{S}_i) p(\mathbb{S}_i|\mathcal{E}_{i-1}) d\mathbb{S}_i}{p(\mathbb{S}_i|\mathcal{E}_{i-1})} \tag{11}$$

We refer interested readers to Barker et al., [4] for more details on these distributions and how they are estimated by the EKF equations that we define in the next section.

### 7.2. EKF Predict and Update Matrix Equations

#### 7.2.1. Measurement Geometry

We aim to use EBS measurements of stars to estimate the camera state $\mathbb{S} = [\mathbf{q}, \omega]$, where $\mathbf{q} \in SO(3)$ is a quaternion encoding a valid 3D rotation, and $\omega \in \mathbb{R}^3$ denotes the 3D angular velocity. For our work, we consider the catalog of stars $\mathbf{S} \in \mathbb{R}^{N\times 3}$ with apparent magnitude $\leq 7$. Each of the $N$ stars is a 3D point normalized to the unit sphere with negligible parallax effects [12]. A single star $\mathbf{s}_i \in \mathbf{S}$ with 3D coordinates $(X, Y, Z)$ is rotated into the camera's frame using a world-to-camera quaternion $\mathbf{q}_{\text{wc}}$[‖],

$$\hat{\mathbf{s}}_i = \mathbf{q}_{\text{wc}}(\mathbf{s}_i), \tag{12}$$

and projected onto the 2D imaging plane using the pinhole camera model

$$x = f\frac{X}{Z}, \quad y = f\frac{Y}{Z}, \tag{13}$$

where $f$ is the focal length.

#### 7.2.2. EKF Predict and Update Equations

Our state is a compound manifold defined by $SO(3) \times \mathbb{R}^3$ (i.e., the camera's rotation and angular velocity). To estimate our state with an EKF, we employ the boxplus operator $\boxplus$ [18], which defines the addition of a 6D residual vector $[\delta_{\theta_x}, \delta_{\theta_y}, \delta_{\theta_z}, \delta_{\omega_x}, \delta_{\omega_y}, \delta_{\omega_z}]$ to our state manifold. Formally, the operator is defined as,

$$\boxplus : \mathbb{S} \times \mathbb{R}^6 \to \mathbb{S}, \tag{14}$$

and provides a mechanism for adding a change in rotation and velocity to our state quaternion and velocity.

We use a constant velocity prior to have the state attitude respond proportionally to the velocity and time,

$$\mathbf{q}^{t+1} = \mathbf{q}^t \boxplus \Delta t\omega = \exp(\Delta t\omega) \cdot \mathbf{q}^t. \tag{15}$$

Here, the boxplus operator uses the exponential map to project a change in angle to the quaternion's Lie group, where it is then multiplied with the quaternion to update the rotation [18]. With a constant velocity prior defined, the estimation of the state mean $\mathbf{u}$ is given by the standard EKF equations,

$$\bar{\mathbf{u}} = f(\mathbf{u}), \tag{16}$$

$$\mathbf{F} = \left.\frac{\mathrm{d}f(\mathbf{u}^t)}{\mathrm{d}\mathbf{u}}\right|_{\mathbf{u}^t}, \tag{17}$$

$$\bar{\mathbf{P}} = \mathbf{F}\mathbf{P}\mathbf{F}^\top + \mathbf{Q}. \tag{18}$$

[‖] We typically omit the wc subscript for brevity.

Eq. 17 defines the Jacobian for the constant velocity model $f(\cdot)$ at time $t$. Eq. 15 uses the model to predict the state mean $\bar{\mathbf{u}}$. Eq. 18 updates the state uncertainty $\mathbf{P}$ using the linearized model $\mathbf{F}$ and process uncertainty $\mathbf{Q}$.

The EKF update phase is defined with the following equations:

$$\mathbf{H} = \frac{\mathrm{d}h(\bar{\mathbf{u}}^t)}{\mathrm{d}\mathbf{u}}\bigg|_{\mathbf{u}^t}, \tag{19}$$

$$\mathbf{y} = \mathbf{z} - h(\bar{\mathbf{u}}), \tag{20}$$

$$\mathbf{K} = \bar{\mathbf{P}}\mathbf{H}^\top(\mathbf{H}\bar{\mathbf{P}}\mathbf{H}^\top + \mathbf{R})^{-1}, \tag{21}$$

$$\mathbf{u} = \bar{\mathbf{u}} \boxplus \mathbf{K}\mathbf{y}, \tag{22}$$

$$\mathbf{P} = (\mathbf{I} - \mathbf{K}\mathbf{H})\bar{\mathbf{P}}. \tag{23}$$

Eq. 19 defines a Jacobian for the measurement model $h(\cdot)$. Eq. 20 is the residual between a measurement $\mathbf{z}$ and the predicted state measurement $h(\bar{\mathbf{x}})$. Eq. 21 defines the Kalman gain, which is used in Eq. 22 to interpolate the prior and measurement via the boxplus operator. Finally, Eq. 23 uses the Kalman gain to update the state covariance.

## 7.3. Derivation of F, Q, H

### 7.3.1. Forward model Jacobian:

The forward model jacobian is a matrix $\mathbf{F} \in \mathbb{R}^{6\times6}$ that linearizes Eq 15. Our Jacobian matrix is comprised of four partial derivatives,

$$\mathbf{F} = \begin{bmatrix} \frac{\mathrm{d}\mathbf{q}^{t+1}}{\mathrm{d}\mathbf{q}^t} & \frac{\mathrm{d}\mathbf{q}^{t+1}}{\mathrm{d}\omega^t} \\ \frac{\mathrm{d}\omega^{t+1}}{\mathrm{d}\mathbf{q}^t} & \frac{\mathrm{d}\omega^{t+1}}{\mathrm{d}\omega^t}. \end{bmatrix} \tag{24}$$

Using identities defined by Bloesch et al. [6], we derive the first partial derivative w.r.t the attitude as

$$\frac{\mathrm{d}\mathbf{q}^{t+1}}{\mathrm{d}\mathbf{q}^t} = \frac{\mathrm{d}}{\mathrm{d}\mathbf{q}^t}\left[\exp(\Delta t\omega^t)\cdot\mathbf{q}^t\right] = \mathbf{C}(\exp(\Delta t\omega^t)) \tag{25}$$

$$= \mathbf{I} + \frac{\sin(||\Delta t\omega^t||)(\Delta t\omega^t)^\times}{||\Delta t\omega^t||} + \frac{(1-\cos(||\Delta t\omega^t||)(\Delta t\omega)^{\times^2}}{||\Delta t\omega^t||^2} \tag{26}$$

$$\approx \mathbf{I} + (\Delta t\omega^t)^\times \tag{27}$$

where $\mathbf{I} \in \mathbb{R}^{3\times3}$ is the identity matrix, $\mathbf{C}(\cdot)$ is Rodriguez' formula applied to an exponential mapped rotation, as defined by Bloesch et al. [6], and $\mathbf{v}^\times = (v_1, v_2, v_3)^\times$ denotes the $3\times3$ skew-symmetric matrix,

$$\mathbf{v} = \begin{bmatrix} 0 & -v_3 & v_2 \\ v_3 & 0 & -v_1 \\ -v_2 & v_1 & 0 \end{bmatrix}. \tag{28}$$

We then derive the partial derivative with respect to velocity. Using the chain rule,

$$\frac{\mathrm{d}\mathbf{q}^{t+1}}{\mathrm{d}\omega^t} = \frac{\mathrm{d}}{\mathrm{d}\omega^t}\left[\exp(\Delta t\omega^t)\cdot\mathbf{q}^t\right] = \mathbf{\Gamma}(\Delta t\omega^t)\Delta t, \tag{29}$$

where $\mathbf{\Gamma}$ is the Jacobian of the exponential map [6] and expanded as

$$\mathbf{\Gamma}(\Delta t\omega^t) = \mathbf{I} + \frac{(1-\cos(||\Delta t\omega^t||))(\Delta t\omega^t)^\times}{||\Delta t\omega^t||^2} + \tag{30}$$

$$\frac{(||\Delta t\omega^t|| - \sin(||\Delta t\omega^t||))(\Delta t\omega^t)^{\times^2}}{||\Delta t\omega^t||^3} \tag{31}$$

$$\approx (\mathbf{I} + 1/2(\Delta t\omega^t)^\times). \tag{32}$$

For the velocity transition function $\omega^{t+1} = \omega^t$, the derivative with respect to the attitude is simply $\frac{\mathrm{d}\omega^{t+1}}{\mathrm{d}\mathbf{q}^t} = 0_{3\times3}$, and the derivative with respect to velocity is the identity matrix, $\frac{\mathrm{d}\omega^{t+1}}{\mathrm{d}\omega^t} = \mathbf{I}_{3\times3}$. Filling in the four partial derivatives yields the $\mathbf{F}$ matrix,

$$\mathbf{F} = \begin{bmatrix} 1 & -\Delta t\cdot\omega_3 & \Delta t\cdot\omega_2 & \Delta t & -\frac{(\Delta t)^2\cdot\omega_3}{2} & \frac{(\Delta t)^2\cdot\omega_2}{2} \\ \Delta t\cdot\omega_3 & 1 & -\Delta t\cdot\omega_1 & \frac{(\Delta t)^2\cdot\omega_3}{2} & \Delta t & -\frac{(\Delta t)^2\cdot\omega_1}{2} \\ -\Delta t\cdot\omega_2 & \Delta t\cdot\omega_1 & 1 & -\frac{(\Delta t)^2\cdot\omega_2}{2} & \frac{(\Delta t)^2\cdot\omega_1}{2} & \Delta t \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}.$$

### 7.3.2. Process noise:

The process noise matrix $\mathbf{Q} \in \mathbb{R}^{6\times6}$ adds our uncertainty in the constant velocity model to the Kalman filter covariance. The continuous white noise is applied to the velocity terms,

$$\mathbf{Q}_c = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}\phi_s \tag{33}$$

where $\phi_s$ is the power spectral density. In practice, the power spectral density is an engineering knob — larger values result in smoother state predictions. The discretization of the noise through our process model [32] is given by

$$\mathbf{Q} = \int_0^{\Delta t} F(t)\mathbf{Q}_c F^\top(t)dt \tag{34}$$

We solve for $\mathbf{Q}$ using a symbolic solver, obtaining the matrix

$$\mathbf{Q} \approx \begin{bmatrix} 0 & 0 & 0 & \frac{(\Delta t)^2}{2} & 0 & 0 \\ 0 & 0 & 0 & 0 & \frac{(\Delta t)^2}{2} & 0 \\ 0 & 0 & 0 & 0 & 0 & \frac{(\Delta t)^2}{2} \\ \frac{(\Delta t)^2}{2} & 0 & 0 & \Delta t & 0 & 0 \\ 0 & \frac{(\Delta t)^2}{2} & 0 & 0 & \Delta t & 0 \\ 0 & 0 & \frac{(\Delta t)^2}{2} & 0 & 0 & \Delta t \end{bmatrix},$$

which is shown as an approximation because we have set small terms to zero.

### 7.3.3. Measurement Jacobian:

The event measurements relate to the attitude using the rotation and projection models of Eq. 12 and Eq. 13. We define

our measurement matrix $\mathbf{H} \in 2 \times 6$ for a single event as

$$\mathbf{H} = \begin{bmatrix} \frac{\mathrm{d}x}{\mathrm{d}\mathbf{q}^t} & 0_{1\times 3} \\ \frac{\mathrm{d}y}{\mathrm{d}\mathbf{q}^t} & 0_{1\times 3}, \end{bmatrix} \tag{35}$$

where $\frac{\mathrm{d}x}{\mathrm{d}\mathbf{q}^t} \in \mathbb{R}^{1\times 3}$, $\frac{\mathrm{d}y}{\mathrm{d}\mathbf{q}^t} \in \mathbb{R}^{1\times 3}$, noting that we do not incorporate velocity into our measurement model (i.e., $\frac{\mathrm{d}x}{\mathrm{d}\omega^t} = 0_{1\times 3}$ and $\frac{\mathrm{d}y}{\mathrm{d}\omega^t} = 0_{1\times 3}$). Considering the event $x$ and $y$ coordinates, the partial derivatives w.r.t. attitude expands as

$$\frac{\mathrm{d}x}{\mathrm{d}\mathbf{q}^t} = \frac{\mathrm{d}}{\mathrm{d}\mathbf{q}^t}\left[f\frac{X}{Z}\right] = \frac{\mathrm{d}}{\mathrm{d}\mathbf{q}^t}\left[f\frac{\mathbf{q}^t(\mathbf{s})_x}{\mathbf{q}^t(\mathbf{s})_z}\right] \tag{36}$$

and

$$\frac{\mathrm{d}y}{\mathrm{d}\mathbf{q}^t} = \frac{\mathrm{d}}{\mathrm{d}\mathbf{q}^t}\left[f\frac{Y}{Z}\right] = \frac{\mathrm{d}}{\mathrm{d}\mathbf{q}^t}\left[f\frac{\mathbf{q}^t(\mathbf{s})_y}{\mathbf{q}^t(\mathbf{s})_z}\right], \tag{37}$$

respectively. In these equations, the quaternion $\mathbf{q}(\cdot)$ rotates the star world coordinates to the camera, and the $(x, y, z)$ subscripts index that dimension of the rotated vector (e.g., $\mathbf{q}^t(\mathbf{s})_x = \mathbf{q}^t(\mathbf{s}) \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}$). For completeness, the rotation of a vector $\mathbf{s} \in \mathbb{R}$ by a quaternion $\mathbf{q} = [q_0, \breve{\mathbf{q}}] = [q_0, q_x, q_y, q_z]$ is defined as,

$$\mathbf{q}(\mathbf{s}) = (2q_0^2 - 1)\mathbf{s} + 2q_0\breve{\mathbf{q}}^\times\mathbf{s} + 2\breve{\mathbf{q}}(\breve{\mathbf{q}}^\top\mathbf{s}). \tag{38}$$

Using the product and chain rules, we compute the derivative of the event location $(x, y)$ with respect to attitude as

$$\frac{\mathrm{d}x}{\mathrm{d}\mathbf{q}^t} = f\left[-\frac{\mathrm{d}Z}{\mathrm{d}\mathbf{q}^t}XZ^{-2} + \frac{\mathrm{d}X}{\mathrm{d}\mathbf{q}^t}Z^{-1}\right] \tag{39}$$

and

$$\frac{\mathrm{d}y}{\mathrm{d}\mathbf{q}^t} = f\left[-\frac{\mathrm{d}Z}{\mathrm{d}\mathbf{q}^t}YZ^{-2} + \frac{\mathrm{d}Y}{\mathrm{d}\mathbf{q}^t}Z^{-1}\right] \tag{40}$$

where the Jacobians of the rotations are given by

$$\frac{\mathrm{d}X}{\mathrm{d}\mathbf{q}^t} = -(\mathbf{q}(\mathbf{s}))_x^\times, \tag{41}$$

$$\frac{\mathrm{d}Y}{\mathrm{d}\mathbf{q}^t} = -(\mathbf{q}(\mathbf{s}))_y^\times, \tag{42}$$

$$\tag{43}$$

and

$$\frac{\mathrm{d}Z}{\mathrm{d}\mathbf{q}^t} = -(\mathbf{q}^t(\mathbf{s}))_z^\times. \tag{44}$$

**Algorithm 2** EBS-EKF Star Tracking

**Input:** Positive event stream $\mathcal{E} = \{\mathbf{x}, t\}_{i=0:N}$,
**Output:** Camera State $\mathbb{S}$ (3D rotation and angular velocity)
**Initialize:** $\mathbb{S}_0 \leftarrow$ astrometry with binned positive events
r $\leftarrow$ max search radius
**for** each event $e_i = (\mathbf{x}_i, t_i)$ **do**
  $\hat{\mathbb{S}}_i \leftarrow$ EKF predict at $t_i$
  $\mathbf{x}_s \leftarrow$ closest projected catalog star to $\mathbf{x}_i$
  **if** distance$(\mathbf{x}_i, \mathbf{x}_s) \leq$ r **then**
    $\bar{\mathbf{v}}_s \leftarrow$ star's normalized linear velocity via $\hat{\mathbb{S}}_i$
    $m_s \leftarrow$ star's apparent magnitude
    $\hat{\mathbf{x}}_i \leftarrow \mathbf{x}_i + \bar{\mathbf{v}}_s \cdot z(m_s)$ # apply offset correction
    $\mathbb{S}_i \leftarrow$ EKF update with $\hat{\mathbf{x}}_i$
  **end if**
**end for**

## 8. Tracking Algorithm Details

We now provide more details for our tracking algorithm, which is summarized in the main text in Alg. 1. We restate the algorithm here, and provide a more detailed description of its operation.

Referring to Alg. 2, we initialize the camera attitude by binning events with a small time window (e.g., 30 ms), identifying event clusters with DBSCAN clustering [11]. Specifically, we accumulate positive events in batches of 60 milliseconds, and use *scikit-learn*'s implementation of DBSCAN [24] with parameters epsilon=2 and min_samples=3. We pass the centroids of identified clusters to astrometry [22], which attempts to compute a camera attitude. We run astrometry with log odds of 14.

After obtaining an initial attitude with astrometry, we begin processing events using our EKF algorithm. First, we use the event's timestamp to update the EKF prediction using Eqs. 16, 17, 18. We then check if the event is within a set pixel radius of a catalog star, given the current attitude. This requires projecting catalog stars onto the imaging plane using the camera state. To do so, we use Eq. 12 to rotate the catalog with the current state, and Eq. 13 to project catalog stars into pixel space. If an event has a nearby catalog star, we use the star's magnitude and linear velocity to apply our intensity-dependent offset correction. We compute the linear velocity using the image Jacobian,

$$\begin{bmatrix} v_x \\ v_y \end{bmatrix} = \begin{bmatrix} -xy/f & f + x^2/f & y \\ -f - y^2/f & xy/f & -x \end{bmatrix} \begin{bmatrix} \omega_x \\ \omega_y \\ \omega_z \end{bmatrix} \tag{45}$$

where $\omega$ is known, since it is part of our EKF state. After applying the offset correction to the event, we use it as a measurement to update the EKF using Eq. 22 and 23.

## 9. Monitor Centroiding Details

**Monitor Details:** The 500 Hz monitor used is the ASUS ROG Swift Pro PG248QP. An important quality of this monitor is that there is minimal pixel value overshoot in-between frames (explained in [1]). This lends frame transitions that are similar to real analog scenes. We did explore using OLED monitors but noticed that the event camera picked up on its refresh rate pattern.

**Simulating Stars on Monitor:** To simulate a moving star on the monitor, we display a Gaussian distribution as shown in Eq. 3 in the paper. We use a 2 pixel standard deviation $\sigma_s$ (2.5 pixels on the camera's sensor) with a constant, unchanging speed across the monitor ($v_x = 35$ sensor pixels per second, $v_y = 0$). The pixel intensity values are also gamma-corrected so that it appears truly Gaussian to the camera sensor. To simulate different intensities stars, we use a combination of lowering the peak pixel intensity (e.g. 255 to 128 to 64) and by placing neutral-density (ND) filters in front of the camera which reduce the gathered light. As a result, each subsequently displayed star in the experiment is half as bright as the previously displayed star. We then have to calibrate the relative magnitude $m_s$ of just one of the displayed stars (found by matching the observed event rate to what we found in our night sky dataset) and the rest can be easily calculated. Examples of the monitor star event profiles are depicted in Fig. 9.

**Pixel Synchronization:** The event camera was placed about 8 feet away from the monitor such that one sensor pixel corresponded to approximately one monitor pixel. To calculate the homography matrix between the sensor and the monitor, we flashed a series of eleven dots on the screen and binned the positive events with 10ms time bins. Knowing the position of the dots on the monitor and calculating the position of the dots on the sensor with the binned frames, we could recover a transformation matrix which relates the two coordinate frames. For this matrix, our mean reprojection error was approximately 0.5 pixels.

**Time Synchronization:** In order to compare events to ground truth star locations in the monitor, we need to align the monitor and event time scales. To do so, we flash a set of pixels on the monitor at the start of an experiment — the events generated from these pixels correspond with the start of the experiment. However, once we place ND filters over the camera, the events from these trigger pixels become delayed due to the low-pass filter effects discussed in the main paper and in [26]. If unaccounted for, this would cause a systematic bias in our low light experiments. Thus, we perform a calibration procedure where we place ND filters over sections of the monitor, and then illuminate all sections simultaneously, allowing us to characterize and correct for the relative differences in event timing and ensure proper synchronization between the camera and monitor. The measured event rates for the trigger pulse under different lighting conditions are shown in Fig. 10.

**Camera/Monitor Synchronization Discrepancies:** Even after accounting for the time and pixel synchronizations discussed in previous sections, we still observed a systematic timing bias in the monitor experiments, where the positive events excessively lagged behind the true star centroid as depicted in the third row and first column of Fig. 9. This error can be corrected by introducing a timing bias into the synchronization. Importantly, *this timing bias is constant across all of the experiments*. We determine this constant timing offset by what minimizes the average error for the maximum likelihood method (blue curve in Fig. 4 in the paper). Since maximum likelihood should provide the most accurate results, this approach ensures an optimal correction. While we acknowledge that this introduces some scientific bias, the primary concern in Fig.4 is the *slope* (or standard deviation) of the centroiding error with respect to star magnitude. This is because centroiding inconsistency is the main source of error in the centroiding algorithms. The calculated timing offset does *not* affect this slope, and both proposed methods—maximum likelihood estimation and the Gaussian approximation—exhibit significantly smaller slopes and standard deviations than alternatives. Finally, it is important to note that these method parameters were fitted to night sky measurements, not to the monitor data. This ensures that the parameters are not overfit to this specific monitor data.

**Offset Curve:** For the monitor centroiding experiments, we use the same offset curve as our real night sky data to ensure we are not overfitting to either modality. As shown in Fig. 4 of the paper, the offset curve significantly reduces centroiding error on the monitor, despite being calibrated on the night sky dataset.

## 10. Real Night Sky Dataset Details

**DVS Hardware:** Our event camera is one of the latest from Prophesee, the EVK4-HD. The EVK4-HD consists of a 1280x720 event-pixel array (pixels are 4.86um in size) and related readout circuitry which is accessible over USB. The system is set up with a 35mm lens focused at infinity with a field of view of 5.7°x 10.2°. This leads to an effective instantaneous field of view for each pixel of approximately 30 arcseconds.

**APS Hardware:** We utilize a Rocket Lab ST-16RT2 star tracker as our reference attitude-determination solution. The ST-16RT2 consists of a 2592 x 1944 pixel CMOS active-pixel sensor, a custom 16mm f/1.6 lens, a 12 cm rigid sun-keepout baffle, and an onboard processor capable of providing attitude solutions at 2Hz over an RS-485 link which we access via a USB to RS-485 converter. The manufacturer states the tracker can achieve accuracies of 5 arcseconds RMS cross-boresight and 55 arcseconds RMS around-boresight, with a maximum slew rate of 3°/second.

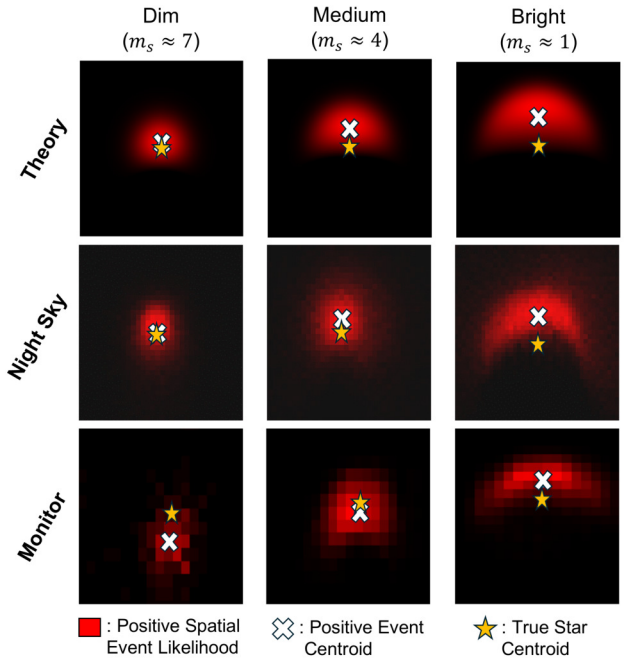


Figure 9. Example spatial event likelihoods from theory, night sky, and monitor measurements. This is similar to results shown in Fig 3 of the main paper. Note that the positive event centroid (white "x") lags behind the true star centroid more in the monitor experiments (3rd row) due to minor discrepancies in monitor/camera synchronization. However, the trend of the positive event centroid offset increasing with star magnitude is consistent across all modalities.

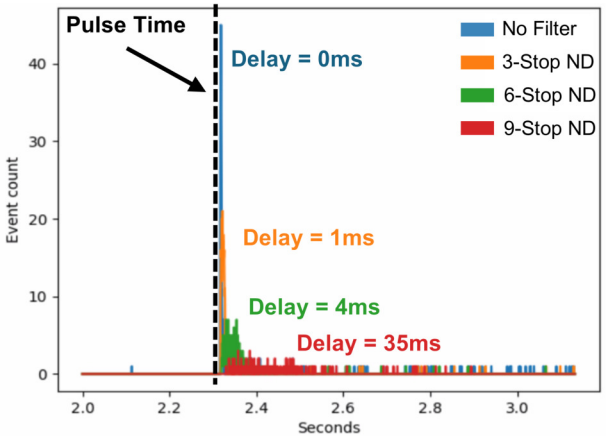


Figure 10. Synchronization timings of monitor experiments. Each neutral-density (ND) filter stop decreases light by 87.5%.

The system is set up for a field of view of 15°x 20°.

**Sensor Stage Motion:** We utilize two Edelkrone HeadONE units set up in a pan/tilt configuration to sweep the night sky with both the EVK4-HD and the ST-16RT2, which are rigidly mounted to each other. These are controlled wirelessly via an Edelkrone Motor Controller dongle. For each experiment, after a short static calibration period the pan/tilt mechanism is used to rotate both sensors at various speeds to simulate satellite rotation.

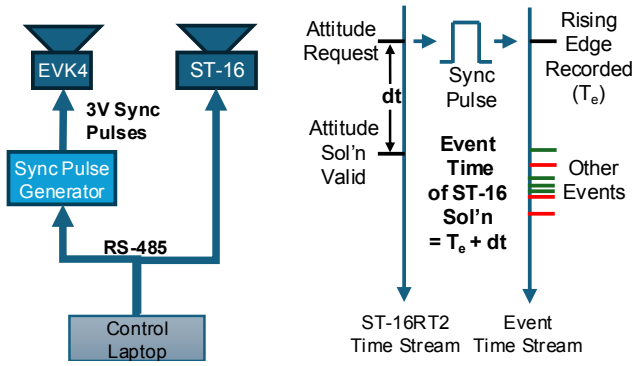


Figure 11. Left: Block diagram of synchronization hardware architecture. Right: Visualized timelines for the commercial star tracker (left) and event camera (right). Attitude requests come from the control laptop, which triggers the Sync Pulse Generator to send a synchronization pulse which the event camera records in its own timestream alongside positive and negative events. The commercial star tracker solution includes the validity time offset $dt$, measured from when the solution request was received. By combining the event-time of the sync pulse and the validity offset, the timestamp of the commercial star tracker's solution relative to the event stream can be recovered.

**DVS/APS Time Synchronization:** We synchronize our event camera with the Rocket Lab star tracker to align our orientation estimates temporally as described in Fig. 11. To do so, we set up a small microcontroller (an Adafruit Trinket M0) which listens to the RS-485 serial line used to communicate with the Rocket Lab tracker. When a solution is requested by our control computer via this serial line, the microcontroller sends a square wave pulse to the EVK4 which records the pulse in the event stream with the appropriate timestamp. Rocket Lab attitude solutions include a validity timestamp relative to the time of the initial solution request/square wave pulse which has been recorded in the EVK4 event stream. This combination of recorded data allows us to identify the precise time when the Rocket Lab-estimated attitude is valid in the event data timeline.

**Relative Attitude Calibration:** To accurately compare attitude solutions between the Rocket Lab APS tracker and the DVS camera, we need to determine the relative rotation between them. Assuming the cameras share a rigid body (i.e., are mounted to each other), the relative rotation remains constant regardless of the cameras' orientations. Therefore, we generally evaluate performance based on the deviation of the DVS tracker from this relative rotation. While stereo calibration using pixel values would be ideal, the Rocket Lab tracker provides quaternion solu-

tions that are precisely calibrated to its imaging plane and undergo proprietary signal processing corrections inaccessible to the user. Consequently, we must rely directly on the quaternion solutions from the Rocket Lab tracker.

To obtain the relative rotation, we perform velocity sweeps of the night sky with both cameras, as shown in the velocity sweep experiments in Fig. 6. We then run our tracker over these sweeps and identify the rotation that minimizes the error between the Rocket Labs solutions and our tracker's solutions. Mathematically, we determine the relative rotation $q_r$ that minimizes the rotation error between the two trackers:

$$\min_{\mathbf{q}_r} \sum_i ||\mathbf{q}^i_{\text{RL}} \cdot \mathbf{q}_r \boxminus \mathbf{q}^i_{\text{DVS}}|| \tag{46}$$

Here, the $\boxminus$ sign represents the boxminus operator for subtracting rotations, as described in Hertzberg et al. [18]. Importantly, we observe that the choice of tracking algorithm has minimal impact on the estimated relative rotation, as any bias (i.e., centroiding error) is removed by performing the velocity sweeps in opposite directions. We measured the consistency of our relative rotation across three velocity sweeps to be within 20 arcseconds. We use the same relative rotation on all tracks and separate the data used for rotation calibration from test data to ensure there is no data leakage.

**Offset Curve Calibration:** Knowledge of the cameras' relative rotation enables us to register events with the "ground truth" star locations predicted by the APS tracker. We measure the frequency of events occurring near star locations, generating the histograms like those shown in the middle row of Fig. 9. These histograms allow us to measure the offset of the event mean to the star's true location. The measured offset for each star magnitude results in our offset calibration curve. In our experiments, we used a velocity sweep pattern (which is *not* used for evaluation in the main paper) to obtain the data for creation of the calibration curve.

**Computing the Theoretical Offset Curve:** To calculate the theoretical offset curve $z(m_s)$ from Eq. 7 in the paper, we first need to find the dark current value $I_0$ in paper Eq. 2 and the cutoff frequency $f_c$ values $a, b$ in paper Eq. 5. With the relative attitude calibration performed in the previous section, we can find the observed offset between the APS tracker (i.e. ground truth) and the centroids of our EBS tracker for each given star magnitude in the field of view. Given these offsets and also knowledge of the star size ($\sigma_s = 2$ pixels) and speed (50 pix/s), we have an empirical offset curve. Substituting these values into paper Eqs. 2, 3, 5, and 6, we then can optimize the values of $I_0, a, b$ such that the theoretical offset curve matches closely to the empirical curve, which lends the values $I_0 = 1$ (normalized so that a star with $m_s = 7$ has peak intensity of 1), $a = 20$ Hz/intensity, and $b = 2$ Hz. This curve is shown in Fig. 3c in the paper.

It is important to note three things about this calibration. The first is that we actually optimize the *normalized* offset curves, where the minimum offset value $z(m_s)$ is set to 0. This is to remove systematic calibration offsets between our different modalities, and therefore we are matching the *slope* of the $z(m_s)$ curve with respect to $m_s$, which is more important for accuracy calibration. The second is that we also test these same values of $I_0, a, b$ in the monitor experiments in Section 4 in the paper and show near-perfect agreement. This suggests that we are not overfitting to the night-sky results but that these values do encompass the sensor characteristics. Finally, the offset curve theoretically does change slightly as a function of star speed $\mathbf{v}$. However, empirically in our night sky dataset we see that it hardly changes the offset, perhaps due to small synchronization issues or calibration. Therefore we just use the same offset curve for different star speeds. Future work should explore this relationship.

## 11. Night Sky Dataset Discussion

In comparison to other works (see Table 3 for a summary of key dataset features), our dataset is unique in being collected from stars visible in an actual night sky. Other datasets were collected using a monitor to simulate stars. While simulating stars allows for reconstructing ground truth of the induced 'orientation' of the event camera star tracker, this type of data has downsides:

- Monitor pixels are not perfectly uniform and may introduce unrealistic fixed pattern noise into event data collected from a monitor.
- The dynamic range of a monitor is limited and quantized to discrete pixel values. A monitor is therefore unable to represent any portions of a star's light which are dimmer than its smallest 'on' value (i.e. 1 if the range of available values is 0-255) or brighter than its highest 'on' value. A monitor star is also limited to a set of discrete brightness levels, while stars in the night sky can produce a continuous set of brightnesses.
- Pixels in a monitor have a finite update rate and a quantized location (a monitor star cannot be located between pixels), introducing unrealistic event patterns into monitor data and sub-pixel inaccuracies in star location. The night sky has an effectively infinite update rate, as we slew the sensors over unchanging light sources (stars).

Tables 2 (of the main text) and 3 summarize and compare to other works the dataset we test on and will release for further study. Most of our experiments are approximately 2.5 to 5 minutes long, ensuring we have a long enough time period to uncover possible drift in various trackers. The total amount of data we release is significantly longer than

| Dataset | Camera | Resolution | FOV | Max Slew Rate | Total Length | Stars |
|---|---|---|---|---|---|---|
| Chin/ Bagchi | Davis 240C | 240 x180 | 20 deg | 4 deg/s | 4.5 min | LCD |
| Latif | Proph. EVK3 | 1280 x720 | 1.5 deg | 0.005 deg/s | 70 sec | LCD |
| **Ours** | **Proph. EVK4** | **1280 x720** | **10 deg** | **7.5 deg/s** | **19.5 min** | **Night Sky** |

Table 3. EBS star tracking dataset comparisons. Our dataset contains more data and faster slew rates than prior works, in addition to be collected from an actual night sky instead of simulated stars on a monitor.

prior datasets.

**Drift Phenomenon:** In some tracks, we observed that the mean difference between the trackers changed as they moved across the sky, with the change becoming more extreme near the horizon. An example of this "drift" in difference can be observed in the across plot of Fig. 26, where the difference gradually increases and decreases throughout the track. One potential explanation is a differential flexure of the cameras, which could change their relative rotation as they point closer to the horizon. In particular, the Rocket Lab tracker has a large sun keep-out baffle on it, which may lead it to be more deflected by gravity than the much smaller event camera when the optical axis is more perpendicular to the gravity vector. Another potential culprit is atmospheric diffraction towards the horizon impacting the accuracy of the trackers [20]. Since the two sensor (APS and EBS) are not perfectly optically aligned, they may experience different attitude errors due to atmospheric refraction. In one experiment, we collected data in a small range of angles near the zenith, and observed that the relative error between the APS and EBS attitude remained relatively constant (i.e., we did not observe the drift phenomenon). Since tracks measured closer to zenith (further from the horizon) are less susceptible to the drift effect, they have a 10-20 second lower arcsecond difference on average.

## 12. Additional Results

**Additional Plots:** We provide difference plots (Figs. 12 - 24) for all tracks shown in Table 2 of the main text, including plots comparing performance with and without our offset correction (Figs. 25 - 36). Referring to the difference plots, our method is typically an order-of-magnitude more accurate than the existing methods. Referring to the offset correction plots, we observe that using our offset typically reduces the error by 30 arcseconds, particularly in the roll direction, which is highly sensitive to centroiding errors [27]. We observe that the offset can drastically improve performance when a bright star is in the FOV. For instance, Fig. 34 shows results for a track that passed over Vega, an exceptionally bright star with an apparent magnitude of .03. In this case, our offset correction improves the mean average difference by over 180 arcseconds.

**Video Results:** We include three video results in the supplemental folder. In *high-velocity-track.mp4*, we display event frames for the high velocity track shown in Fig. 1 of the main text. White circles circle events that were used to update the tracker's attitude (i.e., white circles indicate stars). Note that while we display events as frames (using 30 ms batch times), the algorithm operated asynchronously at 1 KHz; the events are shown as frames for visualization purposes. The video *high_vel.mp4* shows the camera frustum for the high velocity track. We show the reconstructed track by our algorithm in red and the APS track in purple. The APS track failed to provide solutions in the fast regions of the track, resulting in an incorrect track interpolation. Our method provides an accurate reconstruction of the track. We also include a visualization of the frustum for the velocity sweep track in *vel_sweep_1.mp4*.

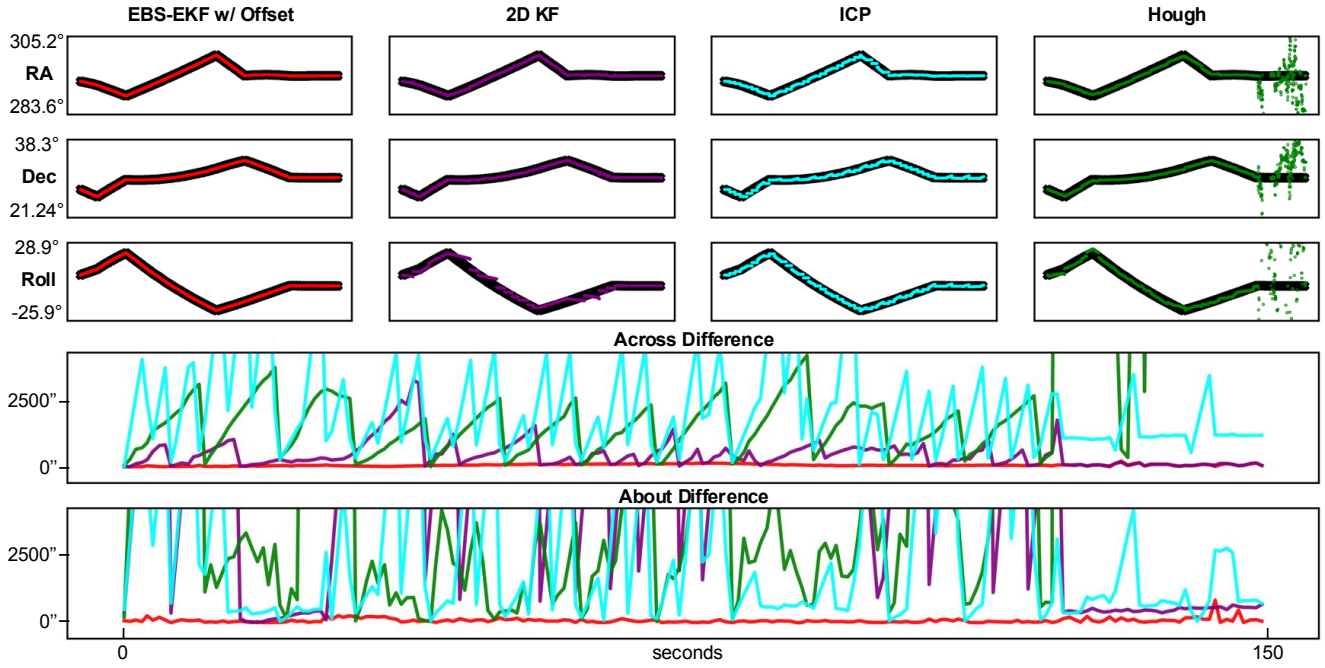


Figure 12. **Multipose 1**. RL samples are shown in black and each method is shown in a unique color.

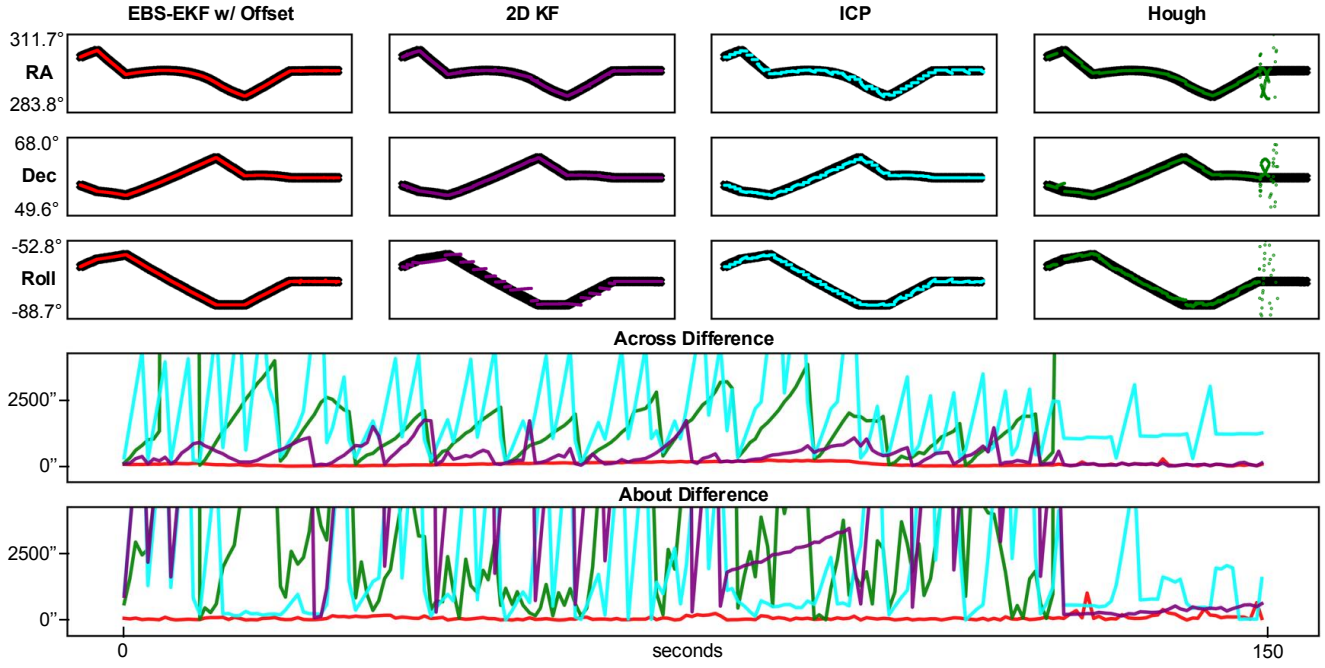


Figure 13. **Multipose 2**. RL samples are shown in black and each method is shown in a unique color.

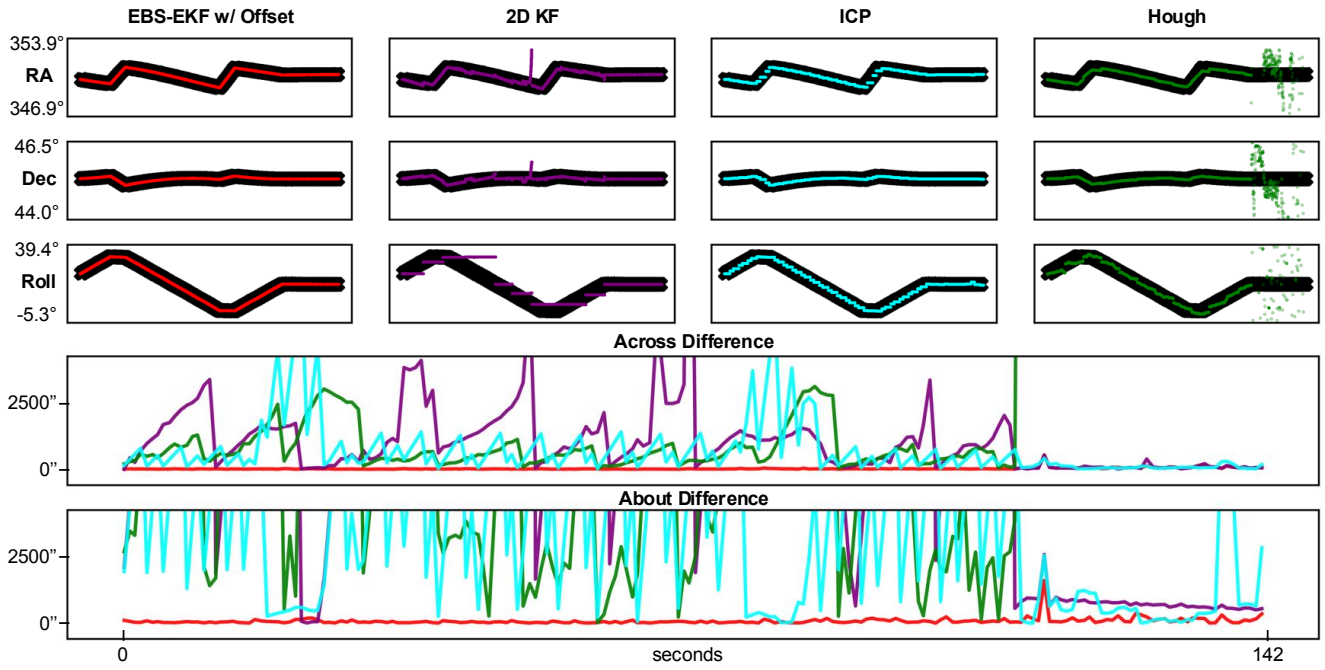


Figure 14. **Multipose 3**. RL samples are shown in black and each method is shown in a unique color.

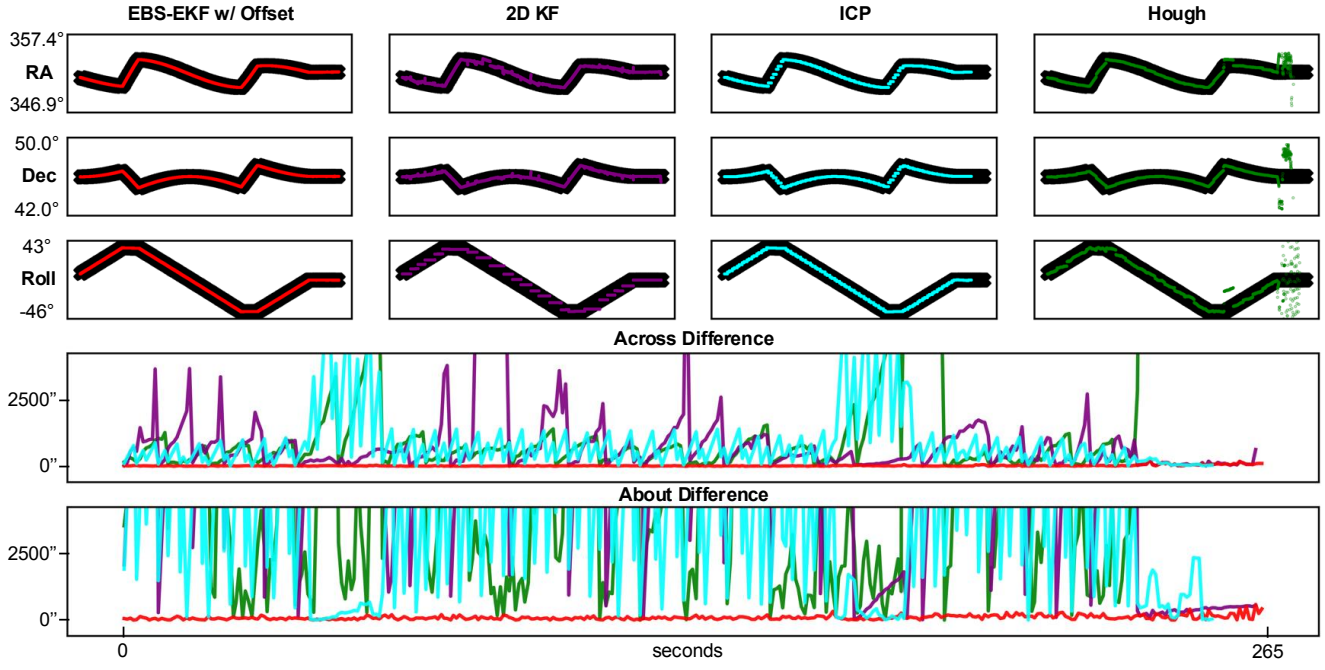


Figure 15. **Multipose 4**. RL samples are shown in black and each method is shown in a unique color.

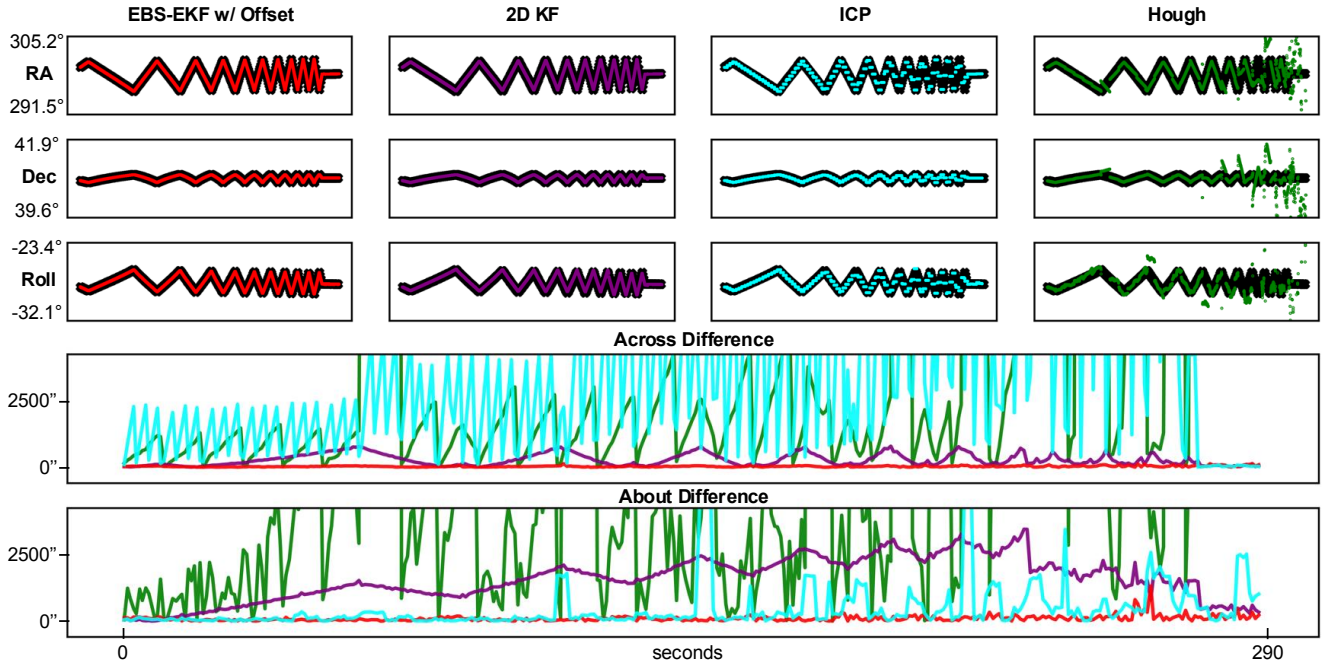


Figure 16. **Velocity Sweep 1**. RL samples are shown in black and each method is shown in a unique color.

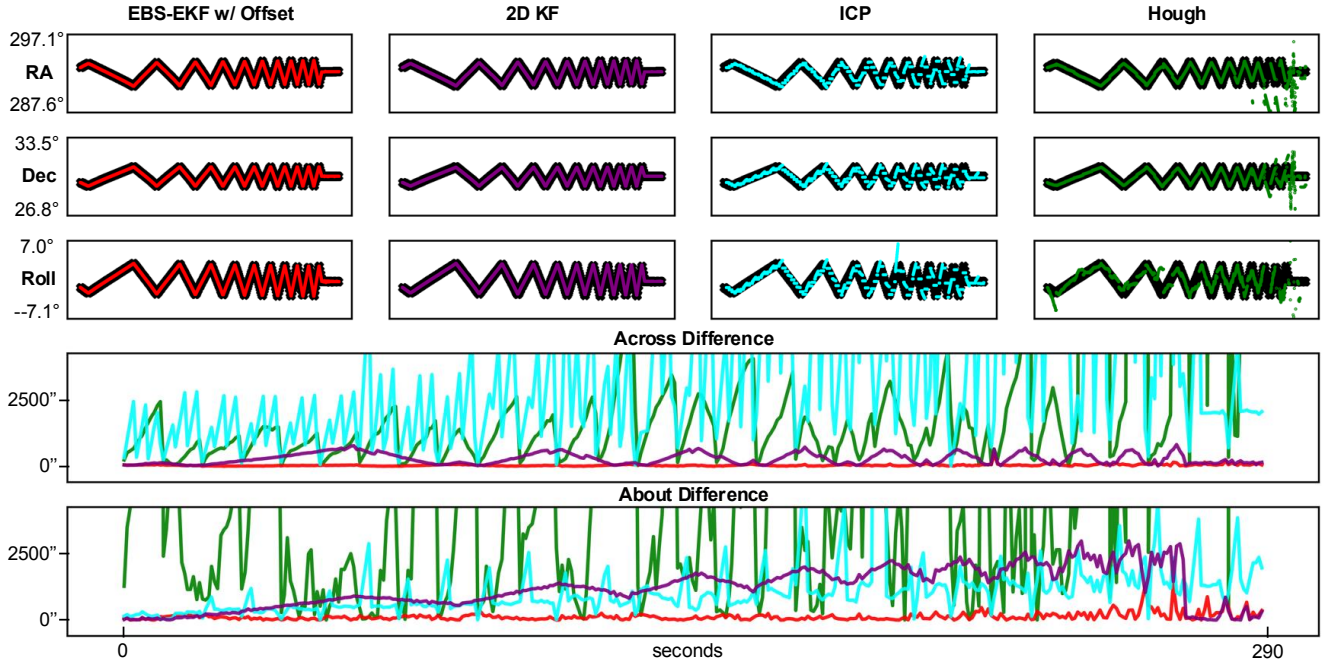


Figure 17. **Velocity Sweep 2**. RL samples are shown in black and each method is shown in a unique color.

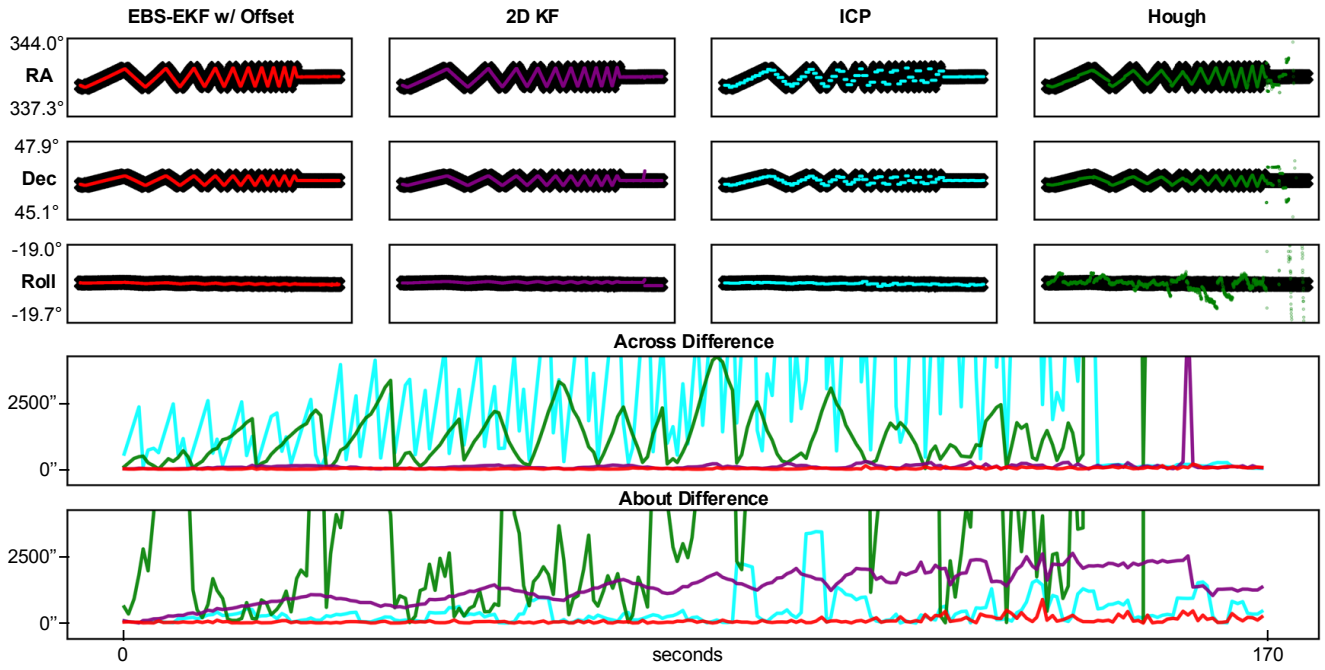


Figure 18. **Velocity Sweep 3**. RL samples are shown in black and each method is shown in a unique color.

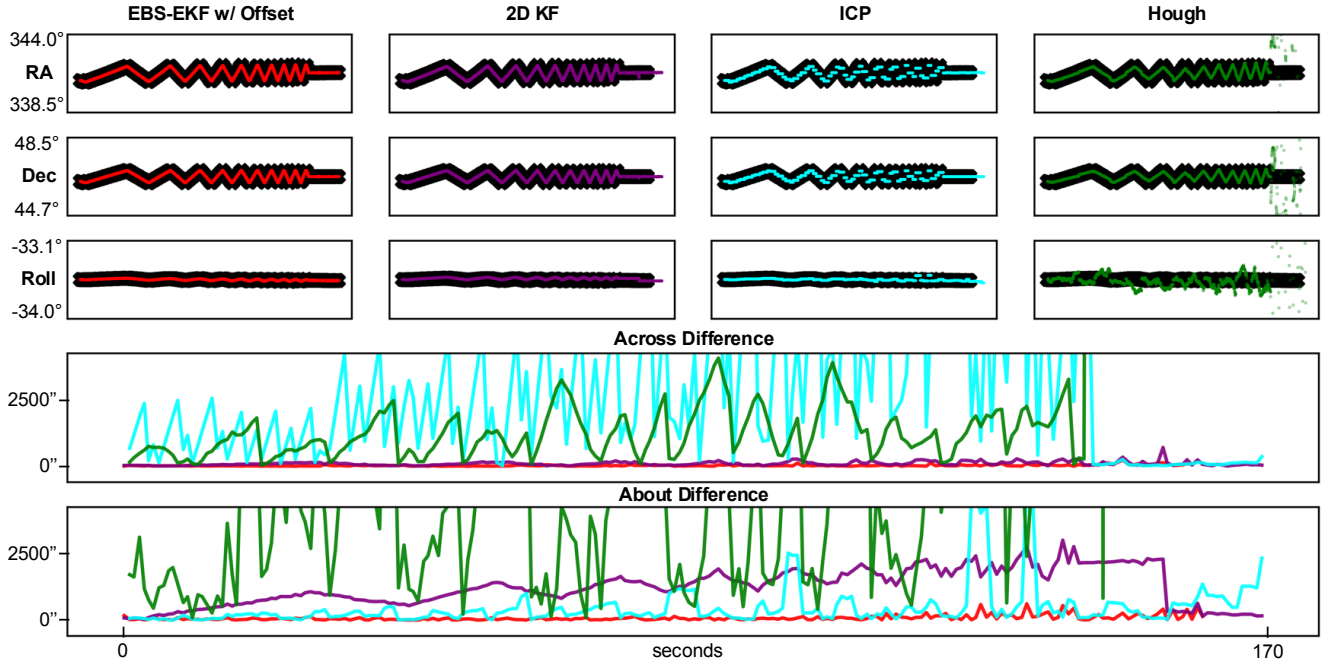


Figure 19. **Velocity Sweep 4**. RL samples are shown in black and each method is shown in a unique color.

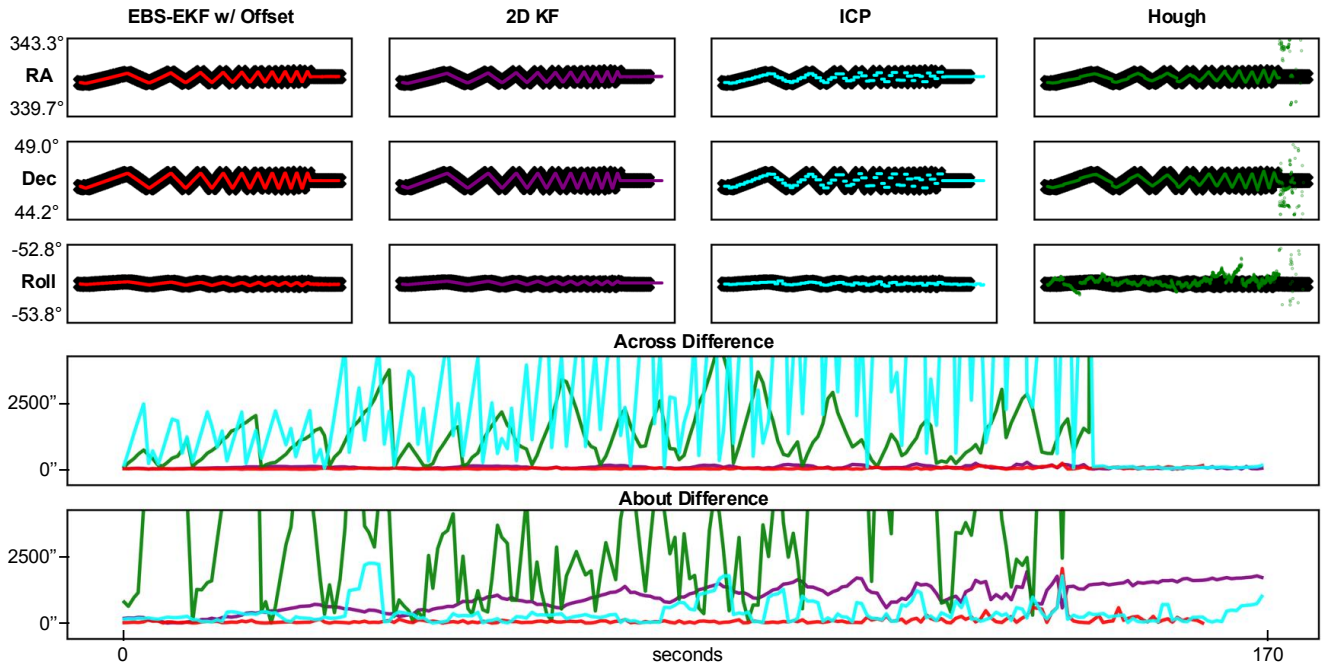


Figure 20. **Velocity Sweep 5**. RL samples are shown in black and each method is shown in a unique color.

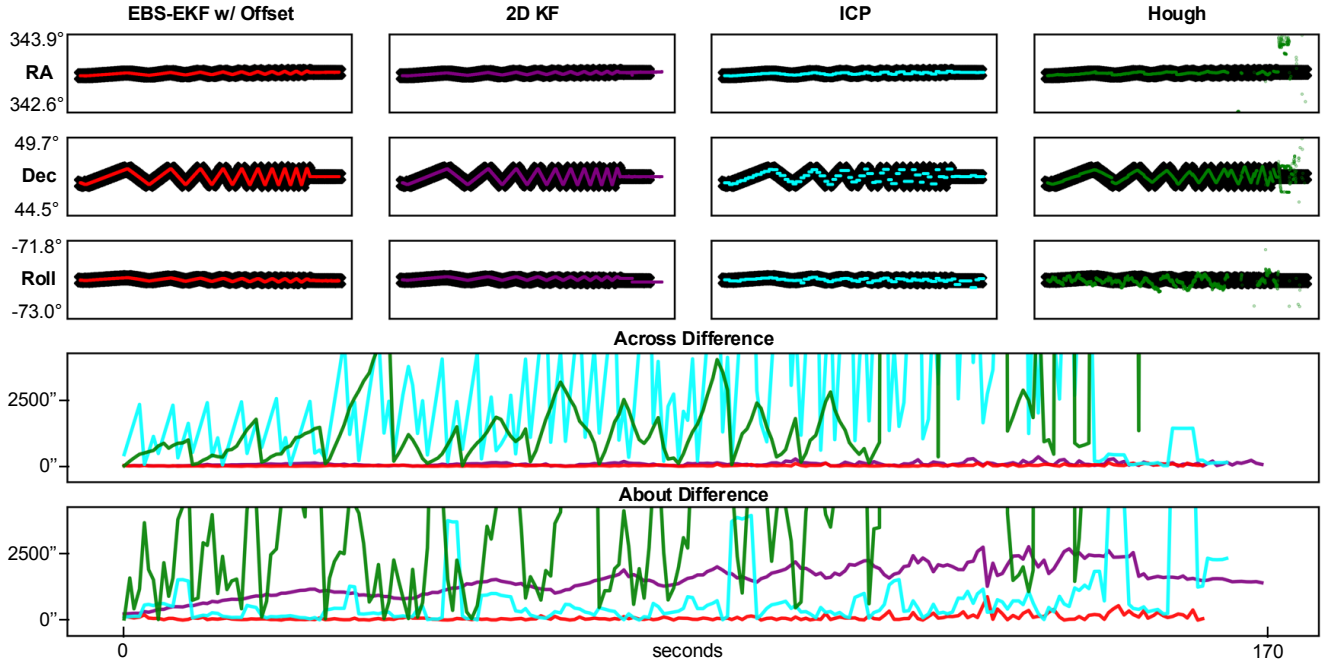


Figure 21. **Velocity Sweep 6**. RL samples are shown in black and each method is shown in a unique color.

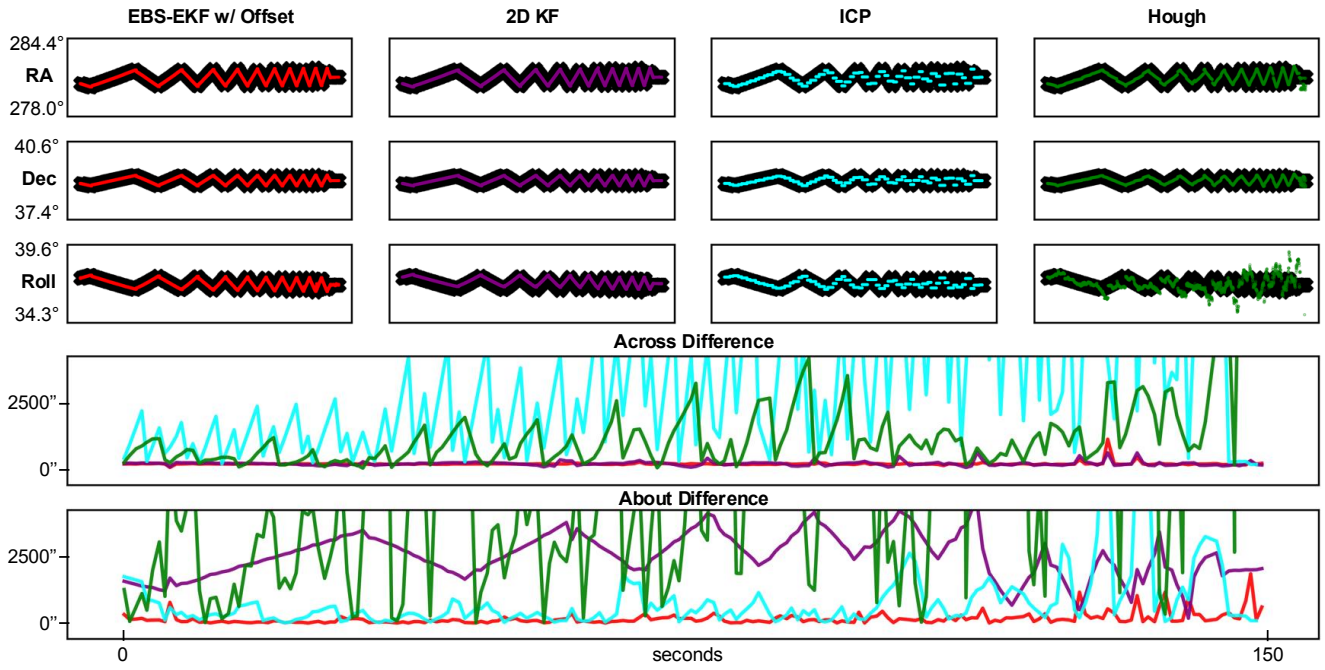


Figure 22. **Velocity Sweep 7**. RL samples are shown in black and each method is shown in a unique color.

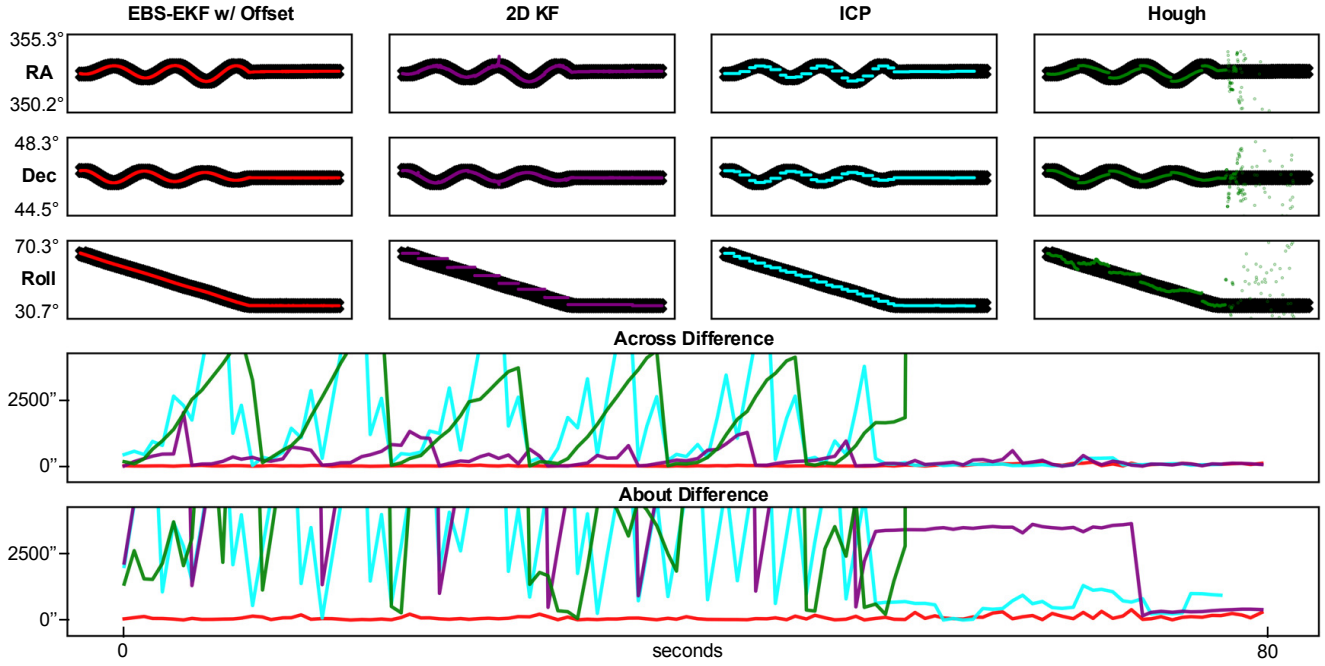


Figure 23. **Smooth Sine**. RL samples are shown in black and each method is shown in a unique color.

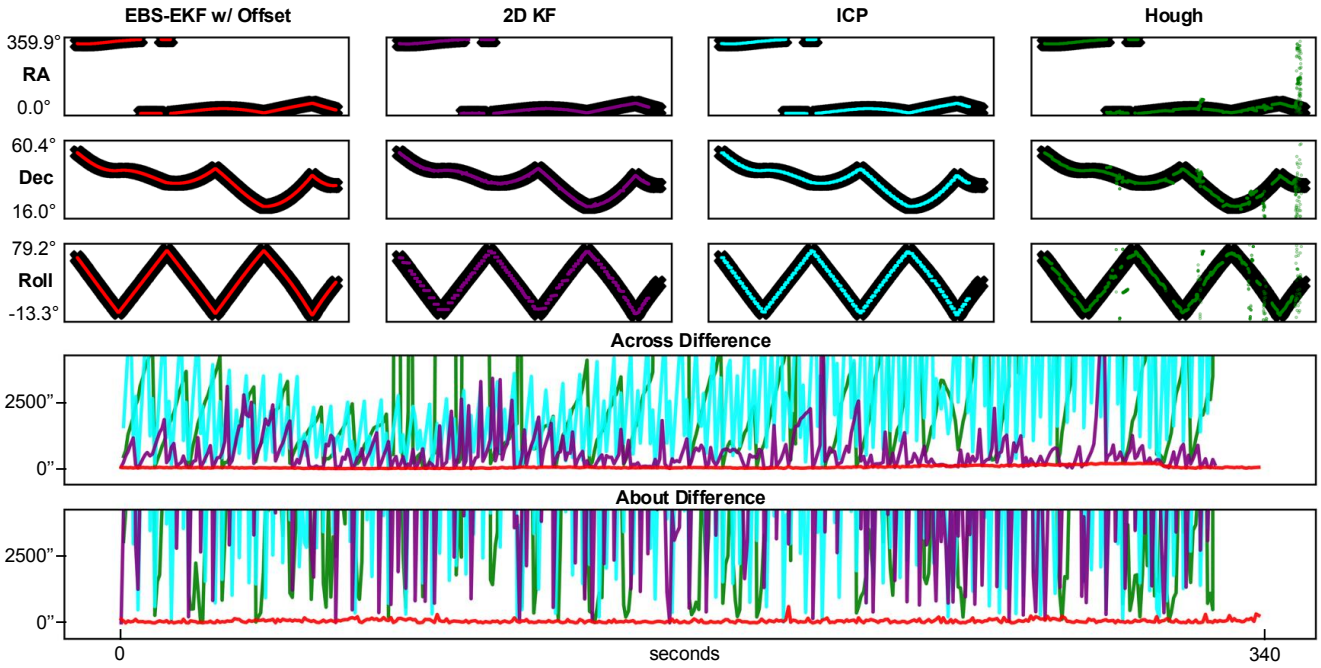


Figure 24. **Tilt Ladder**. RL samples are shown in black and each method is shown in a unique color.

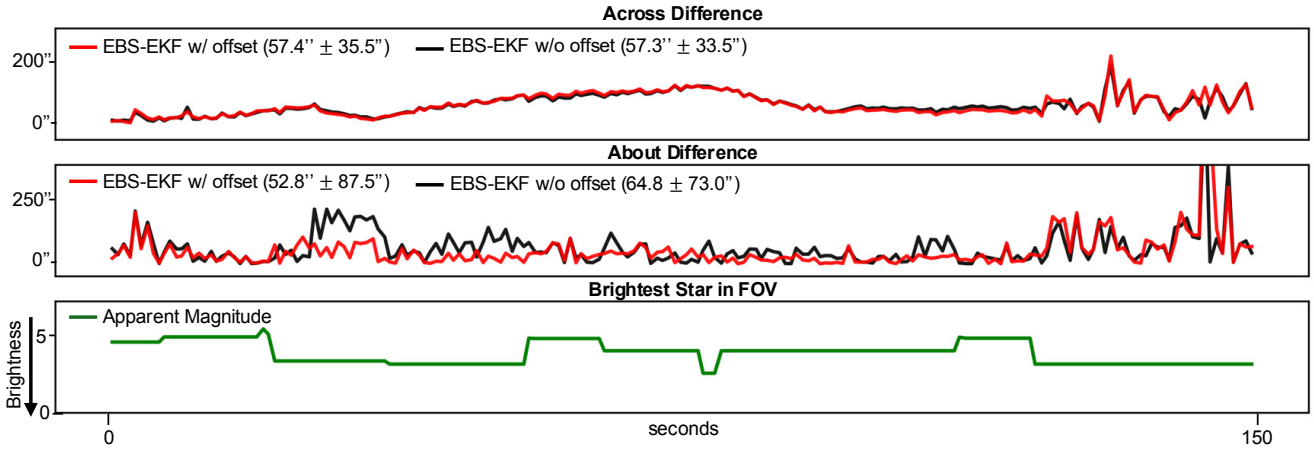


Figure 25. **Multipose 1 EBS-EKF w/ offset vs EBS-EKF w/o Offset**

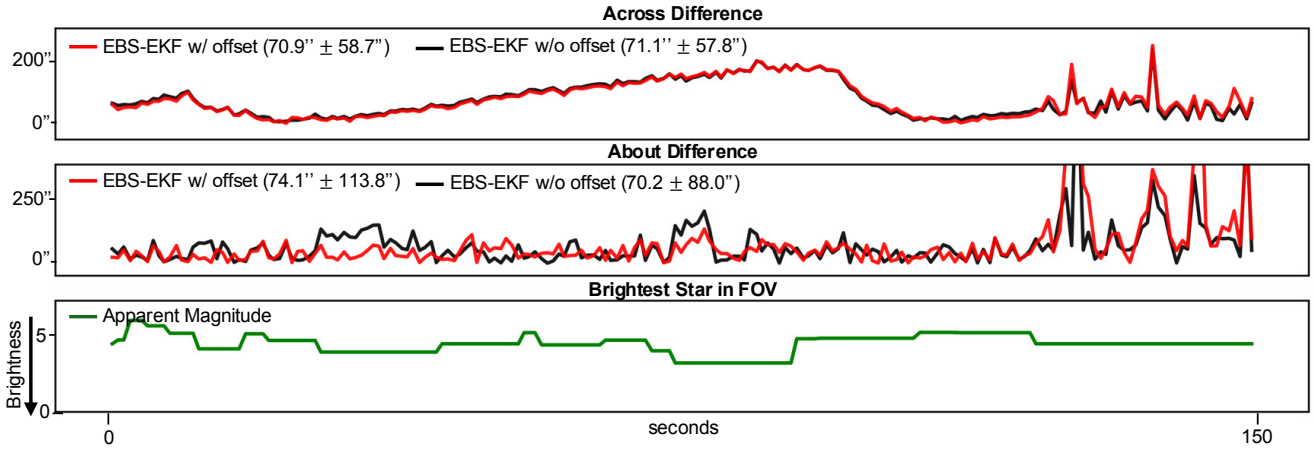


Figure 26. **Multipose 2 EBS-EKF w/ offset vs EBS-EKF w/o Offset**

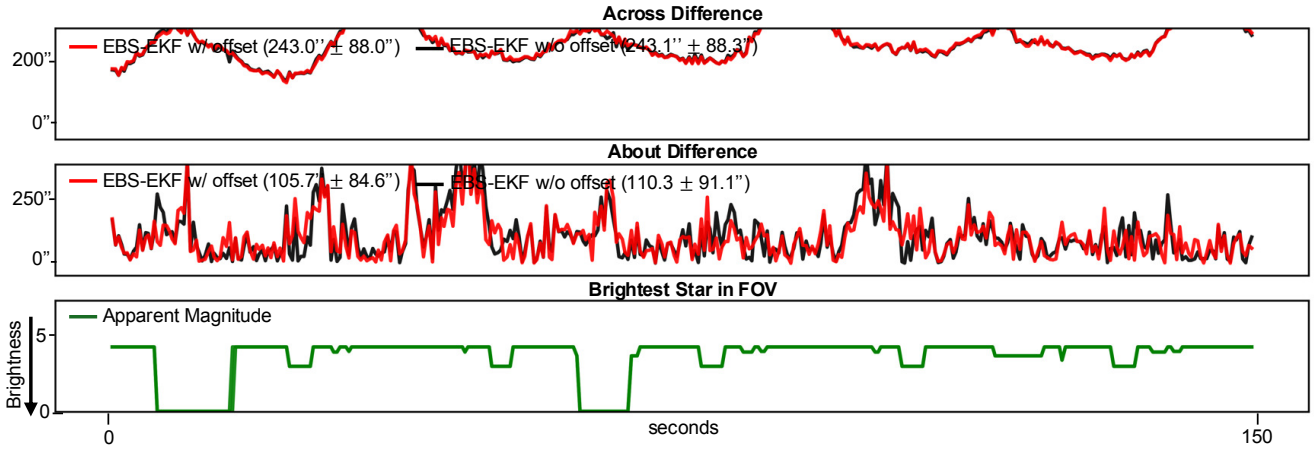


Figure 27. **Multipose 3 EBS-EKF w/ offset vs EBS-EKF w/o Offset**

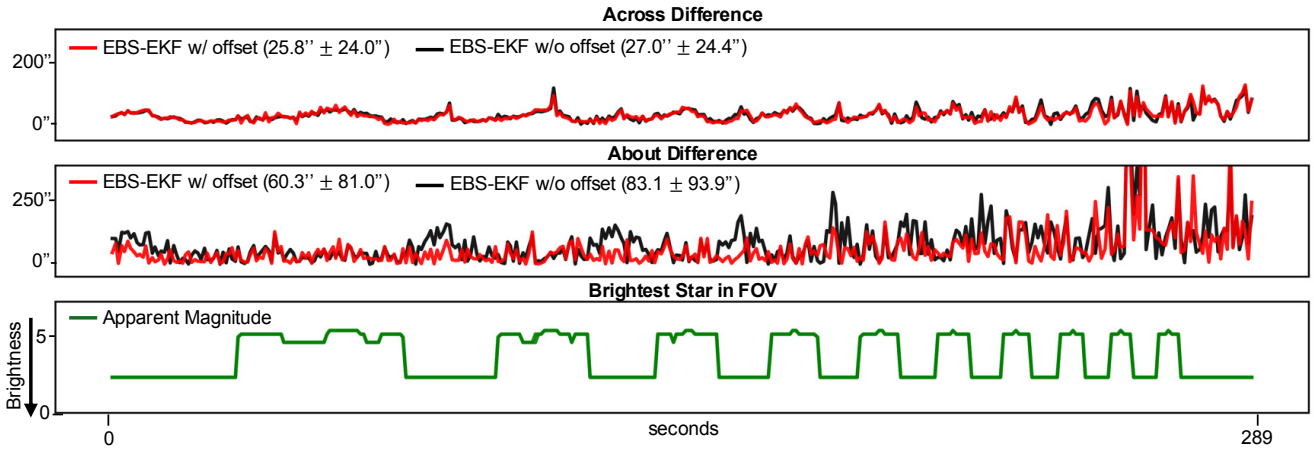


Figure 28. **Velocity Sweep 1 EBS-EKF w/ offset vs EBS-EKF w/o Offset**

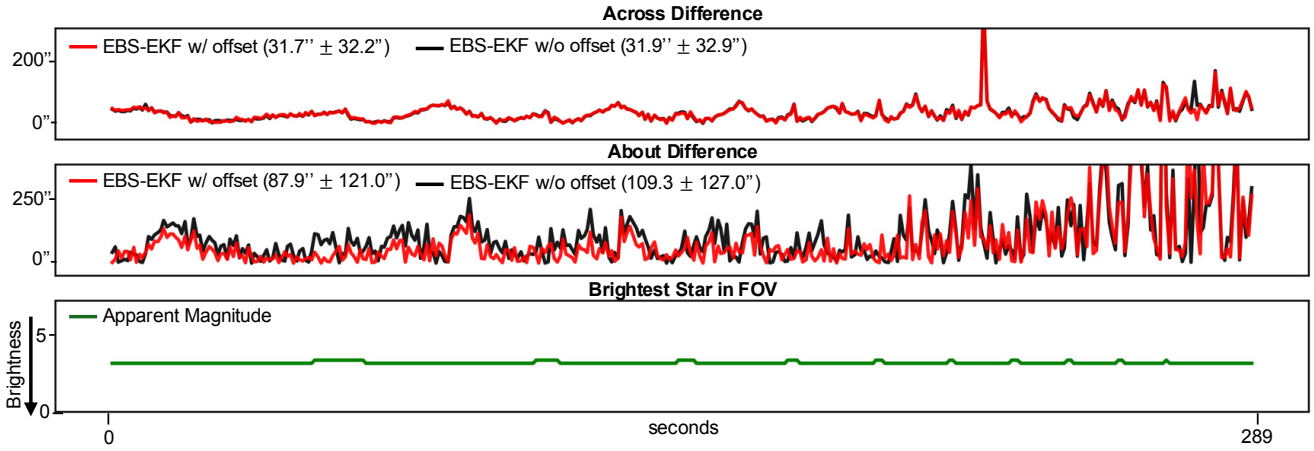


Figure 29. **Velocity Sweep 2 EBS-EKF w/ offset vs EBS-EKF w/o Offset**

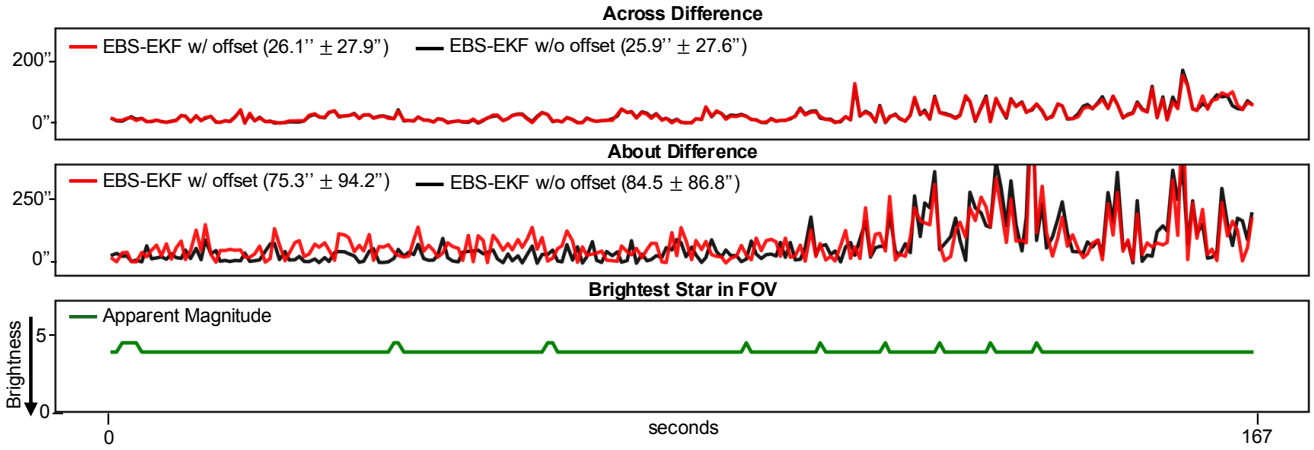


Figure 30. **Velocity Sweep 3 EBS-EKF w/ offset vs EBS-EKF w/o Offset**

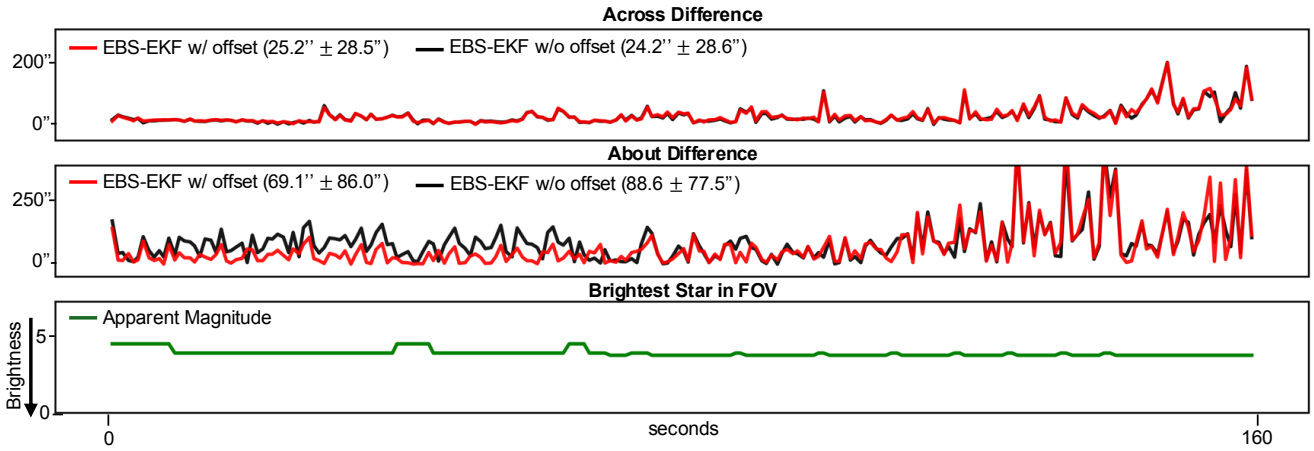


Figure 31. **Velocity Sweep 4 EBS-EKF w/ offset vs EBS-EKF w/o Offset**

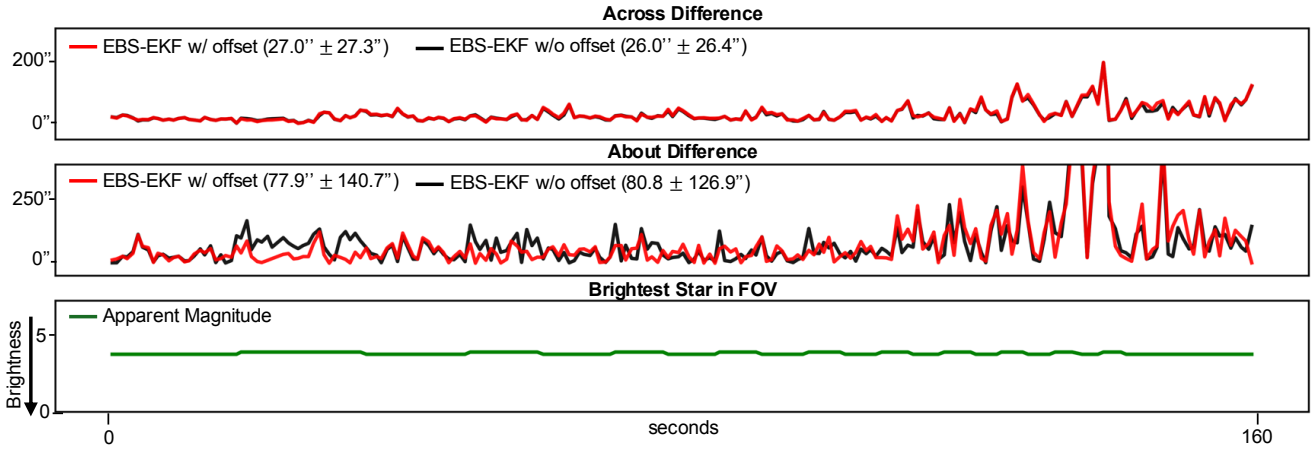


Figure 32. **Velocity Sweep 5 EBS-EKF w/ offset vs EBS-EKF w/o Offset**

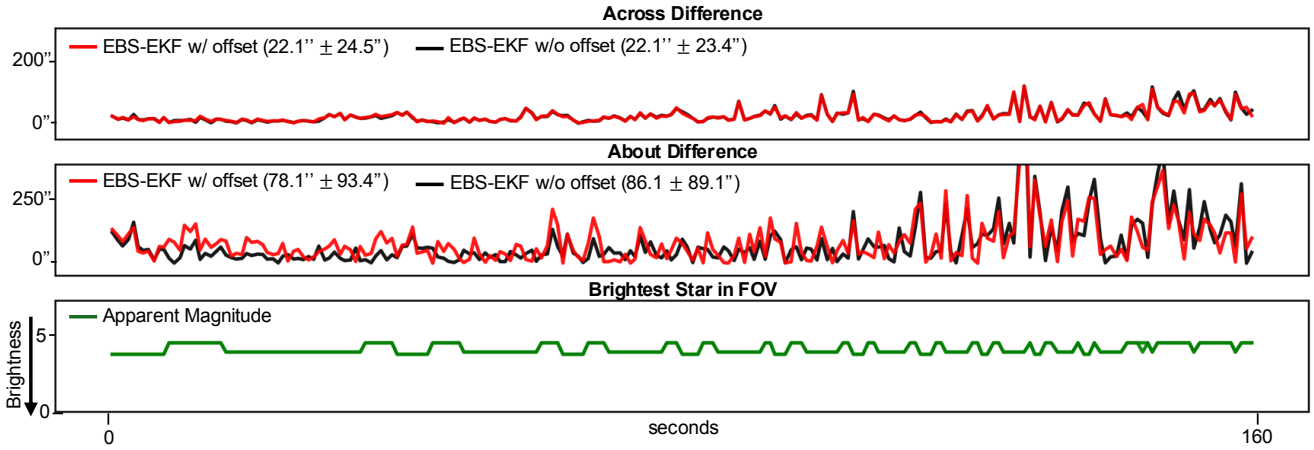


Figure 33. **Velocity Sweep 6 EBS-EKF w/ offset vs EBS-EKF w/o Offset**

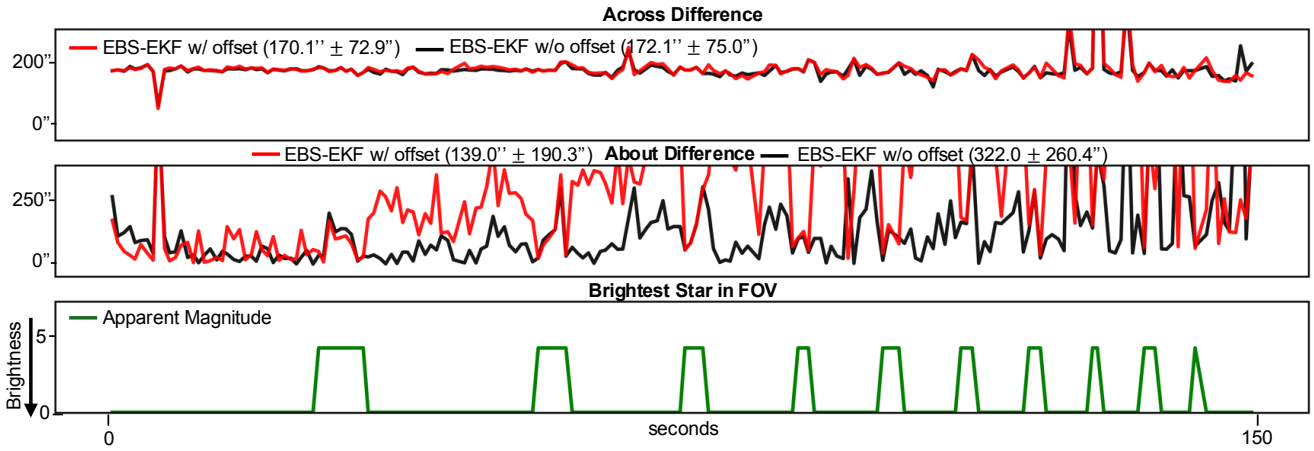


Figure 34. **Velocity Sweep 7 EBS-EKF w/ offset vs EBS-EKF w/o Offset**

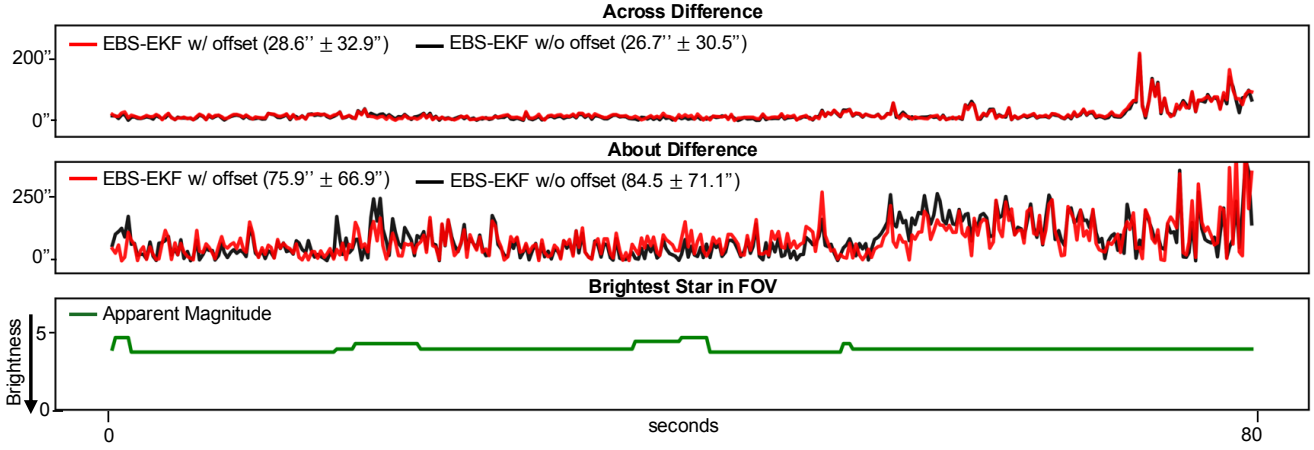




Figure 35. **Smooth Sine EBS-EKF w/ offset vs EBS-EKF w/o Offset**

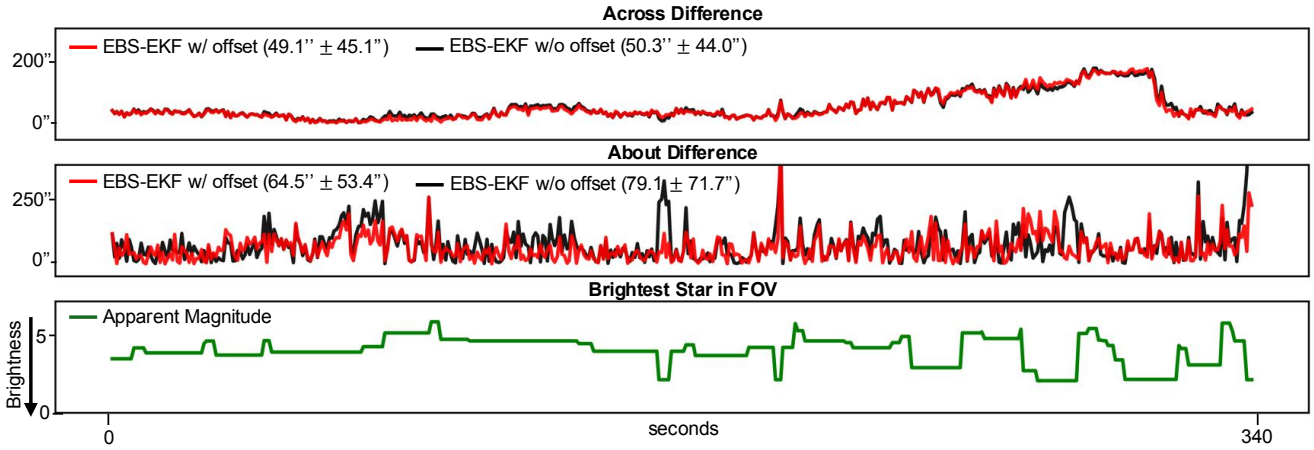


Figure 36. **Tilt Ladder EBS-EKF w/ offset vs EBS-EKF w/o Offset**