Adaptive Multi-Sensor Fusion Based Object Tracking for Autonomous Urban Air Mobility Operations

Thomas Lombaerts* Kimberlee H. Shish†
NASA Ames Research Center, Moffett Field, CA 94035

Gordon Keller‡
University of California, Santa Cruz, CA 95064

Vahram Stepanyan§ Nicholas Cramer¶ Corey Ippolito‖
NASA Ames Research Center, Moffett Field, CA 94035

Autonomous operations are a crucial aspect in the context of Urban Air Mobility and other emerging aviation markets. In order to enable this autonomy, systems must be able to build independently an accurate and detailed understanding of the own vehicle state as well as the surrounding environment, this includes detecting and avoiding moving objects in the sky, which can be cooperative (aircraft, UAM vehicles, etc.) as well as noncooperative (smaller drones, birds, ...). This paper focuses on the object tracking part that relies on adaptive multi-sensor fusion, taking into account specific properties and limitations of different sensor types. Results show the impact of dropouts of individual sensors on the accuracy of the tracking results for this adaptive sensor fusion approach.

I. Introduction

RECENTLY, there have been numerous developments in the area of urban air taxi operations, also known as urban air mobility or on-demand mobility applications that help to meet the challenging mobility needs of an increasing population density in metropolitan areas, taking into account the limited capacity of the currently available transportation infrastructure. Two of the enabling technologies of this transportation concept are the use of new vehicle concepts and advanced automation technologies, such that a safe, efficient, and accessible on-demand service for passengers and cargo is established. Some of these future concepts for on-demand mobility aim for fully autonomous vehicle operations, proposing the use of advanced sensing and algorithms to replace the functions of an on-board pilot [1].

Autonomous functionality promises to offer improved efficiency and cost savings. Increased vehicle autonomy is currently spearheaded by the automotive (such as Google's self-driving car project Waymo, as well as Lyft, etc.) and the UAV industries [2, 3]. Additionally, the exploration of new on-board sequencing, spacing and collision avoidance functions is encouraged by the market in order to increase overall capacity and efficiency in the airspace. [3] In this context, perception systems, capable of building an accurate and detailed understanding of the vehicle state and its surrounding environment, are critical to safe autonomous operations. This surrounding environment includes moving objects in the sky, which can be cooperative (aircraft, UAM vehicles, etc.) as well as noncooperative (smaller drones, birds, ...). The distinction between both lies in the fact that there is some form of data exchange (ADS-B or similar) with the former category in order to coordinate path planning and collision avoidance. This exchange of information is nonexistent for the latter category, which means that other sensors will be required to detect and track these objects independently from the own ship. Object detection and tracking are two essential prerequisites for the purpose of collision avoidance.

*Aerospace Research Engineer, KBR Wyle Services, Intelligent Systems Division, Mail Stop 269-1, AIAA Associate Fellow, email: thomas.lombaerts@nasa.gov

†Aerospace Research Engineer, Intelligent Systems Division, Mail Step 269-1, AIAA member, email: kimberlee.h.shish@nasa.gov

‡Computer Engineering Graduate Student, UCSC, 1156 High St.

§Principal Scientist, KBR Wyle Services, Intelligent Systems Division, Mail Stop 269-1, AIAA senior member, email: vahram.stepanyan@nasa.gov.

¶Aerospace Research Engineer, Intelligent Systems Division, Mail Step 269-1, AIAA member, email: nicholas.b.cramer@nasa.gov.

‖Aerospace Scientist, NASA Ames Research Center, Moffet Field, CA 94035, AIAA Member, email: corey.a.ippolito@nasa.gov

A. Focus of this paper

This paper focuses on the object tracking part that relies on adaptive multi-sensor fusion, taking into account specific properties and limitations of different sensor types. The sensor fusion is done by means of a Kalman Filter. More precisely, the sensor fusion strategy is made adaptive, so that lower sampling rates of specific sensors are taken into account, as well as loss of signal (for example because the object is outside the field of view or the useful range for a particular sensor, or because of a sensor malfunction resulting in a sensor signal dropout), and invalid data. This capability is achieved by incorporating valid flags in the weighting matrix of the Kalman filter.

B. Literature survey

This research is partly inspired by previous work published in DO-365. Appendix F in DO-365 discusses a Kalman Filter based integration approach to combine data from a radar tracker, ADS-B tracker and an AST (active surveillance transponder) tracker [4]. Ref. [5] detects moving objects based on grids, by differentiating between moving and stationary objects in occupancy grids. This is based on the continuous occupation of the grid cells. This publication also contains a literature review of other object tracking methods and a tradeoff between lasers and cameras. Ref. [6] discusses the various elements of a conventional multiple target tracking (MTT) system. Multiple hypothesis tracking (MHT) is the generally accepted preferred method for solving data association problems this this kind of MTT systems. Petrovskaya developed a tracking module that provides reliable tracking of moving vehicles from a high-speed moving platform [7]. The sensors involved are laser range finders. The estimation was done using a single Bayes filter per vehicle. This setup was able to detect poorly visible black vehicles. Ref. [8] tracks with a Kalman Filter based sensor fusion algorithm with different coordinate frames. Sensors used are pulsed radar, aided by two infrared and two visible cameras. It was found that an extended filter in rectangular coordinates was most adequate for airborne applications. Ref. [9] presents a taxonomy of object tracking using 3D sensors. Object tracking algorithms are divided into 2 main categories based on object representation scheme, namely Tracking-by-detection (or Discriminative) Approaches and Model-free (or Generative) Approaches. Two baseline 3D object tracking algorithms are discussed, taking LIDAR point cloud data (PCD's) as input (after the ground removal process). The first one is a baseline Kalman Filter 3D object tracker which is a 3D constant acceleration KF with Gating Data Association for robust object centroid tracking in consecutive PCD's. The second algorithm is a baseline Mean Shift (MS) 3D object tracker, which relies on an iterative procedure to locate objects. Ref. [10] sketches the path from sensor raw data to the list of object hypotheses. However, this method relies heavily on classifications of parking lots, and the assumption that vehicles move along lanes, which is not necessarily a valid assumption in the context of UAM vehicles. Previous work by one of the co-authors focused on visual tracking of maneuvering targets [11].

C. Paper Structure

The structure of this paper is as follows. Sections II and III describe respectively two different two vehicle scenarios and the relevant sensor characteristics which have been used to evaluate the performance of the developed object tracker. Section IV focuses specifically on the image data processing, where Section V elaborates how these processed image data are combined with other (simulated) sensor measurements in a Kalman Filter based object tracker setup. Section VI shows some preliminary results, which included nominal results as well as situations with occasional sensor dropouts. Finally, Section VII gives some overall conclusions and recommendations based on this work.

II. Scenario

Two scenarios are considered in this publication to evaluate the object tracking capabilities of the developed setup. The first scenario involves two dissimilar vehicles on different tracks. The second scenario considers two identical vehicles on the same track but with a constant time separation.

A. Scenario 1

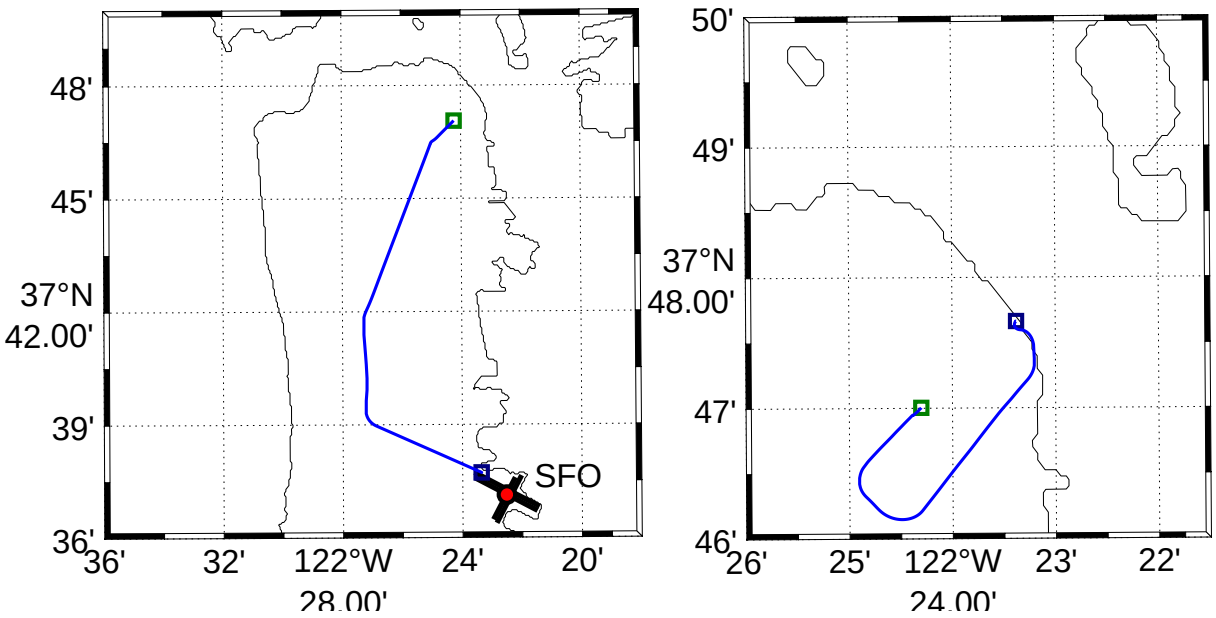
The first scenario involves two dissimilar vehicles, namely a quadrotor configuration and a lift-plus-cruise (LPC) configuration which consists of a set of four lifting rotors and a separate pulling propeller for forward flight. Both configurations are shown in Fig. 1 and were given the callsigns 'Duck' (quad) and 'Goose' (LPC) respectively.

Goose flies from San Francisco Airport to the rooftop of the parking structure at the intersection of Mission and Fifth in downtown San Francisco. Cruise altitude is 1,000 ft and cruise speed is 120 kts. This flight profile includes a full transition from vertical takeoff to forward cruise flight along three consecutive waypoints and a descent followed

(a) lift-plus-cruise configuration ‘Goose’ (b) quadrotor configuration ‘Duck’

Fig. 1 Vehicle configurations

by a vertical final approach to the rooftop. Duck flies from the ferry building along a circuit, followed by the same descent to the same aforementioned parking rooftop. In cruise, Duck flies with 60 kts at 500 ft. For this scenario, both tracks were timed in such a way that they result in a ‘collision’ point during descent towards the landing zone, at the top of descent for Duck, when Goose approaches Duck from above.



(a) Trajectory of Goose (b) Trajectory of Duck

Fig. 2 Trajectories of both vehicles

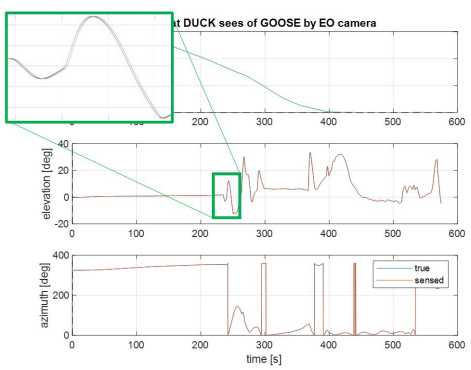
B. Scenario 2

The second scenario consists of two identical trajectories of two identical vehicles. Two quadrotors (see Fig. 2(a)) ‘Duck’ and ‘Goose’ fly both from the ferry building to the aforementioned parking rooftop in downtown SF (see Fig. 2(b)). The cruise altitude is 500 ft and the cruise speed is 60 kts. There is a 10 seconds delay between both, and no loss of separation occurs between both vehicles during this scenario.

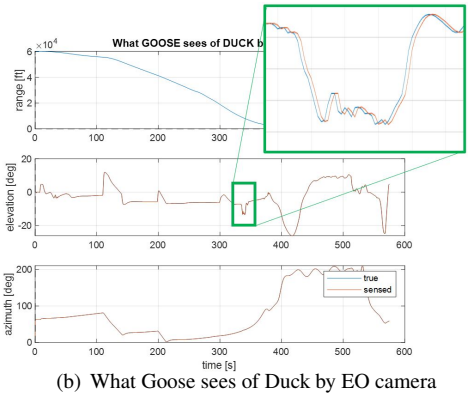
III. Sensor Characteristics

There are five types of sensors that could be used in a general Urban Air Mobility vehicle in the current state of the art, namely ADS-B or some other kind of position sharing communication channel for cooperative vehicles, electro-optical camera's, infrared cameras, lidar and radar. Each sensor type has different sensed variables, together with their respective sensor characteristics, such as limitations on field of view and useful range, noise, latency, update rate and resolution. Some characteristic values from the literature of state-of-the-art sensor hardware [12] are given in Table 1. Some sensors have a fairly narrow field of view, which can be compensated by placing an array of sensors with varying alignment angles to cover an overall wider field of view, depending on weight and power constraints of the vehicle.

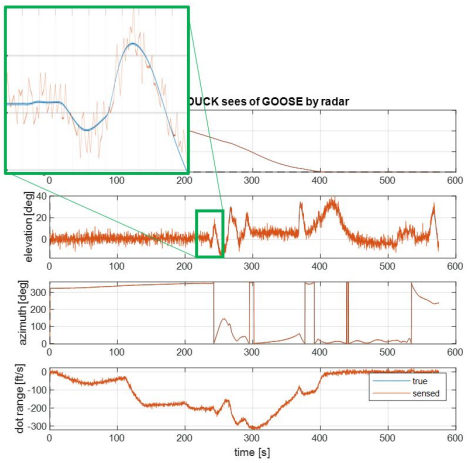
Figure 3 illustrates how both vehicles Duck and Goose see each other in scenario 1 by means of an EO camera and radar equipment respectively, with the characteristics specified in Table 1. Fig. 3(a)–3(b) show the elevation and azimuth of the other vehicle, as detected by Duck in Fig. 3(a) and Goose in Fig. 3(b) respectively. The enlarged areas illustrate the time delay of 0.1s for the EO camera. Fig. 3(c)–3(d) show the range, elevation, azimuth and range rate of the other vehicle, as detected by Duck in Fig. 3(c) and Goose in Fig. 3(d) respectively. The enlarged areas illustrate the latency of 0.1s as well as the significant noise levels for the radar measurements.



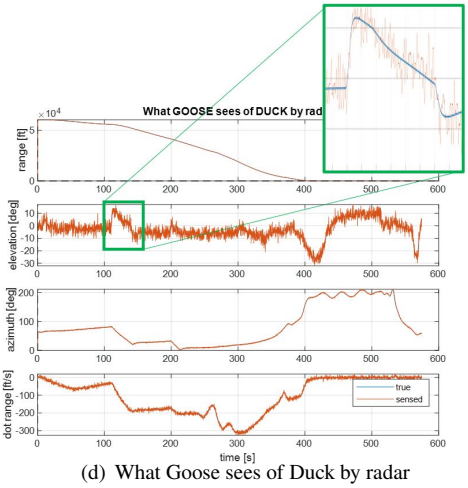
(a) What Duck sees of Goose by EO camera



(b) What Goose sees of Duck by EO camera



(c) What Duck sees of Goose by radar



(d) What Goose sees of Duck by radar

Fig. 3 EO camera and radar measurements for both vehicles in scenario 1

These sensors don't give any information beyond a certain distance (and radar also below a distance threshold), and have a limited field of view as well. However, these constraints have not been taken into account here. Field of view limits can be circumvented by placing multiple sensors in an array, and current range limits are too restrictive for the scenarios that were considered. Moreover, it is assumed that these limitations will improve significantly as new state-of-the-art sensors with the latest technology will become available.

Table 1 Sensor characteristics

sensor	ADS-B	EO camera	IR camera	lidar	radar
sensed variables	position: lon, lat [deg] altitude [ft] hor spd: Vn,Ve [kts] vert spd: hdot [ft/s] ID: callsign	azimuth [deg] elevation [deg]	azimuth [deg] elevation [deg]	azimuth [deg] elevation [deg] range [ft]	azimuth [deg] elevation [deg] range [ft] dot range [ft/s]
limits on field of view	–	±1.2deg	±3.1deg	az: ±30deg el: ±15deg	az: ±60deg el: ±40deg
max useful range	–	18,700m	2,400m	1,000m	2,000m dot range: 14.56m/s
min useful range	–	0m	0m	0m	20m dot range: −14.56m/s
noise sigma	–	$9.1 \cdot 10^{-3}$deg	$6.79 \cdot 10^{-2}$deg	az/el: $2.5 \cdot 10^{-2}$deg range: $2 \cdot 10^{-2}$m	az: 1deg el: 3deg range: 3.25m rangedot: 2m/s
latency	0.5s	0.1s	0.1s	0.01s	0.1s
update rate	1/s	30fps	30fps	100Hz	10Hz
resolution	lon/lat/alt: $1 \cdot 10^{-8}$deg Vn/Ve: 4kts hdot: 64ft/min	$1.3 \cdot 10^{-3}$deg	$9.7 \cdot 10^{-3}$deg	az/el: $2.5 \cdot 10^{-2}$deg range: $1 \cdot 10^{-2}$m	az/el: 1deg range: 3.25m dot range: 0.91m/s

IV. Image Data Processing

Visual object detection (VOD) from digital images is a heavily researched problem in the field of Computer Vision (CV). Early methods of VOD often required starting positions, a priori models of what to search for, or other aiding information. The advent of Machine Learning (ML)-based VOD methods represent a major milestone in the field with the adoption of Convolutional Neural Nets (CNN) as visual processing models. These methods are trained to locate and detect specific targets depending on the application. CNNs are structured as cascaded neural layers which transform inputs into an n-dimensional vector representation. In CNN-based object detection, a training phase tunes the model weights. Abstracted sub-patterns (or "features") are generated in the process. A training dataset with annotated objects serves as the ground truth which the model attempts to converge to. Deep Learning (DL) extends CNN by increasing the number of layers. In the case of single VOD, the model expects a single object to be present in the image. A testing dataset is used to validate the efficacy of the trained model, and this process is repeated with new hyperparameters corresponding to the architecture, until the model is suitably accurate without being over-trained. Multi-Object Detection (MOD) works similarly to VOD but can detect an arbitrary number of objects from an arbitrary number of classes to be detected within an image. Two primary tasks are required in MOD: detection and classification. Detection is the process of identifying where in the image objects are located. Locations may be represented by a single point (e.g., "center-of-gravity"), a bounding box (often represented as pair of opposing corner points) which incorporates the projected dimensions of the object, or the arbitrarily-shaped group of pixels exclusively belonging to the object known as a mask. Classification sorts these objects into classes corresponding to their ground truth annotations. The classification model is pre-trained with representative sets of images for each class, and as such the algorithm will ideally only detect the objects for which it has been trained.

Two-stage detectors are a class of CNN-based MOD algorithms which combine the detection and classification in a sequential fashion, first identifying regions of the image corresponding to potential objects and subsequently identifying said objects. Single-stage MOD on the other hand employs a faster means of processing image at the cost of slightly poorer accuracy than two-stage methods*. In single-stage detectors, a single network is used to accomplish segmentation and detection, or a predefined set of dense subregions are used in lieu of learning-based segmentation. As such, this class of MOD algorithms is ideal for faster processing on CPUs (and, in some cases, real-time processing on GPUs) and makes them preferable to two-stage algorithms when MOD is being used in-the-loop for controls and avoidance.

You Only Look Once (YOLO) is a single-stage MOD algorithm first proposed in 2016 in [13]. The algorithm achieves single-stage processing using a predefined set of dense subregions to identify the dominant objects within. The variant of YOLO used in this work (YOLOv3) is the third official release of the algorithm wherein an updated network is trained and miscellaneous improvements are introduced [14]. YOLOv3 as a tradeoff between appropriate algorithm performance and the availability of repositories at the time of the research conducted. A pytorch-based implementation of YOLOv3 available under the GPL-3.0 license was utilized for this work [15]. The primary YOLOv3 darknet model comes pre-trained on COCO 2017 [16] when cloning the original repository. At the time of writing this manuscript, a YOLOv5 repository has been folded into the now archived repository for the YOLOv3 variant.

Visual sensing is simulated in this work via a simple pipeline. Flight videos for a Beechcraft Baron 58 are generated in an X-plane simulation for the two scenarios described in Section II. The camera vantage point is a first-person view roughly approximating where placement of a front-facing camera on a UAM/AAM aircraft could be placed in practice. The imagery is then processed using MOD to estimate where objects of concern are located relative to the ownship. Frames from the flight videos serve as input to a script running a pre-trained YOLOv3 darknet model. The pseudocode for this script is based on sample code in [15]. The input frames are passed through the darknet YOLOv3 model on each iteration for the primary loop.

Annotated videos are rendered as outputs from the YOLOv3 program. Two representative image frames are depicted in Figure 4 which illustrate how the algorithm tags detections. Our implementation of the example script from [15] outputs a plain-text file containing same the detection information for parsing and filtering. Detections are comma-delimited and frames are newline-delimited in the output file. A simple filtering script parses and strips away superfluous detection instances from the output file (i.e., detections not relevant to the object tracking Kalman filter), after which only frame instances containing "airplane"-labelled objects are preserved. Frame instances where the detector does not register the other aircraft when in frame or when the aircraft is out of frame constitute skips in detection.

The filtering script also converts the top-left/bottom-right bounding box coordinates to the center point of the

*Misidentification or misinterpretation of a region of the image as an appropriate object is false positive, and missed objects which should have been detected and classified constitute false negatives.

bounding box. The center point is calculated via the following equation:

$$p_c = p_{tl} + \frac{p_{br} - p_{tl}}{2} \tag{1}$$

where $p_{tl} = (x_{tl}, y_{tl})$ is the bounding box top-left corner, $p_{br} = (x_{br}, y_{br})$ is the bounding box bottom-right corner, and $p_c = (u, v)$ is the center point of the detection bounding box measured in pixels. Dimension data (e.g., height and width) are excluded from the Kalman filter in this work, though this data could be useful in a modified version of the Kalman filter for approximating the rate of change of the detected object's projected size. The center coordinates for the highest-confidence "airplane" detection in each frame is then passed to the object tracking Kalman filter.

As the simulated camera is rigidly fixed relative to the ownship reference frame, the conversion from image space into the camera frame space (which is body-fixed) is as follows:

$$\begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix}^{-1} s \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} X_C \\ Y_C \\ Z_C \end{bmatrix} \tag{2}$$

where f_x, f_y, c_x, and c_y are intrinsic camera parameters, and X_C, Y_C, and Z_C are the coordinates of the center point of the bounding box in the camera frame. Azimuth γ and elevation α for the detected object may then be derived from the detected object's position relative to the body-fixed frame via:

$$\gamma = \arctan\left(\frac{X_C}{f}\right) \tag{3}$$

$$\alpha = \arctan\left(\frac{Y_C}{f}\right) \tag{4}$$

(a) First YOLO processed image

(b) Second YOLO processed image

Fig. 4 Two YOLOv3 annotated image examples from x-plane simulations of the aircraft encounter described in Section II. The detection details include the class of the detected object (e.g., "airplane"), the bounding box for the detection, and the confidence level of the detection as a float between 0 and 1.

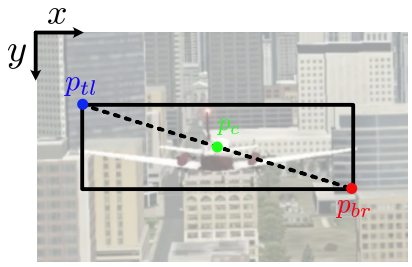


Fig. 5 Bounding box coordinates for the top-left (p_{tl}) and bottom-right(p_{br}), and the center (p_c), by YOLOv3.

V. Object Tracker Setup

The actual object tracker is a conventional Kalman Filter as elaborated in detail in Ref. [4]. The filter gain calculation is modified in order to incorporate valid flags for the different sensor measurements, which makes this method adaptive and robust against invalid data, loss of signal, outliers and lower sampling rates. The current implementation uses measurements from an EO camera and a radar, but this modular architecture is fairly straightforward to extend so that other sensors can be incorporated as well. Fig. 6 illustrates the general overview how the Kalman Filter is set up. The prediction step assumes constant velocity for calculating the predicted state $\hat{\mathbf{x}}_k$ based on the previous state $\mathbf{x}_{k-1}$. The state consists of six elements, namely the three position coordinates and the three velocity coordinates, all in ENU coordinates (East–North–Up). Another prediction strategy is using multiple models for constant velocity, constant acceleration and constant turn rate, where the one is selected that fits the measurements best. While this approach would increase accuracy and flexibility, it would also increase complexity and computational load, while experiments have shown that the current accuracy under constant velocity assumption is satisfactory. A minor delay for the tracking signal occurred in turns, but well within the acceptable margins. A predicted measurement $\hat{\mathbf{z}}_k$ is calculated, which is a function of the predicted state $\mathbf{h}(\hat{\mathbf{x}}_k)$. This predicted measurement consists of the measured variables that are provided by both sensors, namely range and range rate, azimuth and elevation from the radar as well as azimuth and elevation from the EO camera. By comparing the actual measurement $\mathbf{z}_k$ with the predicted measurement $\hat{\mathbf{z}}_k$, one can calculate the innovation $\varepsilon = \mathbf{z}_k - \hat{\mathbf{z}}_k = \mathbf{z}_k - \mathbf{h}(\hat{\mathbf{x}}_k)$. Main drivers for the innovation are errors in the predicted state (due to the constant speed assumption) and sensor disturbances. As last but one, the Kalman gain matrix $\mathbf{W}$ is calculated by means of the predicted covariance $\hat{\mathbf{P}}_k$, measurement matrix $\mathbf{H} = \frac{d\mathbf{h}(\mathbf{x})}{d\mathbf{x}}$, and the innovation covariance matrix $\mathbf{S}$. This Kalman gain matrix $\mathbf{W}$ is weighted with valid flags to deal appropriately with the sensor measurements as explained previously. The corrected state $\mathbf{x}_k$ is calculated by means of predicted state $\hat{\mathbf{x}}_k$, the innovation ε and the Kalman gain matrix $\mathbf{W}$ via the following equation: $\mathbf{x}_k = \hat{\mathbf{x}}_k + \mathbf{W} \cdot \varepsilon$. Finally, the corrected covariance $\mathbf{P}_k$ is calculated. The complete procedure is described in more detail below.

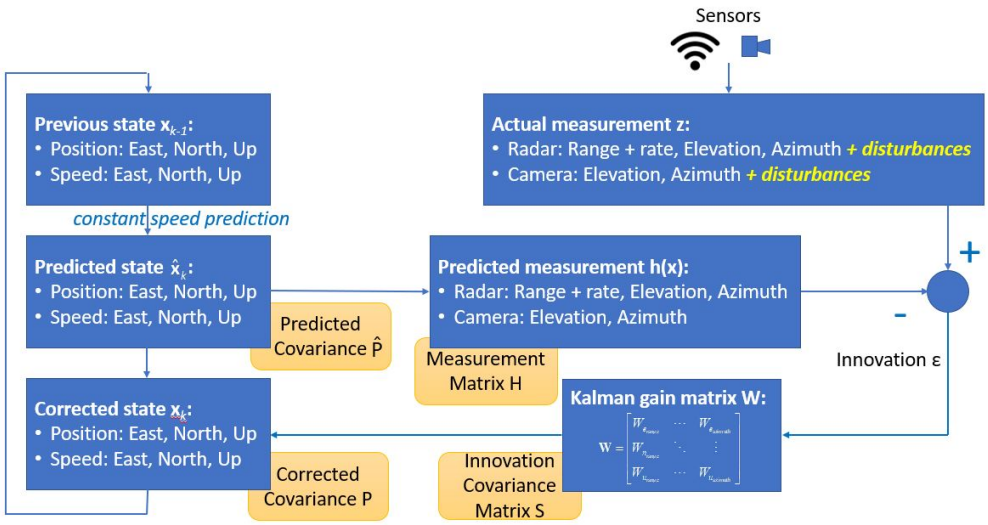


Fig. 6 General overview of the Kalman Filter setup

A. State Prediction $\hat{\mathbf{x}}_k$

The one step ahead prediction of the state vector is split up as follows:

$$\hat{\mathbf{x}}_k = \begin{bmatrix} \hat{\mathbf{x}}_{k_1} \\ \hat{\mathbf{x}}_{k_2} \end{bmatrix} \tag{5}$$

With the position in the upper half of the vector:

$$\hat{\mathbf{x}}_{k_1} = \begin{bmatrix} \hat{E} \\ N \\ U \end{bmatrix}_k = \begin{bmatrix} E \\ N \\ U \end{bmatrix}_{k-1} + \Delta t \cdot \begin{bmatrix} \dot{E} \\ N \\ U \end{bmatrix}_{0_{k-1}} \tag{6}$$

And the velocity in the lower half of the vector:

$$\hat{\mathbf{x}}_{k_2} = \begin{bmatrix} \hat{\dot{E}} \\ N \\ U \end{bmatrix}_k = \begin{bmatrix} \dot{E} \\ N \\ U \end{bmatrix}_{0_{k-1}} \tag{7}$$

Note that the coordinates are East–North–Up and that the velocity is assumed to be close to constant. Moreover:

$$\begin{bmatrix} \dot{E} \\ N \\ U \end{bmatrix}_{0_{k-1}} = \mathbf{L}^\top \cdot \begin{bmatrix} \dot{E} \\ N \\ U \end{bmatrix}_{T_{k-1}} \tag{8}$$

where $\mathbf{L}^\top$ is the rotation matrix from the target ENU-frame to the ownship ENU-frame, as elaborated in the Appendix.

B. State Error Covariance Prediction $\hat{\mathbf{P}}_k$

$$\hat{\mathbf{P}}_k = \mathbf{F}\mathbf{P}_{k-1}\mathbf{F}^\top + \mathbf{Q} \tag{9}$$

with:

$$\mathbf{F} = \left[\begin{array}{ccc|ccc} 1 & 0 & 0 & \Delta t & 0 & 0 \\ 0 & 1 & 0 & 0 & \Delta t & 0 \\ 0 & 0 & 1 & 0 & 0 & \Delta t \\ \hline & \mathbf{0}_{3\times3} & & 1 & 0 & 0 \\ & & & 0 & 1 & 0 \\ & & & 0 & 0 & 1 \end{array}\right] \qquad \mathbf{Q} = \mathbf{G}\boldsymbol{\Gamma}\mathbf{G}^\top \tag{10}$$

and:

$$\mathbf{G} = \left[\begin{array}{ccc} 0.5\Delta t^2 & 0 & 0 \\ 0 & 0.5\Delta t^2 & 0 \\ 0 & 0 & 0.5\Delta t^2 \\ \hline \Delta t & 0 & 0 \\ 0 & \Delta t & 0 \\ 0 & 0 & \Delta t \end{array}\right] \qquad \boldsymbol{\Gamma} = \begin{bmatrix} \sigma_e^2 & 0 & 0 \\ 0 & \sigma_n^2 & 0 \\ 0 & 0 & \sigma_u^2 \end{bmatrix} \tag{11}$$

C. Predicted Measurement Calculation $\hat{\mathbf{z}}_k$

The predicted measurement vectors of radar and EO camera consist of range ρ, elevation α, azimuth γ, and range rate $\dot{\rho}$, and are described as follows:

$$\hat{\mathbf{z}}_k = \begin{bmatrix} \hat{\mathbf{z}}_{k_{\text{radar}}} & \hat{\mathbf{z}}_{k_{\text{EOcam}}} \end{bmatrix}^\top \tag{12}$$

$$\hat{\mathbf{z}}_{k_{\text{radar}}} = \begin{bmatrix} \hat{\rho} & \hat{\alpha} & \hat{\gamma} & \hat{\dot{\rho}} \end{bmatrix}^\top \tag{13}$$

$$\hat{\mathbf{z}}_{k_{\text{EOcam}}} = \begin{bmatrix} \hat{\alpha} & \hat{\gamma} \end{bmatrix}^\top \tag{14}$$

since the EO camera cannot infer any distance information due to the lack of stereo view and depth perception. The individual measured variables are defined as follows:

$$\hat{\rho} = \sqrt{\hat{X}_B^2 + \hat{Y}_B^2 + \hat{Z}_B^2} = \sqrt{\hat{E}^2 + \hat{N}^2 + \hat{U}^2} \tag{15}$$

$$\hat{\alpha} = \arctan \frac{-\hat{Z}_B}{\hat{\rho}_{\text{hor}}} \tag{16}$$

$$\hat{\gamma} = \arctan \frac{\hat{Y}_B}{\hat{X}_B} \tag{17}$$

$$\dot{\hat{\rho}} = \frac{\hat{X}_B \cdot \Delta\dot{\hat{X}}_B + \hat{Y}_B \cdot \Delta\dot{\hat{Y}}_B + \hat{Z}_B \cdot \Delta\dot{\hat{Z}}_B}{\hat{\rho}} = \frac{\hat{E} \cdot \Delta\dot{\hat{E}} + \hat{N} \cdot \Delta\dot{\hat{N}} + \hat{U} \cdot \Delta\dot{\hat{U}}}{\hat{\rho}} \tag{18}$$

where:

$$\hat{\rho}_{\text{hor}} = \sqrt{\hat{X}_B^2 + \hat{Y}_B^2} \tag{19}$$

and:

$$\begin{bmatrix} \hat{X}_B \\ Y_B \\ Z_B \end{bmatrix} = \mathbf{M}^\top \cdot \begin{bmatrix} \hat{E} \\ N \\ U \end{bmatrix}_k = \mathbf{M}^\top \cdot \hat{\mathbf{x}}_{k_1} \qquad \begin{bmatrix} \dot{\hat{X}}_B \\ Y_B \\ Z_B \end{bmatrix} = \mathbf{M}^\top \cdot \begin{bmatrix} \dot{\hat{E}} \\ N \\ U \end{bmatrix}_k = \mathbf{M}^\top \cdot \hat{\mathbf{x}}_{k_2} \tag{20}$$

where $\mathbf{M}^\top$ is the rotation matrix from the ENU reference frame to the body fixed reference frame, as defined in the Appendix.

D. Kalman Filter Gain W

The following steps are performed in order to calculate the gain of the filter:

1) Calculate measurement matrix **H**
2) Calculate innovation covariance matrix **S**
3) Calculate gain matrix **W**

Each step will be elaborated in detail next:

1. Measurement matrix **H**

The measurement matrix **H** is basically the Jacobian of the predicted measurement vector $\hat{\mathbf{z}}_k = \begin{bmatrix} \hat{\mathbf{z}}_{k_{\text{radar}}} & \hat{\mathbf{z}}_{k_{\text{EOcam}}} \end{bmatrix}^\top = \begin{bmatrix} \hat{\mathbf{h}}_{\text{radar}}(\hat{\mathbf{x}}_k) & \hat{\mathbf{h}}_{\text{EOcam}}(\hat{\mathbf{x}}_k) \end{bmatrix}^\top$ with respect to the state vector $\hat{\mathbf{x}}_k = \begin{bmatrix} \hat{\mathbf{x}}_{k_1} & \hat{\mathbf{x}}_{k_2} \end{bmatrix}^\top$, with $\mathbf{x}_{k_1}$ being the position components and $\mathbf{x}_{k_2}$ the velocity components, both in the ENU reference frame. Decomposing in 4 blocks results in the following structure:

$$\mathbf{H}(\mathbf{x}_k) = \frac{d\hat{\mathbf{h}}(\hat{\mathbf{x}}_k)}{d\hat{\mathbf{x}}_k} = \begin{bmatrix} \frac{d\hat{\mathbf{h}}_{\text{radar}}(\hat{\mathbf{x}}_{k_1}, \hat{\mathbf{x}}_{k_2})}{d\hat{\mathbf{x}}_{k_1}} & \frac{d\hat{\mathbf{h}}_{\text{radar}}(\hat{\mathbf{x}}_{k_1}, \hat{\mathbf{x}}_{k_2})}{d\hat{\mathbf{x}}_{k_2}} \\ \frac{d\hat{\mathbf{h}}_{\text{EOcam}}(\hat{\mathbf{x}}_{k_1})}{d\hat{\mathbf{x}}_{k_1}} & \frac{d\hat{\mathbf{h}}_{\text{EOcam}}(\hat{\mathbf{x}}_{k_1})}{d\hat{\mathbf{x}}_{k_2}} = \mathbf{0}_{2\times 3} \end{bmatrix} \tag{21}$$

Note that the radar measurements include the sensed range rate $\dot{\rho}$, which depends on the velocity components and thus $\mathbf{x}_{k_2}$. The EO camera measurements include only elevation α and azimuth γ, thus depending only on the position components and thus $\mathbf{x}_{k_1}$. Each block in Eq. (21) is elaborated next:

$$\frac{d\hat{\mathbf{h}}_{\text{radar}}(\hat{\mathbf{x}}_{k_1}, \hat{\mathbf{x}}_{k_2})}{d\hat{\mathbf{x}}_{k_1}} = \begin{bmatrix} \frac{d\hat{\rho}}{dE} & \frac{d\hat{\rho}}{dN} & \frac{d\hat{\rho}}{dU} \\ \frac{d\hat{\alpha}}{dE} & \frac{d\hat{\alpha}}{dN} & \frac{d\hat{\alpha}}{dU} \\ \frac{d\hat{\gamma}}{dE} & \frac{d\hat{\gamma}}{dN} & \frac{d\hat{\gamma}}{dU} \\ \hline \frac{d\dot{\hat{\rho}}}{dE} & \frac{d\dot{\hat{\rho}}}{dN} & \frac{d\dot{\hat{\rho}}}{dU} \end{bmatrix} = \begin{bmatrix} \begin{bmatrix} \frac{d\hat{\rho}}{dX_B} & \frac{d\hat{\rho}}{dY_B} & \frac{d\hat{\rho}}{dZ_B} \\ \frac{d\hat{\alpha}}{dX_B} & \frac{d\hat{\alpha}}{dY_B} & \frac{d\hat{\alpha}}{dZ_B} \\ \frac{d\hat{\gamma}}{dX_B} & \frac{d\hat{\gamma}}{dY_B} & \frac{d\hat{\gamma}}{dZ_B} \end{bmatrix} \cdot \mathbf{M}^\top \\ \hline \frac{d\dot{\hat{\rho}}}{dE} \quad \frac{d\dot{\hat{\rho}}}{dN} \quad \frac{d\dot{\hat{\rho}}}{dU} \end{bmatrix} = \begin{bmatrix} \begin{bmatrix} \frac{\hat{X}_B}{\hat{\rho}_0} & \frac{\hat{Y}_B}{\hat{\rho}_0} & \frac{\hat{Z}_B}{\hat{\rho}_0} \\ \frac{\hat{X}_B \cdot \hat{Z}_B}{\hat{\rho}_{\text{hor}_0} \hat{\rho}_0^2} & \frac{\hat{Y}_B \cdot \hat{Z}_B}{\hat{\rho}_{\text{hor}_0} \hat{\rho}_0^2} & \frac{-\hat{\rho}_{\text{hor}}}{\hat{\rho}_0^2} \\ \frac{-\hat{Y}_B}{\hat{\rho}_{\text{hor}_0}^2} & \frac{\hat{X}_B}{\hat{\rho}_{\text{hor}_0}^2} & 0 \end{bmatrix} \cdot \mathbf{M}^\top \\ \hline \frac{d\dot{\hat{\rho}}}{dE} \quad \frac{d\dot{\hat{\rho}}}{dN} \quad \frac{d\dot{\hat{\rho}}}{dU} \end{bmatrix} \tag{22}$$

with:

$$\frac{d\hat{\dot{\rho}}}{dE} = \frac{1}{\hat{\rho}_0}\left(\Delta\hat{\dot{E}} - \frac{\hat{E}}{\hat{\rho}_0}\hat{\dot{\rho}}\right) \tag{23}$$

$$\frac{d\hat{\dot{\rho}}}{dN} = \frac{1}{\hat{\rho}_0}\left(\Delta\hat{\dot{N}} - \frac{\hat{N}}{\hat{\rho}_0}\hat{\dot{\rho}}\right) \tag{24}$$

$$\frac{d\hat{\dot{\rho}}}{dU} = \frac{1}{\hat{\rho}_0}\left(\Delta\hat{\dot{U}} - \frac{\hat{U}}{\hat{\rho}_0}\hat{\dot{\rho}}\right) \tag{25}$$

The denominators $\hat{\rho}_0$ in Eq. (22),(23),(24),(25) and $\hat{\rho}_{\text{hor}_0}$ in Eq. (22) are singularity protected as follows:

$$\hat{\rho}_0 = \max\left(\rho, \varepsilon_\rho\right) \tag{26}$$

$$\hat{\rho}_{\text{hor}_0} = \max\left(\rho_{\text{hor}}, \varepsilon_{\rho_{\text{hor}}}\right) \tag{27}$$

where ε_ρ and $\varepsilon_{\rho_{\text{hor}}}$ are fairly small positive values to prevent singularities.

$$\frac{d\hat{\mathbf{h}}_{\text{radar}}\left(\hat{\mathbf{x}}_{k_1}, \hat{\mathbf{x}}_{k_2}\right)}{d\hat{\mathbf{x}}_{k_2}} = \begin{bmatrix} \frac{d\hat{\rho}}{d\hat{E}} & \frac{d\hat{\rho}}{d\hat{N}} & \frac{d\hat{\rho}}{d\hat{U}} \\ \frac{d\hat{\alpha}}{d\hat{E}} & \frac{d\hat{\alpha}}{d\hat{N}} & \frac{d\hat{\alpha}}{d\hat{U}} \\ \frac{d\hat{\gamma}}{d\hat{E}} & \frac{d\hat{\gamma}}{d\hat{N}} & \frac{d\hat{\gamma}}{d\hat{U}} \\ \hline \frac{d\hat{\dot{\rho}}}{d\hat{E}} & \frac{d\hat{\dot{\rho}}}{d\hat{N}} & \frac{d\hat{\dot{\rho}}}{d\hat{U}} \end{bmatrix} = \begin{bmatrix} \begin{bmatrix} \frac{d\hat{\rho}}{d\hat{X}_B} & \frac{d\hat{\rho}}{d\hat{Y}_B} & \frac{d\hat{\rho}}{d\hat{Z}_B} \\ \frac{d\hat{\alpha}}{d\hat{X}_B} & \frac{d\hat{\alpha}}{d\hat{Y}_B} & \frac{d\hat{\alpha}}{d\hat{Z}_B} \\ \frac{d\hat{\gamma}}{d\hat{X}_B} & \frac{d\hat{\gamma}}{d\hat{Y}_B} & \frac{d\hat{\gamma}}{d\hat{Z}_B} \end{bmatrix} \cdot \mathbf{M}^\top \\ \hline \frac{d\hat{\dot{\rho}}}{d\hat{E}} \quad \frac{d\hat{\dot{\rho}}}{d\hat{N}} \quad \frac{d\hat{\dot{\rho}}}{d\hat{U}} \end{bmatrix} = \begin{bmatrix} \mathbf{0}_{3\times3} \\ \hline \frac{\hat{E}}{\hat{\rho}_0} \quad \frac{\hat{N}}{\hat{\rho}_0} \quad \frac{\hat{U}}{\hat{\rho}_0} \end{bmatrix} \tag{28}$$

$$\frac{d\hat{\mathbf{h}}_{\text{EOcam}}\left(\hat{\mathbf{x}}_{k_1}\right)}{d\hat{\mathbf{x}}_{k_1}} = \begin{bmatrix} \frac{d\hat{\alpha}}{d\hat{E}} & \frac{d\hat{\alpha}}{d\hat{N}} & \frac{d\hat{\alpha}}{d\hat{U}} \\ \frac{d\hat{\gamma}}{d\hat{E}} & \frac{d\hat{\gamma}}{d\hat{N}} & \frac{d\hat{\gamma}}{d\hat{U}} \end{bmatrix} = \begin{bmatrix} \frac{d\hat{\alpha}}{d\hat{X}_B} & \frac{d\hat{\alpha}}{d\hat{Y}_B} & \frac{d\hat{\alpha}}{d\hat{Z}_B} \\ \frac{d\hat{\gamma}}{d\hat{X}_B} & \frac{d\hat{\gamma}}{d\hat{Y}_B} & \frac{d\hat{\gamma}}{d\hat{Z}_B} \end{bmatrix} \cdot \mathbf{M}^\top = \begin{bmatrix} \frac{\hat{X}_B \cdot \hat{Z}_B}{\hat{\rho}_{\text{hor}_0}\hat{\rho}_0^2} & \frac{\hat{Y}_B \cdot \hat{Z}_B}{\hat{\rho}_{\text{hor}_0}\hat{\rho}_0^2} & \frac{-\hat{\rho}_{\text{hor}}}{\hat{\rho}_0^2} \\ \frac{-\hat{Y}_B}{\hat{\rho}_{\text{hor}_0}^2} & \frac{\hat{X}_B}{\hat{\rho}_{\text{hor}_0}^2} & 0 \end{bmatrix} \cdot \mathbf{M}^\top \tag{29}$$

2. Innovation covariance matrix $\mathbf{S}$

The innovation covariance matrix is calculated as follows:

$$\mathbf{S} = \mathbf{H}\hat{\mathbf{P}}_k\mathbf{H}^\top + \mathbf{R}_{\text{sensors}} \tag{30}$$

where: $\mathbf{R}_{\text{sensors}} = \text{diag}\left[\sigma^2_{\rho_{\text{radar}}} \;\; \sigma^2_{\alpha_{\text{radar}}} \;\; \sigma^2_{\gamma_{\text{radar}}} \;\; \sigma^2_{\dot{\rho}_{\text{radar}}} \;\; \sigma^2_{\alpha_{\text{EOcam}}} \;\; \sigma^2_{\gamma_{\text{EOcam}}}\right]$. A diagonal matrix of sensor valid flags is defined next: $\mathbf{M}_{\text{sensors}_{\text{valid}}} = \text{diag}\left[\mu_{\text{radar}} \;\; \mu_{\text{radar}} \;\; \mu_{\text{radar}} \;\; \mu_{\text{radar}} \;\; \mu_{\text{EOcam}} \;\; \mu_{\text{EOcam}}\right]$. These valid flags $\mu_\bullet$ are one for valid sensor readings or zero for invalid ones. These flags allow to take into account lower sampling rates for specific sensors, signal dropouts (e.g. object is outside field of view), sensor malfunctions, invalid signals, etc. This setup assumes that there is a monitoring algorithm in place that drives the valid flag values accordingly. This valid flag matrix is needed for calculating the inverse of the innovation covariance matrix:

$$\mathbf{S}^{-1}_{\text{valid}} = \mathbf{M}_{\text{sensors}_{\text{valid}}} \cdot \mathbf{S}^{-1} \cdot \mathbf{M}_{\text{sensors}_{\text{valid}}} \tag{31}$$

3. Gain matrix $\mathbf{W}$

The Kalman filter gain matrix is calculated with all the aforementioned information as follows:

$$\mathbf{W} = \hat{\mathbf{P}}_k\mathbf{H}^\top\mathbf{S}^{-1}_{\text{valid}} \tag{32}$$

$$\mathbf{W}_{\text{valid}} = \mathbf{W}\mathbf{M}_{\text{sensors}_{\text{valid}}} \tag{33}$$

Invalid flags in $\mathbf{M}_{\text{sensors}_{\text{valid}}}$ will result in zero columns in $\mathbf{W}_{\text{valid}}$. As a consequence, that sensor information will not be used for the correction step. This means that the prediction step will have a higher relative importance for the states that are the main drivers of these invalid measurements, except when there are other valid sensor measurements that contain significant information of these states. This is illustrated in the results section.

E. State Update $\mathbf{x}_k$

The innovation is calculated first, which is the difference between the predicted measurements and the actual sensor measurements, including sensor limitations and perturbations:

$$\varepsilon = \mathbf{z}_k - \hat{\mathbf{z}}_k = \mathbf{z}_k - \mathbf{h}\left(\hat{\mathbf{x}}_k\right) \tag{34}$$

This innovation ε is then used to calculate the state update by means of the previously computed Kalman Gain matrix:

$$\mathbf{x}_k = \hat{\mathbf{x}}_k + \mathbf{W}_{\text{valid}} \cdot \varepsilon \tag{35}$$

F. Geodetic Coordinates Calculation

Next step is calculating the absolute position of the target in WGS-84 coordinates. This is achieved by combining the own position and the relative position of the target with respect to own position, which is the first half of the corrected state vector $\mathbf{x}_{k_1}$. Since both are defined in different reference frames, they are first converted to the ECEF (earth centered earth fixed) reference frame, then added together and converted back to WGS-84.

The own position is converted straight from WGS-84 (i.e. latitude, longitude and altitude) to ECEF via the transformation elaborated in the appendix, resulting in the position coordinates $[X \;\; Y \;\; Z]^\top_{\text{ECEF}}$. The relative position of the target with respect to the own position in ECEF coordinates is calculated as follows:

$$\begin{bmatrix} X \\ Y \\ Z \end{bmatrix}_{(S \to T)_{\text{ECEF}}} = \mathbf{T} \begin{bmatrix} E \\ N \\ U \end{bmatrix}_k = \mathbf{T}\mathbf{x}_{k_1} \tag{36}$$

where $\mathbf{T}$ is the rotation matrix from the ENU reference frame to the ECEF reference frame, as defined in the Appendix.

The absolute position of the target in ECEF coordinates is then calculated as follows:

$$\begin{bmatrix} X \\ Y \\ Z \end{bmatrix}_{T_{\text{ECEF}}} = \begin{bmatrix} X \\ Y \\ Z \end{bmatrix}_{\text{ECEF}} + \begin{bmatrix} X \\ Y \\ Z \end{bmatrix}_{(S \to T)_{\text{ECEF}}} \tag{37}$$

As the last step, this absolute target position is converted back to WGS-84 (i.e. latitude, longitude and altitude) via the reverse transformation elaborated in the Appendix.

G. Velocity Update

The velocity components of the target, which are the second half of the corrected state vector $\mathbf{x}_{k_2}$ are defined with respect to the ownship ENU reference frame, these are transformed to the target ENU reference frame as follows:

$$\begin{bmatrix} \dot{E} \\ \dot{N} \\ \dot{U} \end{bmatrix}_{k_T} = \mathbf{L}^\top \begin{bmatrix} \dot{E} \\ \dot{N} \\ \dot{U} \end{bmatrix}_k = \mathbf{L}^\top \mathbf{x}_{k_2} \tag{38}$$

H. State Error Covariance Update $\mathbf{P}_k$

By first defining:

$$\mathbf{W}_H = \mathbf{I}_{6\times 6} - \mathbf{W}_{\text{valid}}\mathbf{H} \tag{39}$$

where $\mathbf{I}_{6\times 6}$ is a 6-by-6 identity matrix, the state error covariance matrix is then updated in the ownship reference frame:

$$\mathbf{P}_k = \mathbf{W}_H \hat{\mathbf{P}}_k \mathbf{W}_H^\top + \mathbf{W}_{\text{valid}} \mathbf{R}_{\text{sensors}} \mathbf{W}_{\text{valid}}^\top \tag{40}$$

which is finally converted to the target reference frame via the aforementioned rotation matrix $\mathbf{L}$:

$$\mathbf{P}_{k_T} = \begin{bmatrix} \mathbf{L} & \mathbf{0}_{3\times 3} \\ \mathbf{0}_{3\times 3} & \mathbf{L} \end{bmatrix} \mathbf{P}_k \tag{41}$$

VI. Select Results

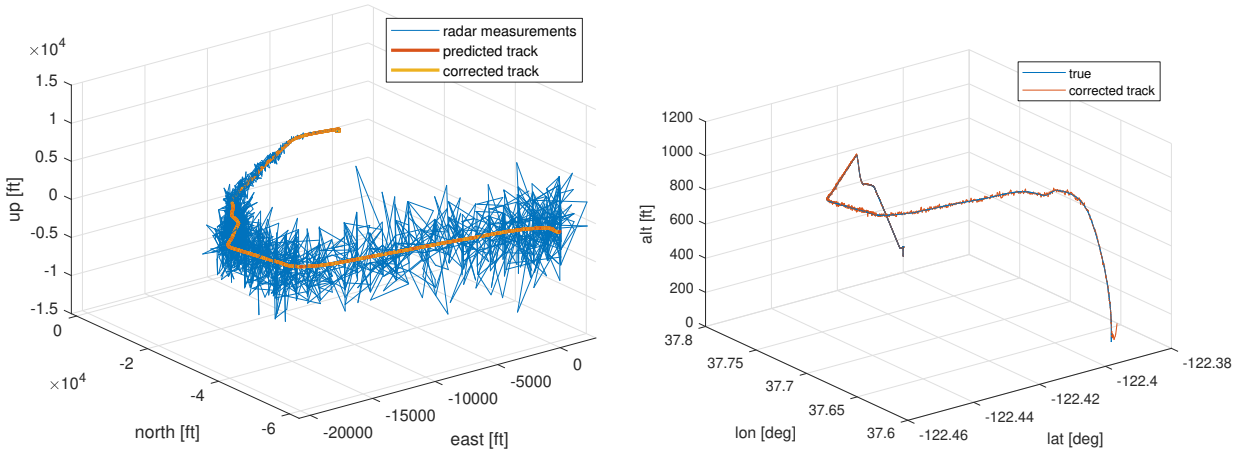
A selection of results is presented here, consisting of four different configurations: first a nominal configuration with default sensors is shown for scenario 1. Thereafter occasional dropouts are considered for the EO camera as well as for the radar, both also for scenario 1. Lastly, nominal results are presented with processed image data as elaborated in Section IV, applied on scenario 2.

A. Nominal Results

First a nominal configuration is considered with default fully functioning sensors, applied for scenario 1 as described in Section II.A. This specific example focuses on the situation where Duck is tracking Goose. Fig. 7(a) compares the estimated tracks (predicted as well as corrected) with the noisy radar measurement. This shows that most radar noise is along the vertical axis, caused by the largest standard deviation for elevation noise (see Table 2) combined with the long distance between both vehicles. This noise magnitude becomes significantly smaller when the vehicles get closer to each other, as shown close to the origin of Fig. 7(a), which is the landing spot. Fig. 7(b) compares the corrected track with the true reference (which is unknown to the object tracker). This figure shows that the filter effectively removes much of the radar noise thanks to the addition of the less noisy EO camera data.

Table 2 Noise σ values for the sensors

noise σ	EO camera	radar
range	–	$\pm 3.25m$
range rate	–	$\pm 2m/s$
elevation	$0.0094deg$	$\pm$**3 *deg***
azimuth	$0.0094deg$	$\pm 1deg$



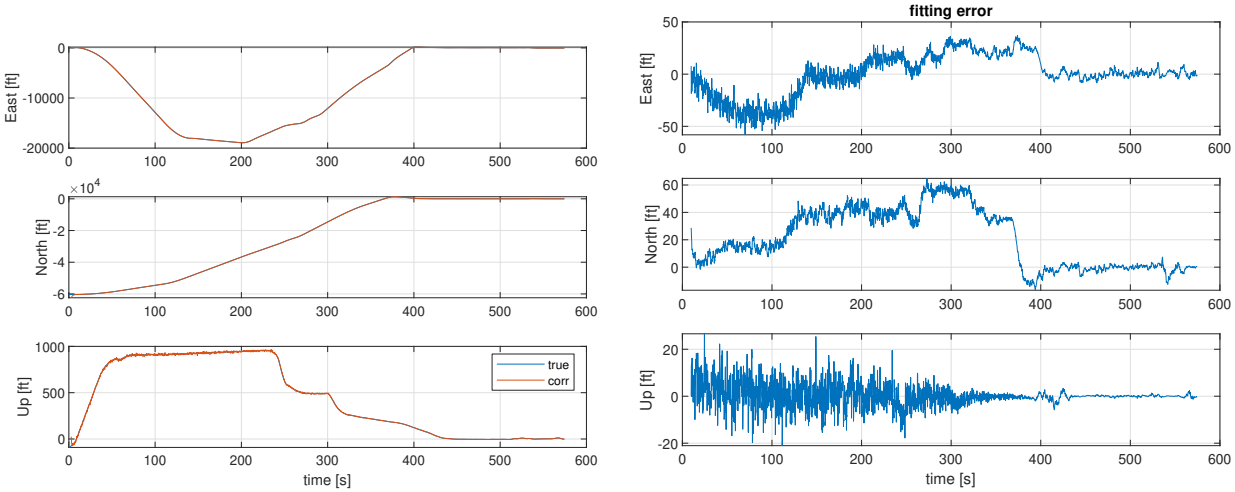
(a) Estimated track (predicted and corrected) with noisy radar measurement

(b) Estimated track with true reference

Fig. 7 Estimated track in three dimensions for nominal scenario 1

Fig. 8(a) compares the estimated track in East–North–Up coordinates, which is the first half of the estimated state vector $\mathbf{x}_{k_1}$, with the true track. The estimate is a good fit, and the fitting errors in each dimension, as shown in Fig. 8(b), are small. These fitting errors are also small when compared to the actual noise in the radar data, as shown side-by-side in Fig. 9.

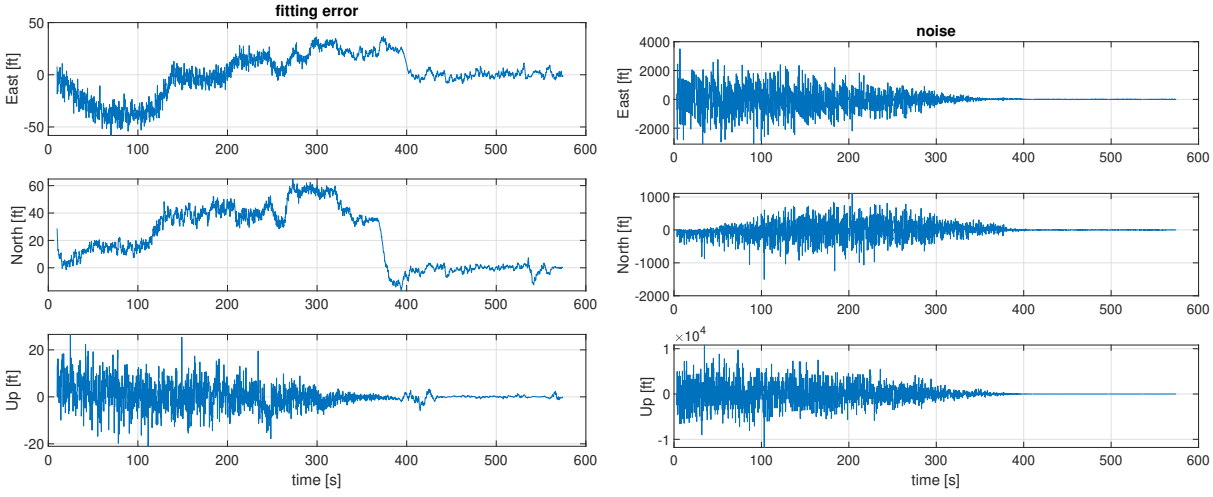
Fig. 10(a) compares the estimated velocity components in East–North–Up coordinates, which is the second half of the estimated state vector $\mathbf{x}_{k_2}$, with the true velocity components. The estimated velocity components are significantly noisier, especially in the vertical dimension. Fig. 10(b) shows the standard deviations for respectively the predicted



(a) Estimated track East – North – Up

(b) Fitting error of estimated track

Fig. 8 Estimated track time histories for nominal scenario 1

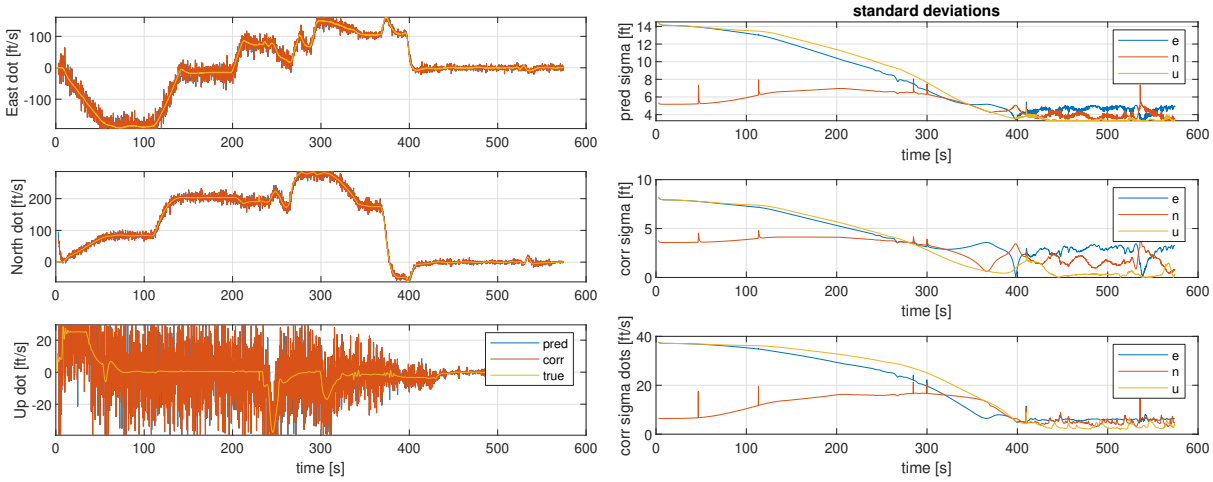


(a) Fitting error of estimated track

(b) Actual radar noise added to track

Fig. 9 Fitting error and noise levels for nominal scenario 1

track components, the corrected track components and the corrected velocity components. A few observations here are that the standard deviations are lowered from the prediction to the correction step, as extra sensor information is coming in, and the standard deviations have the largest magnitude for the velocity components, as expected. Larger sigma values in the prediction step give more weight to the correction step.

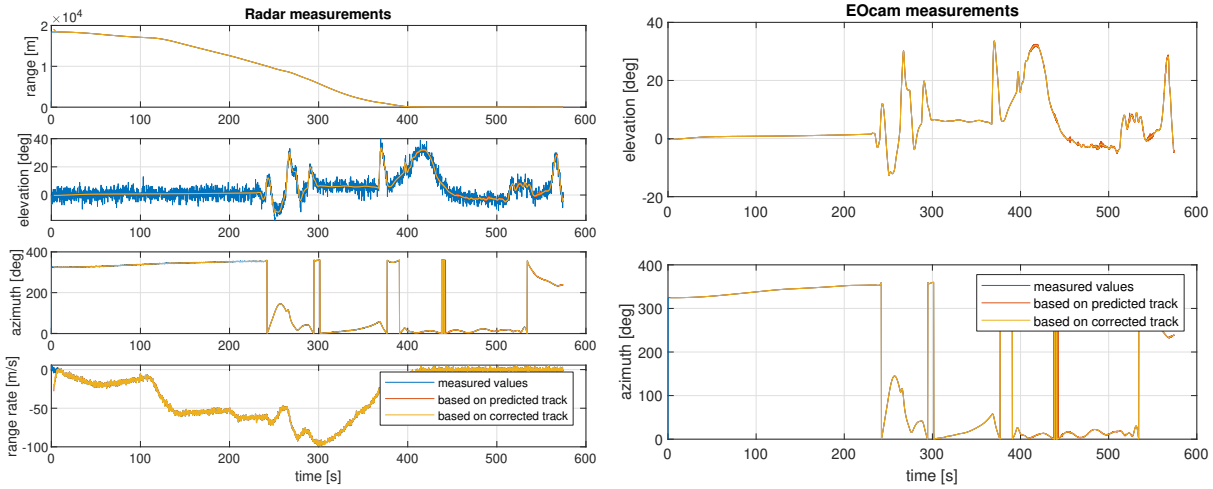


(a) Estimated velocity East – North – Up

(b) Standard deviations for track and speeds

Fig. 10 Estimated velocity and standard deviations for nominal scenario 1

Fig. 10 compare the actual measurements with the state-based calculated measurements. The predicted measurement is calculated based on the predicted state based on the previously defined function $\hat{\mathbf{z}} = \mathbf{h}(\hat{\mathbf{x}})$. The difference between actual measurement and predicted measurement is the innovation: $\varepsilon = \mathbf{z} - \hat{\mathbf{z}}$. The corrected measurement is calculated based on the corrected state in a similar fashion. Fig. 11(a) shows the results for the radar measurements and Fig. 11(b) for the EO camera observations. Once again, these graphs show a good fit, and will serve as baseline for the next two examples with sensor dropouts. Fig. 11(a) also shows the much noisier elevation measurement by radar, but a significantly less noisy reconstructed measurement based on the estimated states.



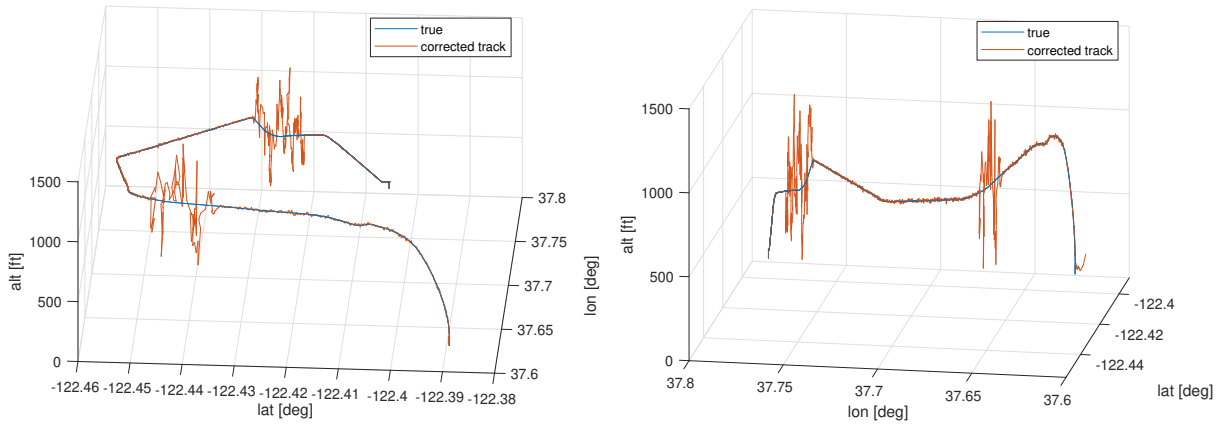
(a) Comparison radar measurements with state calculated measurements

(b) Comparison EO camera measurements with state calculated measurements

Fig. 11 Comparison of measurements with state calculated measurements for nominal scenario 1

B. Results for Occasional EO Camera Dropouts

This example is set up similarly as the previous nominal one for scenario 1, but here the camera fails for two short time spans during the simulated flight, first for 20 seconds, thereafter for 50 seconds. Fig. 12 shows the estimated track and compares it with the true reference, as seen from two different vantage points, seen from the South and from the West respectively. These figures show predominantly vertical deviations in the estimated position, which are mainly caused by the largest noise magnitude in the radar elevation measurements, as shown in Table 2.

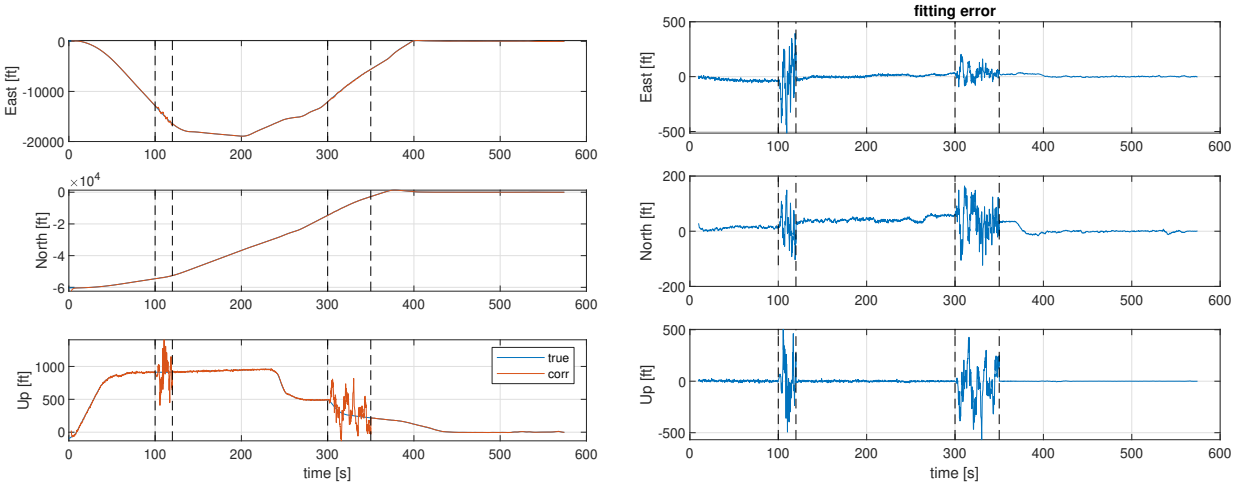


(a) Estimated track with true reference, seen from the South

(b) Estimated track with true reference, seen from the West

Fig. 12 Estimated track in three dimensions for scenario 1 with occasional EO camera dropouts

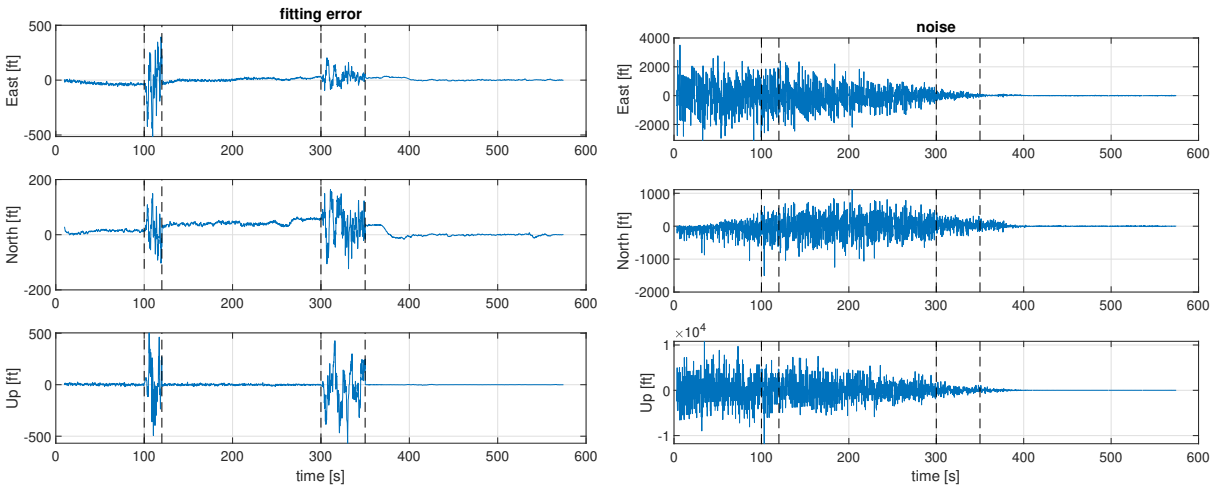
Fig. 13(a) compares the estimated track in East–North–Up coordinates with the true track, and the fitting errors in each dimension are shown in Fig. 13(b). The deviations are in all axes, but they are mostly noticeable along vertical axis due to the magnitude scales, which is consistent with the previous observations. Despite the loss of camera information, these fitting errors are still relatively small when compared to the actual noise in the radar data, as shown side-by-side in Fig. 14.



(a) Estimated track East – North – Up

(b) Fitting error of estimated track

Fig. 13 Estimated track time histories for scenario 1 with occasional EO camera dropouts

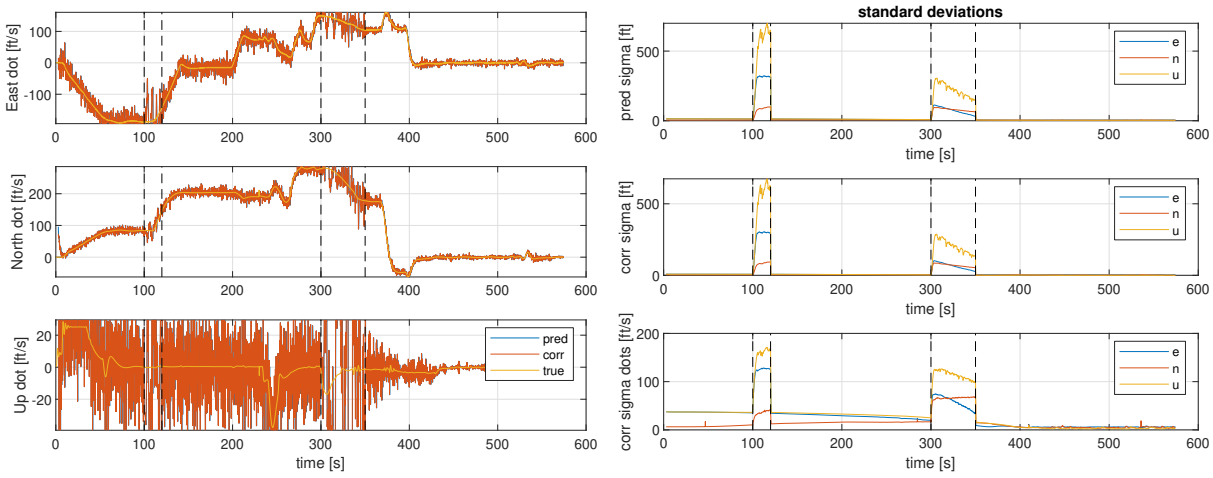


(a) Fitting error of estimated track

(b) Actual radar noise added to track

Fig. 14 Fitting error and noise levels for scenario 1 with occasional EO camera dropouts

Fig. 15(a) compares the estimated velocity components with the true velocity components. The differences are noticeable when the camera fails, again especially in the vertical dimension. Fig. 15(b) shows the standard deviations for position and velocity in a similar way as earlier. All standard deviations increase significantly in the relevant time spans, again the most in the upward direction.

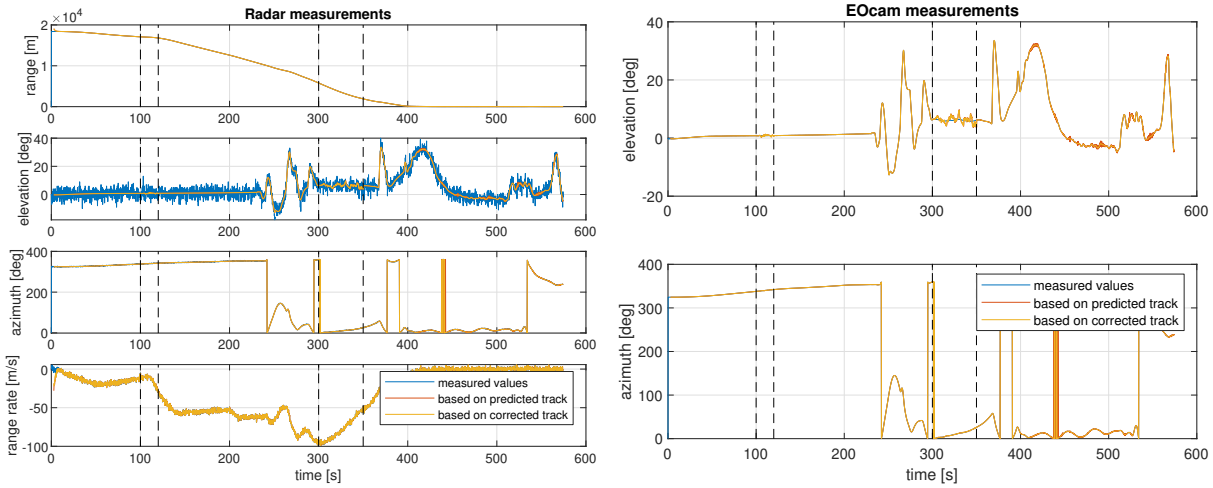


(a) Estimated velocity East – North – Up

(b) Standard deviations for track and speeds

Fig. 15 Estimated velocity and standard deviations for scenario 1 with occasional EO camera dropouts

An important observation for the reconstructed measurements in Fig. 16 is for the reconstructed elevation signal for the EO camera during the dropout timespans. During these timespans, the reconstructed camera measurement is much noisier, since it is reconstructed based on states that are estimated with noisy radar measurements only. This noisy deviation is less apparent for longer distances.

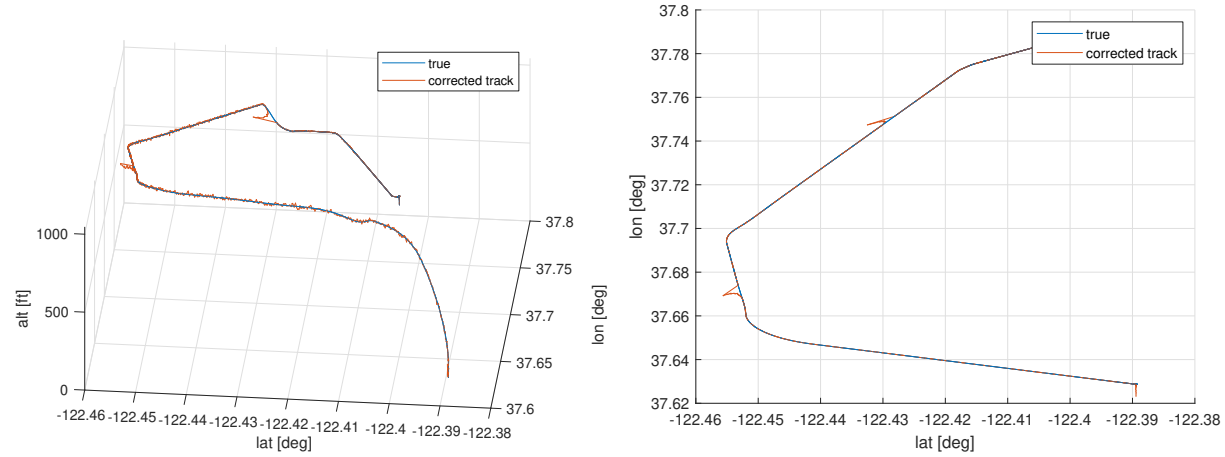


(a) Comparison radar measurements with state calculated measurements (b) Comparison EO camera measurements with state calculated measurements

Fig. 16 Comparison of measurements with state calculated measurements for scenario 1 with occasional EO camera dropouts

C. Results for Occasional Radar Dropouts

This example is similar as the previous two, but here the radar fails for two short time spans, namely first 15 seconds, thereafter another 10 seconds. The estimated track is shown in Fig. 17. There is no substitute in the camera signals for the range and range rate measurements in the radar observations, highlighted in Table 3, and due to the lack of information the track starts to drift along the line of sight, as illustrated in Fig. 17. During the first time span, Duck is still on the ground, thus the drift is tilted upward. Duck is in flight for the second timespan, which results in a less upward deviation. The correction happens immediately after the first new radar measurement is received.



(a) Estimated track with true reference, seen from the South (b) Estimated track with true reference, seen from above

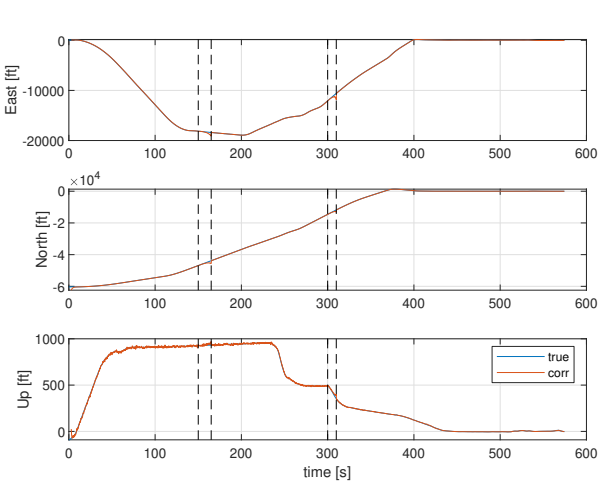
Fig. 17 Estimated track in three dimensions for scenario 1 with occasional radar dropouts

Fig. 18(a) compares the estimated track in East–North–Up coordinates with the true track, and the fitting errors in each dimension are shown in Fig. 18(b). Initially, the drift builds up very slowly, but increases exponentially over time. The fitting error is huge compared to the noise, but there is fundamental information missing here.

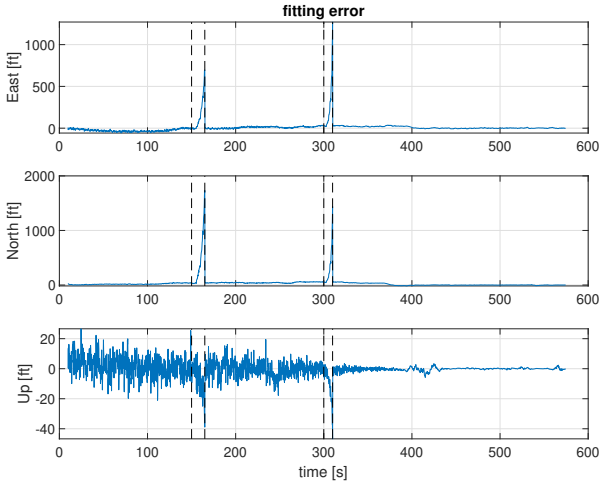
Fig. 19(a) compares the estimated velocity components with the true velocity components. The differences are

Table 3 Measuremed quantities for the sensors

variable	EO camera	radar
range	✗	✓
range rate	✗	✓
elevation	✓	✓
azimuth	✓	✓



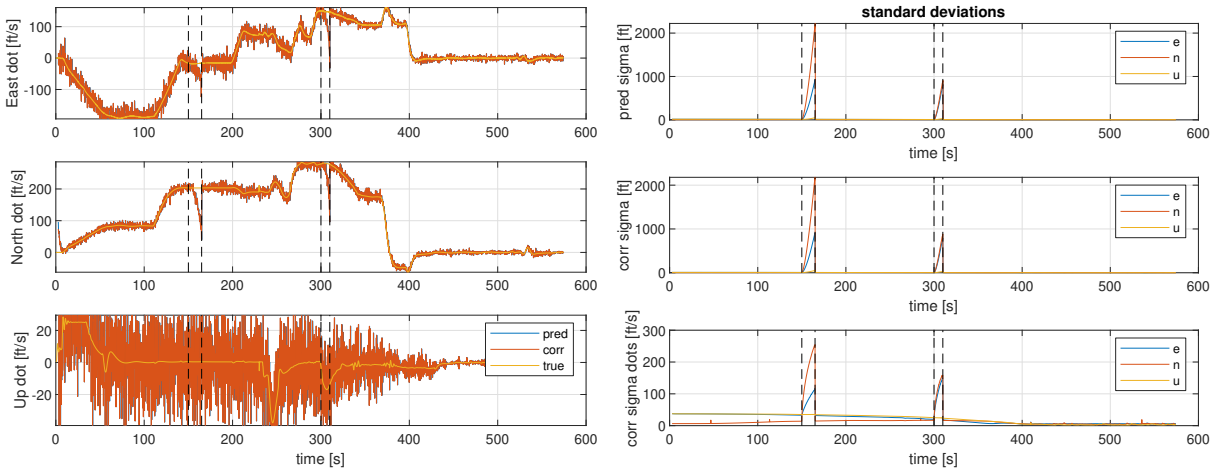
(a) Estimated track East – North – Up



(b) Fitting error of estimated track

Fig. 18 Estimated track time histories for scenario 1 with occasional radar dropouts

noticeable when the radar fails, in all dimensions. Fig. 19(b) shows the standard deviations for position and velocity in a similar way as earlier. All standard deviations increase significantly in the relevant time spans, all observations here are consistent with previous trends.

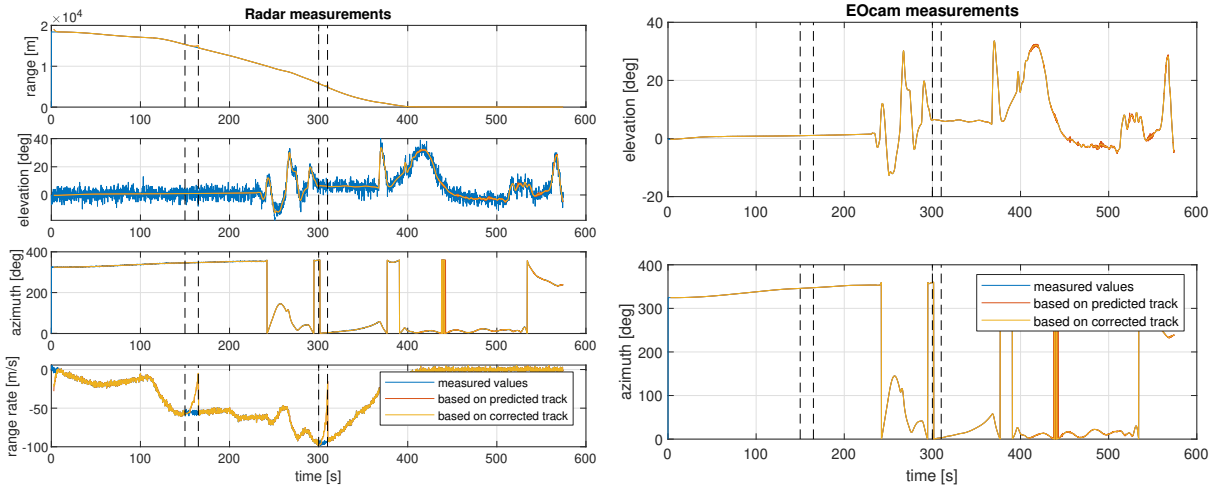


(a) Estimated velocity East – North – Up

(b) Standard deviations for track and speeds

Fig. 19 Estimated velocity and standard deviations for scenario 1 with occasional radar dropouts

An important observation for the reconstructed measurements in Fig. 20 is for the reconstructed range and range rate signals for the radar during the dropout timespans. During these timespans, the reconstructed range and range rate values drift away exponentially, since there is some fundamental information missing here. These trends confirm the earlier observation that the drift in range and range rate is along the line of sight.



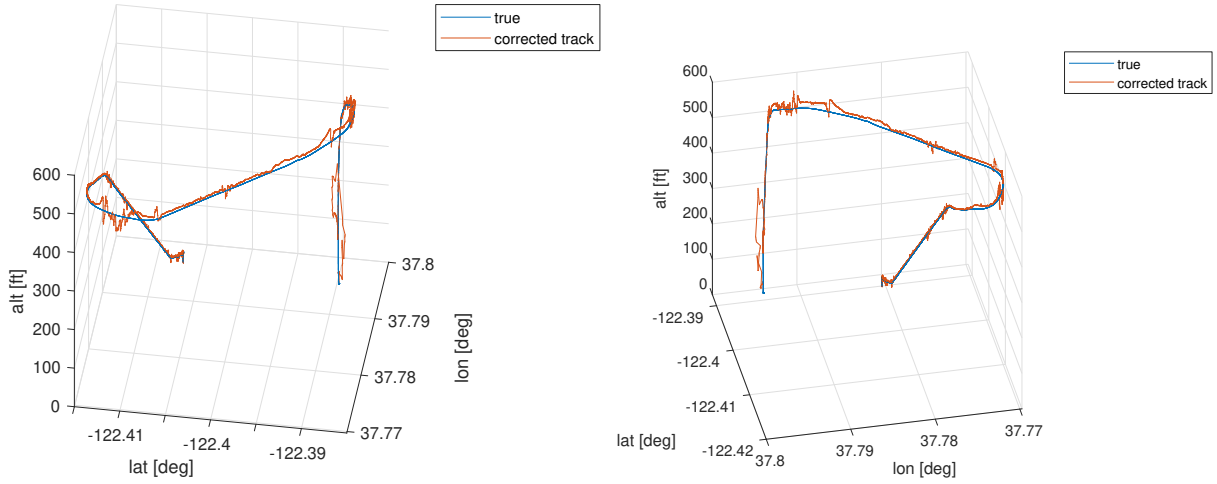
(a) Comparison radar measurements with state calculated measurements

(b) Comparison EO camera measurements with state calculated measurements

Fig. 20 Comparison of measurements with state calculated measurements for scenario 1 with occasional radar dropouts

D. Nominal Results with processed image data

This example is based on scenario 2, where the camera information consists of processed image data, as was explained in Section IV. For this example, the camera field of view limits were taken into account, which were ±18deg for the elevation and ±30deg for the azimuth. The image processing software occasionally also missed to detect Goose for a few instances while inside the field of view. Just like before, the biggest noise level occurs in the radar elevation signal as previously shown in Table 2, combined with a longer range this causes primarily vertical deviations, as shown in Fig. 21.

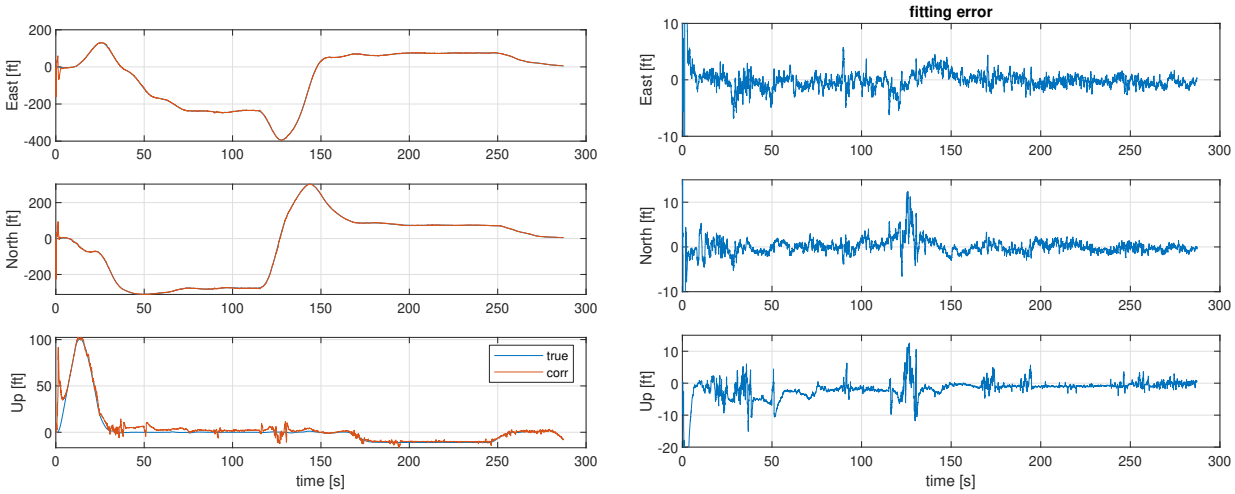


(a) Estimated track with true reference, seen from the South

(b) Estimated track with true reference, seen from the West

Fig. 21 Estimated track in three dimensions for scenario 1 with processed image data

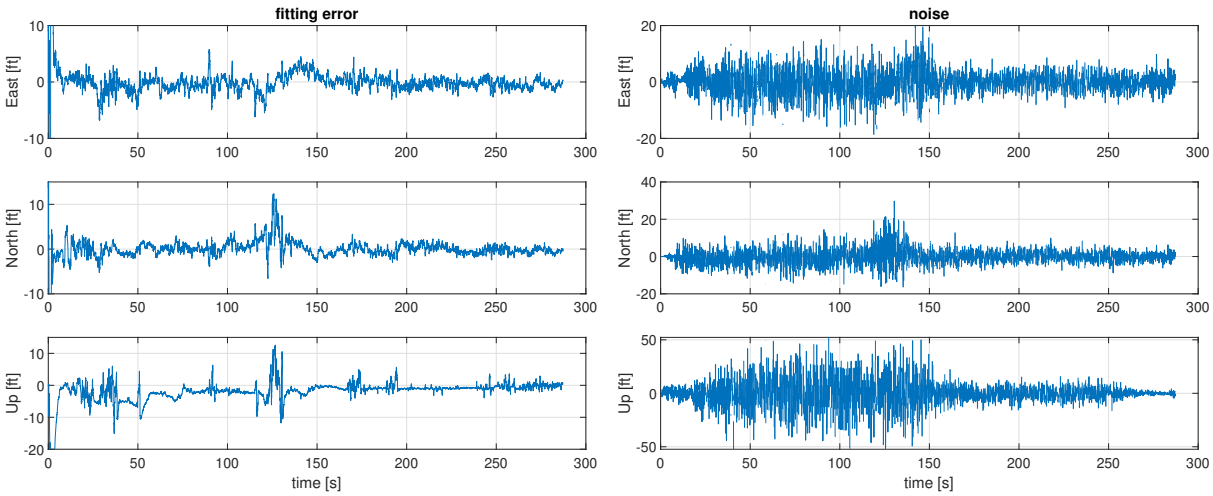
Fig. 22(a) compares the estimated track in East–North–Up coordinates with the true track, and the fitting errors in each dimension are shown in Fig. 22(b). The deviations are in all axes, but they are mostly noticeable along the vertical axis due to the magnitude scales (lower signal to noise ratio). Despite the loss of camera information, these fitting errors are still significantly smaller when compared to the actual noise in the radar data, as shown side-by-side in Fig. 23.



(a) Estimated track East – North – Up

(b) Fitting error of estimated track

Fig. 22 Estimated track time histories for scenario 1 with processed image data

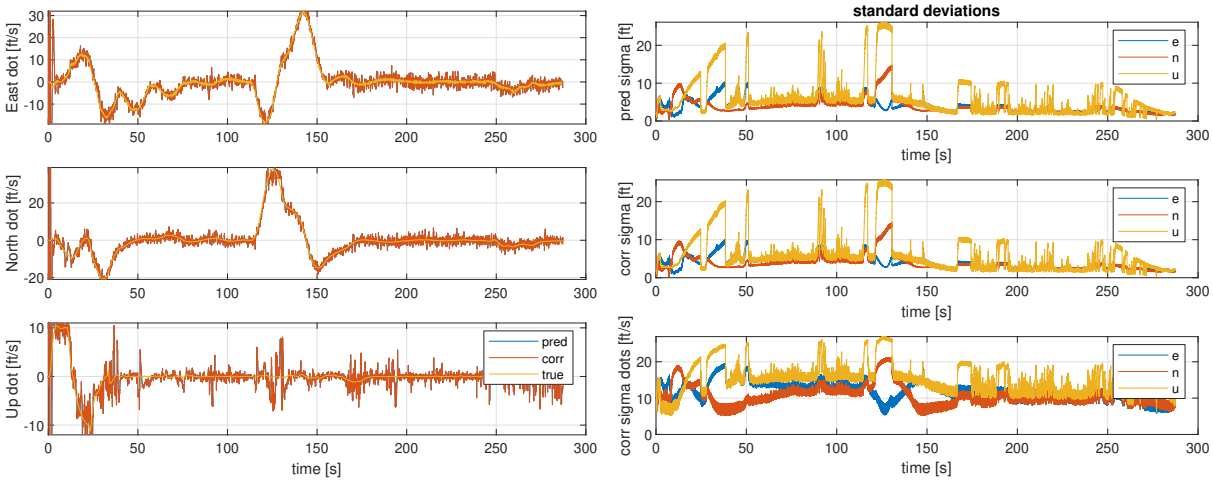


(a) Fitting error of estimated track

(b) Actual radar noise added to track

Fig. 23 Fitting error and noise levels for scenario 1 with processed image data

Also for the speed components as shown in Fig. 24(a), the differences are noticeable when the camera fails, again especially vertically. Also worth noting is that the estimated velocity component up seems significantly less noisy here. However, an important sidenote here is that the maximum distance between the two vehicles in scenario 2, considered here, is considerably less than the distance between both during most of scenario 1. The standard deviations in Fig. 24(b) also increase in the relevant time spans when no image processing data are available. The 'dropout' timespans are most straightforward to distinguish in these standard deviations time histories, and are consistent with the other findings in the previous figures of this example.



(a) Estimated velocity East – North – Up

(b) Standard deviations for track and speeds

Fig. 24 Estimated velocity and standard deviations for scenario 1 with processed image data

VII. Conclusions and recommendations

In this research, a baseline Kalman Filter based object tracker was developed in the context of Urban/Advanced Air Mobility, where a capability is needed such that eVTOL vehicles can track each other as well as non-cooperative objects (drones, birds) based on on-board sensors only. The sensors considered in this baseline approach are EO camera and radar. Both have their specific advantages and drawbacks. Radar measurements are notably more noisy than camera observations. However, camera measurements provide azimuth and elevation only, thus lacking range information. This sensor fusion strategy is made adaptive, so that lower sampling rates of specific sensors are taken into account, as well as loss of signal (for example because the object is outside the field of view or the useful range for a particular sensor, or because of a sensor malfunction resulting in a sensor signal dropout), and invalid data. This capability is achieved by incorporating valid flags in the weighting matrix of the Kalman filter. Initial results shown in this research confirm satisfactory performance, which is negatively impacted by a loss of signal of one of the sensors, in the way as one would expect. Loss of camera information results in a more noisy estimate, especially in the vertical plane. A momentary drop in radar data results in drift of the estimate, due to the lack of range information.

Subsequent research in this project is in the field of distributed sensing, where multiple vehicles as well as ground-based observers (such as radars) with similar as well as dissimilar sensor suites exchange sensor data and/or track estimates, in order to circumvent or minimize the negative impact of the aforementioned loss of signal of one or multiple on-board sensors.

Acknowledgments

This work was performed under the NASA Aeronautics Research Mission Directorate (ARMD), Transformative Tools and Technologies (TTT) Project. The authors would like to thank Caleb Adams, Keerthana Kannan and George Gorospe for their invaluable contributions to this project. Moreover, a special acknowledgment to Chester Dolph and Evan Kawamura for the good collaboration in combining our individual research work for moving forward to the next steps of object tracking using real sensor data and Kalman filtering applied for vision based landing guidance.

Appendix: Reference Frame Transformations for Kalman Filter

Throughout this publication, five different reference frames are used: geodetic (WGS-84), Earth-Centered Earth-Fixed (ECEF), East–North–Up (ENU), body fixed and sensor spherical reference frames. They are all illustrated in Fig. 25.

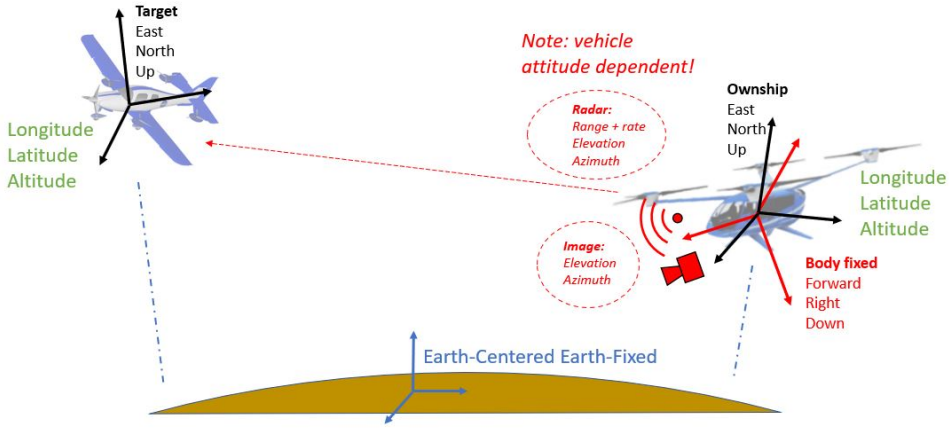


Fig. 25 Reference frames used by Kalman Filter

The geodetic reference frame expresses the spherical coordinates as longitude λ_s with respect to the Greenwich meridian, latitude ϕ_s with respect to the equator and altitude h_s above the average sea-level. The ECEF reference frame is Cartesian with the origin in the Earth center and its coordinates are as follows: the Z_{ECEF} axis extends through true North, X_{ECEF} points through the earth surface where the equator and Greenwich meridian intersect, and Y_{ECEF} is finally perpendicular on the two previous axes. The ENU Cartesian reference frame is vehicle centered with the axes

pointing East–North–Up. As illustrated in Fig. 25, the 'up' direction is location dependent, as well as the others. As a consequence, the ENU frames for ownship and target are differently oriented. This difference increases for increasing distances between both. This must be reflected in the calculations. Next is the body fixed reference frame, which is Cartesian and centered at the vehicle's center of gravity. The X_B axis points forward, Y_B axis points to the right and the Z_B axis is downwards. Finally, the vehicle centered spherical reference frame expresses the coordinates as range ρ (distance from center of gravity), elevation α (vertical angle with respect to the $X_B - Y_B$-plane) and azimuth γ (horizontal angle with respect to the forward axis). The sensor measurements from radar and camera are expressed in this spherical reference frame. Note that both latter vehicle fixed reference frames are vehicle attitude dependent.

The transformations between the different aforementioned reference frames are illustrated in Fig. 26. The transformations between body-fixed, ENU, ECEF as well as between different ENU's of different vehicles at different locations are pure angular rotations by means of the respective rotation matrices $\mathbf{M}(\phi, \theta, \psi)$ over the three Euler angles between body fixed and ENU, $\mathbf{T}(\lambda_s, \phi_s)$ over longitude and latitude between ENU and ECEF and $\mathbf{L}(\lambda_s, \phi_s, \lambda_t, \phi_t)$ over longitude and latitude of both locations between different ENU's at different locations. The mappings between body fixed and sensor spherical as well as between geodetic and ECEF are more complex nonlinear transformations. Each of them is elaborated next.

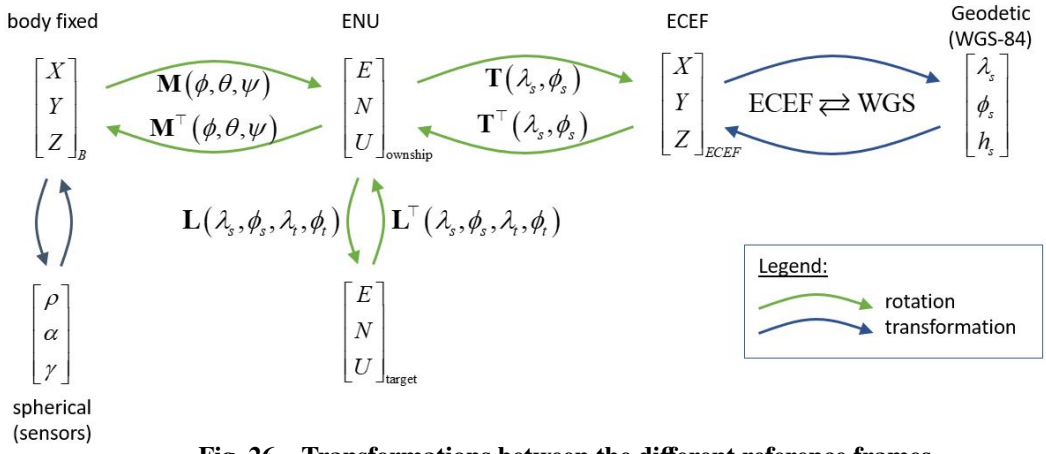


Fig. 26 Transformations between the different reference frames

Transformation from body-fixed to spherical and backwards

The sensor spherical coordinates are defined as a function of the body-fixed coordinates as follows:

$$\rho = \sqrt{X_B^2 + Y_B^2 + Z_B^2} \tag{42}$$

$$\alpha = \arctan \frac{-Z_B}{\rho_{\text{hor}}} \tag{43}$$

$$\gamma = \arctan \frac{Y_B}{X_B} \tag{44}$$

$$\tag{45}$$

where:

$$\rho_{\text{hor}} = \sqrt{X_B^2 + Y_B^2} \tag{46}$$

And the other way around:

$$X_B = \rho \cos\alpha \cos\gamma \tag{47}$$

$$Y_B = \rho \cos\alpha \sin\gamma \tag{48}$$

$$Z_B = \rho \sin\alpha \tag{49}$$

Rotation from body-fixed to ENU and backwards

From body-fixed to ENU is a sequential rotation over the three Euler angles roll ϕ, pitch attitude θ and yaw ψ[17]:

$$\mathbf{M}(\phi,\theta,\psi) = \begin{bmatrix} \sin\psi\cos\theta & \cos\phi\cos\psi + \sin\phi\sin\psi\sin\theta & -\sin\phi\cos\psi + \cos\phi\sin\psi\sin\theta \\ \cos\psi\cos\theta & -\cos\phi\sin\psi + \sin\phi\cos\psi\sin\theta & \sin\phi\sin\psi + \cos\phi\cos\psi\sin\theta \\ \sin\theta & -\sin\phi\cos\theta & -\cos\phi\cos\theta \end{bmatrix} \tag{50}$$

The transpose is used for the backwards direction.

Rotation from ENU to ECEF and backwards

From the vehicle centered ENU to the earth centered ECEF is a sequential rotation over the longitude λ_s and latitude ϕ_s of the current position[17]:

$$\mathbf{T}(\lambda_s,\phi_s) = \begin{bmatrix} -\sin\lambda_s & -\cos\lambda_s\sin\phi_s & \cos\lambda_s\cos\phi_s \\ \cos\lambda_s & -\sin\lambda_s\sin\phi_s & \sin\lambda_s\cos\phi_s \\ 0 & \cos\phi_s & \sin\phi_s \end{bmatrix} \tag{51}$$

The transpose is used for the other way around.

Rotation between ENU's

A rotation between ENU's is basically a rotation from the first ENU to ECEF and subsequently from ECEF back to the second ENU:

$$\mathbf{L}(\lambda_s,\phi_s,\lambda_t,\phi_t) = \mathbf{T}^\top(\lambda_t,\phi_t)\cdot\mathbf{T}(\lambda_s,\phi_s) \tag{52}$$

Substituting Eq. (51) results in:

$$\begin{aligned} \mathbf{L} \quad & (\lambda_s,\phi_s,\lambda_t,\phi_t) = \\ = \quad & \mathbf{T}^\top(\lambda_t,\phi_t)\cdot\mathbf{T}(\lambda_s,\phi_s) \\ = \quad & \begin{bmatrix} -\sin\lambda_t & -\cos\lambda_t\sin\phi_t & \cos\lambda_t\cos\phi_t \\ \cos\lambda_t & -\sin\lambda_t\sin\phi_t & \sin\lambda_t\cos\phi_t \\ 0 & \cos\phi_t & \sin\phi_t \end{bmatrix}^T \begin{bmatrix} -\sin\lambda_s & -\cos\lambda_s\sin\phi_s & \cos\lambda_s\cos\phi_s \\ \cos\lambda_s & -\sin\lambda_s\sin\phi_s & \sin\lambda_s\cos\phi_s \\ 0 & \cos\phi_s & \sin\phi_s \end{bmatrix} \\ = \quad & \begin{bmatrix} \cos(\lambda_t-\lambda_s) & \sin(\lambda_t-\lambda_s)\sin\phi_s & -\sin(\lambda_t-\lambda_s)\cos\phi_s \\ -\sin(\lambda_t-\lambda_s)\sin\phi_t & \cos\phi_t\cos\phi_s + \sin\phi_t\sin\phi_s\cos(\lambda_t-\lambda_s) & \cos\phi_t\sin\phi_s - \sin\phi_t\cos\phi_s\cos(\lambda_t-\lambda_s) \\ \sin(\lambda_t-\lambda_s)\cos\phi_t & \sin\phi_t\cos\phi_s - \cos\phi_t\sin\phi_s\cos(\lambda_t-\lambda_s) & \sin\phi_t\sin\phi_s + \cos\phi_t\cos\phi_s\cos(\lambda_t-\lambda_s) \end{bmatrix} \end{aligned} \tag{53}$$

Transformation from ECEF to geodetic and backwards

ECEF $\Rightarrow$ geodetic

The target position in WGS-84 geodetic coordinates is obtained following the 15-step process presented below (extracted from RTCA DO-317A)[4]:

$$r_{\text{ecef}} = \sqrt{X_t^2 + {Y_t}^2} \tag{54}$$

$$E^2 = a^2 - b^2 \tag{55}$$

$$F = 54b^2Z_t^2 \tag{56}$$

$$G = r_{\text{ecef}}^2 + \left(1-\varepsilon^2\right)Z_t^2 - \varepsilon^2E^2 \tag{57}$$

$$C = \frac{\varepsilon^4 F r_{\text{ecef}}^2}{G^3} \tag{58}$$

$$S = \sqrt[3]{1 + C + \sqrt{C^2 + 2C}} \tag{59}$$

$$P = \frac{F}{3\left(S+\frac{1}{S}+1\right)^2 G^2} \tag{60}$$

$$Q = \sqrt{1+2\varepsilon^4 P} \tag{61}$$

$$r_o = \frac{-P\varepsilon^2 r_{\text{ecef}}}{1+Q} + \sqrt{\frac{a^2}{2}\left(1+\frac{1}{Q}\right) - \frac{P\left(1-\varepsilon^2\right)Z_t^2}{Q(1+Q)} - \frac{Pr_{\text{ecef}}^2}{2}} \tag{62}$$

$$U = \sqrt{\left(r_{\text{ecef}} - \varepsilon^2 r_o\right)^2 + Z_t^2} \tag{63}$$

$$V = \sqrt{\left(r_{\text{ecef}} - \varepsilon^2 r_o\right)^2 + \left(1-\varepsilon^2\right)Z_t^2} \tag{64}$$

$$Z_o = \frac{b^2 Z_t}{aV} \tag{65}$$

$$\phi_t = \tan^{-1}\left(\frac{Z_t + \varepsilon'^2 Z_o}{r_{\text{ecef}}}\right) \tag{66}$$

$$\lambda_t = \tan^{-1}\left(\frac{Y_t}{X_t}\right) \tag{67}$$

$$h_t = U\left(1-\frac{b^2}{aV}\right) \tag{68}$$

where:
$a = 6.378137 \cdot 10^6 m$, semimajor axis of the Earth
$b = 6.3567523142 \cdot 10^6 m$, semiminor axis of the Earth
$\varepsilon = 0.081819190842622$, first eccentricity of the Earth
$\varepsilon' = 0.082094437935831$, second eccentricity of the Earth
ϕ_t = target latitude [rad]
λ_t = target longitude [rad]
h_t = target geodetic altitude [m]

geodetic $\Rightarrow$ *ECEF*

The ownship's Earth-Centered, Earth-Fixed (ECEF) position is calculated by[4]:

$$\begin{bmatrix} X_S \\ Y_s \\ Z_s \end{bmatrix} = \begin{bmatrix} (N_s+h_s)\cos\phi_s\cos\lambda_s \\ (N_s+h_s)\cos\phi_s\sin\lambda_s \\ \left(N_s\left(1-\varepsilon^2\right)+h_s\right)\sin\phi_s \end{bmatrix} \tag{69}$$

where:
ϕ_s = ownship geodetic latitude [rad]
λ_s = ownship geodetic longitude [rad]
h_s = ownship geodetic altitude [m]
$N_s = \frac{a}{\sqrt{1-\varepsilon^2\sin^2\phi_s}}$
$a = 6.378137 \cdot 10^6 m$, semimajor axis of the Earth
$\varepsilon = 0.081819190842622$, first eccentricity of the Earth
If the altitude of the ownship is measured as height above Mean Sea Level (MSL) from a barometric sensor, it should be converted to WGS-84 Height Above Ellipsoid (HAE) using the following steps:

$$\xi(\phi_s) = g_{0_{\text{eq}}}\frac{1+k\sin^2\phi_s}{\sqrt{1-\varepsilon^2\sin^2\phi_s}} \tag{70}$$

$$R(\phi_s) = \frac{a}{1+f+m_r-2f\sin^2\phi_s} \tag{71}$$

$$h_S = \frac{R(\phi_s)^h}{\frac{R(\phi_s)\xi(\phi_s)}{\xi\left(\frac{\pi}{4}\right)} - h} \tag{72}$$

where:
ϕ_s = ownship geodetic latitude [rad]
$\varepsilon = 0.081819190842622$, first eccentricity of the Earth
$g_{0_{\text{eq}}} = 9.7803253359 m/s^2$, theoretical gravity at the equator
$g_{0_{\text{pl}}} = 9.8321849378 m/s^2$, theoretical gravity at the poles
$k = \frac{b \cdot g_{0_{\text{pl}}}}{a \cdot g_{0_{\text{eq}}}} - 1$
$\xi\,(\phi_s)$ = normal gravity on the surface of an ellipsoid at $\phi_s [m/s^2]$
$a = 6.378137 \cdot 10^6 m$, semimajor axis of the Earth
$b = 6.3567523142 \cdot 10^6 m$, semiminor axis of the Earth
$f = \frac{a-b}{a}$, flattening of the Earth
$m_r = 0.003449787$, gravity ratio
$R\,(\phi_s)$ = radius of the Earth at ϕ_s [m]
h = measured ownship height above MSL
h_s = ownship geodetic altitude [m]

References

[1] Stoschek, A., "Exploring Sense-and-Avoid Systems for Autonomous Vehicles," Online article, Dec. 2017. URL https://acubed.airbus.com/blog/vahana/exploring-sense-and-avoid-systems-for-autonomous-vehicles/, retrieved February 16, 2021.

[2] Holden, J., and Goel, N., "Uber Elevate: Fast-Forwarding to a Future of On-Demand Urban Air Transportation," techreport, Uber Inc., San Francisco, CA, Oct. 2016. URL https://www.uber.com/elevate.pdf.

[3] Holmes, B., Parker, R., Stanley, D., Garrow, L., Masson, P., and Olcott, J., "NASA Strategic Framework for On-Demand Air Mobility," techreport, National Institute of Aerospace, Jan. 2017. URL http://www.nianet.org/ODM/reports/ODMStrategicFramework-Final170308.pdf, nASA Contract No: NNL13AA08B.

[4] *Minimum Operational Performance Standards for Detect and Avoid Systems*, DO-365, RTCA Inc., 2019, techreport Appendix F: Example Functional Description For Airborne Surveillance Data Processor (ASDP) Subsystem. URL https://standards.globalspec.com/std/14281907/rtca-do-365.

[5] Pancham, A., Tlale, N., and Bright, G., "Literature Review of SLAM and DATMO," *4th Robotics and Mechatronics Conference of South Africa (RobMech 2011)*, CSIR International Conference Center, Pretoria, 2011. URL http://www.robmech.co.za/proceed/ROBMECH2011_Pancham_LiteratureReviewofSLAMandDATMO.pdf.

[6] Blackman, S., "Multiple hypothesis tracking for multiple target tracking," *IEEE Aerospace and Electronic Systems Magazine*, Vol. 19, No. 1, 2004, pp. 5–18. doi:10.1109/maes.2004.1263228.

[7] Petrovskaya, A., and Thrun, S., "Model Based Vehicle Tracking for Autonomous Driving in Urban Environments," *Robotics: Science and Systems IV*, Robotics: Science and Systems Foundation, 2008. doi:10.15607/rss.2008.iv.023.

[8] Fasano, G., Accardo, D., Moccia, A., and Paparone, L., "Airborne Multisensor Tracking for Autonomous Collision Avoidance," *2006 9th International Conference on Information Fusion*, IEEE, 2006. doi:10.1109/icif.2006.301724.

[9] Girao, P., Asvadi, A., Peixoto, P., and Nunes, U., "3D object tracking in driving environment: A short review and a benchmark dataset," *2016 IEEE 19th International Conference on Intelligent Transportation Systems (ITSC)*, IEEE, 2016. doi:10.1109/itsc.2016.7795523.

[10] Urmson, C., Anhalt, J., Bagnell, D., Baker, C., Bittner, R., Clark, M. N., Dolan, J., Duggins, D., Galatali, T., Geyer, C., Gittleman, M., Harbaugh, S., Hebert, M., Howard, T. M., Kolski, S., Kelly, A., Likhachev, M., McNaughton, M., Miller, N., Peterson, K., Pilnick, B., Rajkumar, R., Rybski, P., Salesky, B., Seo, Y.-W., Singh, S., Snider, J., Stentz, A., Whittaker, W. "., Wolkowicki, Z., Ziglar, J., Bae, H., Brown, T., Demitrish, D., Litkouhi, B., Nickolaou, J., Sadekar, V., Zhang, W., Struble, J., Taylor, M., Darms, M., and Ferguson, D., "Autonomous driving in urban environments: Boss and the Urban Challenge," *Journal of Field Robotics*, Vol. 25, No. 8, 2008, pp. 425–466. doi:10.1002/rob.20255.

[11] Stepanyan, V., and Hovakimyan, N., "Visual Tracking of a Maneuvering Target," *Journal of Guidance, Control, and Dynamics*, Vol. 31, No. 1, 2008, pp. 66–80. doi:10.2514/1.29758.

[12] Shish, K. H., Cramer, N. B., Gorospe, G., Lombaerts, T., Stepanyan, V., and Kannan, K., "Survey of Capabilities and Gaps in External Perception Sensors for Autonomous Urban Air Mobility Applications," *AIAA Scitech 2021 Forum*, American Institute of Aeronautics and Astronautics, 2021. doi:10.2514/6.2021-1114.

[13] Redmon, J., Divvala, S., Girshick, R., and Farhadi, A., "You only look once: Unified, real-time object detection," *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 779–788.

[14] Redmon, J., and Farhadi, A., "Yolov3: An incremental improvement," *arXiv preprint arXiv:1804.02767*, 2018.

[15] Ultralytics, "YOLOv3 in PyTorch > ONNX > CoreML > TFLite," `https://github.com/ultralytics/yolov3.git`, 2020.

[16] Lin, T.-Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Dollár, P., and Zitnick, C. L., "Microsoft coco: Common objects in context," *European conference on computer vision*, Springer, 2014, pp. 740–755.

[17] Mohinder, S., Grewal, P., Lawrence, R., Weill, A. P., and Andrews, "Appendix C: Coordinate Transformations," *Global Positioning Systems, Inertial Navigation, and Integration*, John Wiley & Sons, Inc., 2006, pp. 456–501. doi:10.1002/9780470099728.app3.