# An Extendable GNSS Sensor-Fusion Algorithm For Consumer Drone Positioning

Tristan Hasseler, Derek Knowles, *Stanford University*

## BIOGRAPHIES

**Tristan Hasseler** is an MS student in the Department of Aeronautics and Astronautics at Stanford University. He obtained his B.S. in Aeronautics and Astronautics from the University of Washington, Seattle. His research interests include vehicle stability & control systems.

**Derek Knowles** is an MS student in the Department of Mechanical Engineering at Stanford University. He obtained his B.S. in Mechanical Engineering from Brigham Young University. His research interests include autonomous robotic perception and control.

## ABSTRACT

In order to fly safely around obstacles and humans, aerial robots must be able to calculate their position in the environment with extreme precision. We are motivated to achieve a more precise position/geolocation value than what we can achieve with GNSS alone. The goal of this paper is to use raw GNSS psuedoranges combined with the sensor suite available on a standard commercial drone to create as precise a geolocation as possible. We implemented an extended Kalman filter that combined GNSS psuedoranges, barometric pressure, and inertial measurement unit velocities. We compare a consumer drone's proprietary position solution with our calculated position solution using sensor fusion and solely GNSS pseudoranges. We demonstrate the added precision that our sensor fusion position solution gives over using only GNSS psuedoranges.

## 1. INTRODUCTION

Many new technologies are being developed to increase the presence of consumer drones in public spaces. Ongoing proliferation of drones in civilian airspace emphasizes the need for drone operation to be safe and reliable. Of particular interest is accurate and precise positioning. In order for autonomous guidance and anti-collision systems to be effective, each drone must have a reliable and precise real-time position estimate. Drones are increasingly being used by consumers, businesses, and government agencies for purposes such as photography, medical deliveries, poaching prevention, and mapping forest fires. All drones must be equipped with precise positioning capabilities in order for high priority drones (such as a drone delivering blood to a hospital) to safely navigate in an environment cluttered with consumer drones.

While high-end consumer drones (such as those developed by DJI) may posess on-board positioning systems, these systems are proprietary and not publicly accessible. In this project, we develop and test a method for fast and accurate consumer drone positioning using publicly available Global Navigation Satellite System (GNSS) measurements combined with a consumer-grade inertial measurement unit (IMU) sensor and a barometer. We present our algorithm as an open-source, fully-extendable sensor fusion platform.

For GNSS position determination, we leverage both the US-based GPS satellite constellation as well as the Russian GLONASS system. We utilize open-source GNSS logger software developed by Google for Android phones [1]. Using this software, we track and receive raw pseudorange measurements from available satellites, which are input into an Extended Kalman Filter (EKF) to obtain a baseline position solution. Further, we extend our EKF positioning algorithm incorporate the on-board IMU sensor and barometer from our consumer drone platform to augment the pseudorange-based position solution. For this project, we use a DJI Mavic Pro as our sensor platform.

To quantitatively assess our positioning solution, we compare our calculated results with DJI's proprietary solution – a solution which has access to full GNSS data and a high-fidelity 6 degree-of-freedom dynamical model (complete with full access to rate gyros, hardware-in-the-loop control logic, and real-time individual motor PWM readouts). To directly compare position solutions, we perform multiple flight tests with a Google Pixel 3XL phone running GNSS logger software attached directly onto the Mavic body [1]. To this end, we compare our solution (based on GNSS measurements taken by the Pixel) with the on-board Mavic solution in real-time. We conclude that our solution balances accuracy, simplicity, and speed of computation when compared

with the proprietary DJI Mavic solution. Of note, we emphasize that our platform provides satisfactory positioning with a fraction of the sensors and computation power used by the DJI solution. In this regard, we are encouraged for future extensions of this algorithm to address the positioning problem for a wide suite of consumer drones. Lastly, we present preliminary results from a framework which could be used to extend our sensor fusion algorithm to include video odometry from on-board camera systems.

## 2. RELATED WORK

We will briefly mention notable works in the field of aerial navigation with particular emphasis on GNSS sensor fusion.

A popular means of combining measurements from different sensors is to use a Kalman filter [2] or its nonlinear extension called an extended Kalman filter [3]. Position solutions based on GNSS measurements tend to work well in open-sky environments, but are error-prone or fail near tall obstacles or inside buildings. Position estimates based on visual odometry tend to work more reliably than their GNSS counterparts in indoor environments. Lynen et al. used this tradeoff to their advantage by calculating two separate position solutions for an aerial robot [4]. One position solution used GNSS measurements while the other was an EKF that had measurements from an IMU, barometer, and monocular camera being fed into it. Their EKF is supposedly extendable, but since the EKF is designed in the IMU frame, each added sensor must be transformed into the IMU frame in order to be used.

Unscented Kalman filters (UKF) have also been used to provide state estimation in a wide array of circumstances. Shen et al. developed a UKF for a flying vehicle to transition between indoor and outdoor environments [5] . They make use of a GPS receiver, laser range finder, stero cameras, barometer, and IMU. Their method is complex in order to deal with absolute vs. relative measurements, timing delays across sensors, and the need to track past states for visual odometry.

Nemra et al. proposed State-Dependent Riccati Equation (SDRE) nonlinear filtering as a third means to combine heterogeneous sensor measurements [6]. This method is well suited for highly nonlinear systems as it does not have the same linearization errors as Kalman filters do. Their paper demonstrates their SDRE filtering by fusing together measurements from an IMU and GPS receiver.

While the previously stated works make contributions to the accuracy of sensor-fused state estimation, none of them used raw GNSS psuedoranges to calculate their position solutions.

We also note that previous work has been done in the area of consumer drone flight data reduction and analysis. Prastya et al. developed a novel workflow for parsing and interpreting DJI drone flight logs [7]. These logs provide every point of data recorded by the drone (for example, altitude, heading, GPS location, motor outputs, fault modes, temperature, accelerations, etc.). This data has been extremely useful for determining fault and cause of failure during accidents. The data analysis workflow, however, is also widely relevant to our application. Following the method outlined in [7], we can extract the data from the DJI Mavic flight logs and have a direct location solution ”truth” with which to compare our solution.

## 3. PROBLEM FORMULATION

Noting the contributions provided by previous related works, we desired to create a sensor fusion technique that was simple and could be modified to accept raw GNSS psuedoranges as measurements. In the following section, we will explain in detail our approach that fuses raw GNSS pseudoranges, IMU, and barometer measurements. We combined these measurements using an extended Kalman filter in order to create a three degree of freedom state position solution (namely, latitude, longitude, and altitude).

To gauge the accuracy of our developed algorithm with respect to the proprietary DJI solution, we will quantitatively assess our position solution using a set of performance metrics. First, we will calculate the instantaneous latitude, longitude, and altitude errors at a given time-step (with respect to a DJI position truth) as follows:

$$\epsilon_{lat,i} \equiv |\phi_i - \phi_{DJI,i}| \tag{1}$$

$$\epsilon_{long,i} \equiv |\lambda_i - \lambda_{DJI,i}| \tag{2}$$

$$\epsilon_{h,i} \equiv |h_i - h_{DJI,i}| \tag{3}$$

Where $\phi_i, \lambda_i, h_i$ are respectively the latitude, longitude, and altitude components determined by our positioning algorithm at a given time-step $i$, expressed in the standard Earth geodetic reference frame. Further, we define the average latitude, longitude, and altitude positioning errors as follows:

$$\bar{\epsilon}_{lat} \equiv \frac{1}{N} \sum_{i=1}^{N} \epsilon_{lat,i} \tag{4}$$

$$\bar{\epsilon}_{long} \equiv \frac{1}{N} \sum_{i=1}^{N} \epsilon_{long,i} \tag{5}$$

$$\bar{\epsilon}_{h} \equiv \frac{1}{N} \sum_{i=1}^{N} \epsilon_{h,i} \tag{6}$$

Where $N$ is the total number of time-steps.

## 4. APPROACH

When we began developing our position solution, our goal was to develop as simple and precise a solution as possible. To that end, we started by finding a position solution with a single sensor and added sensors as time allowed. We first started by finding the position solution using raw GNSS psuedoranges only. We implemented an extended Kalman filter that used a static motion model for the predict step and the GNSS psuedoranges for the update step. The state vector that we estimate in the extended Kalman filter is $\mu = [x_{ECEF}, y_{ECEF}, z_{ECEF}, t_{bias}]$ where the first three elements of the state are the drone position in the Earth-centered, Earth-fixed coordinate frame and $t_{bias}$ is the clock bias. We initialize the state vector by first computing the weighted least squares approximation and using the result for the initial state vector. Algorithm 1 shows the EKF algorithm that we used in order to calculate a position solution with raw GNSS measurements only. The results section discusses the position plots and errors from a flight test where we implemented this method.

**Algorithm 1** GNSS Only EKF

**Inputs:**
    GNSS pseudoranges
    Satellite positions
**Initialize:**
    $\mathbf{T} \leftarrow$ concatenated and sorted timesteps
**for** $t \in \mathbf{T}$ **do**
    **Predict:**
        $\hat{\mu}_{t|t-1} = F\hat{\mu}_{t-1|t-1}$
        $P_{t|t-1} = F_t P_{t-1|t-1} F_t^T + Q$
    **Update GNSS:**
        $z_t \leftarrow$ GNSS measurements
        $\tilde{y}_t = z_t - h(\hat{\mu}_{t|t-1})$
        $K_t = P_{t|t-1} H_t^T (R + H_t P_{t|t-1} H_t^T)^{-1}$
        $\hat{\mu}_{t|t} = \hat{\mu}_{t|t-1} + K_t \tilde{y}_t$
        $P_{t|t} = (I - K_t H_t) P_{t|t-1} (I - K_t H_t)^T + K_t R K_t^T$
**end for**

After our EKF where we only used GNSS raw measurements, we further improved upon our method by integrating inertial measurement unit (IMU) data from the commercial drone. The IMU data is less noisy than GNSS data, but drifts with time. After adding the IMU data to the EKF, the lattitude and longitude position errors improved, but the altitude estimate was still poor. In order to get a better altitude estimate for the consumer drone, we added barometer measurements to the EKF.

Our final proposed position solution is made up of two separate extended Kalman filters (EKFs). The first EKF fuses together raw GNSS measurements and the IMU measurements in order to estimate the drone's latitude and longitude. This EKF uses the IMU velocity measurements as the prediction update and the raw GNSS measurements for the update step. Algorithm 3 shows the EKF procedure to estimate the drone's latitude and longitude. The second EKF uses a static motion model for the predict step

and the barometer measurements for the update step in order to estimate the drone's altitude. Algorithm 2 shows the procedure that we used to estimate the drone's altitude. For our flight tests described in the results section, we measured GNSS signals from a Google Pixel 3 XL and IMU and barometer measurements came from the consumer drone platform. A diagram of our algorithm components is shown in Fig. 1. Python implementation of Algorithms 1, 3, and 2 are all open-source [8].

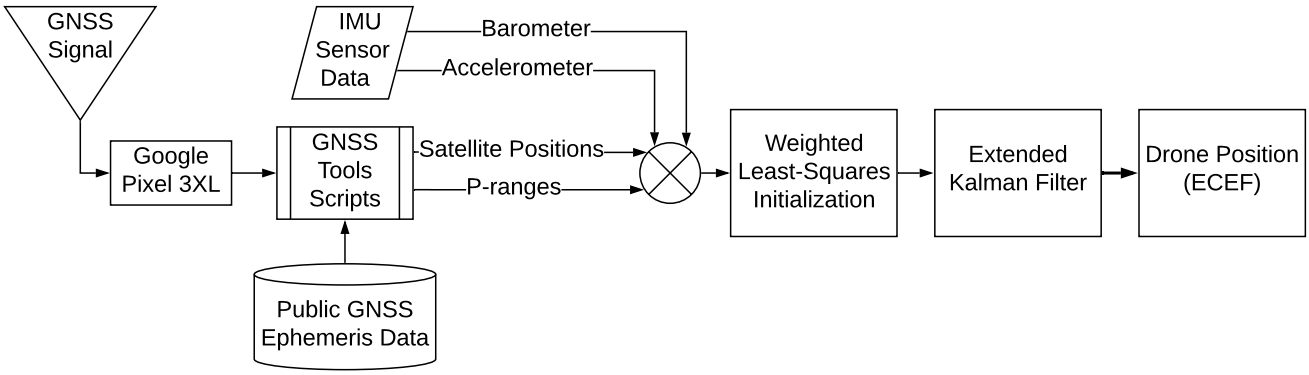



Figure 1: Algorithm process schematic

For each EKF implementation, we performed sensor noise characterization in order to estimate values for the Kalman filter's measurement noise matrix (R) and/or process noise matrix (Q). The R matrix was taken to be an identity covariance matrix, with diagonal terms corresponding respectively to the covariance of GNSS pseudoranges and barometer measurements ($\sigma^2_{sat}$ and $\sigma^2_{baro}$). Satellite measurement covariances were extracted from the GNSS analysis program, while the barometer covariance was estimated using a stationary sensor noise characterization, shown in Fig. 2. Similarly, the Q matrix consisted of IMU velocity covariances determined through the characterization run. The IMU velocity standard deviations $\sigma_x$ and $\sigma_y$ were taken to be 0.25 m/s. This value was chosen as a middle-ground value between $\sigma_x$ and $\sigma_y$ and was shown to provide the most accurate positioning solution. The barometer standard deviation $\sigma_{baro}$ was selected to be 0.5 m based on the characterization run. For specific implementations of each Kalman filter matrix, see the developed code in [8].

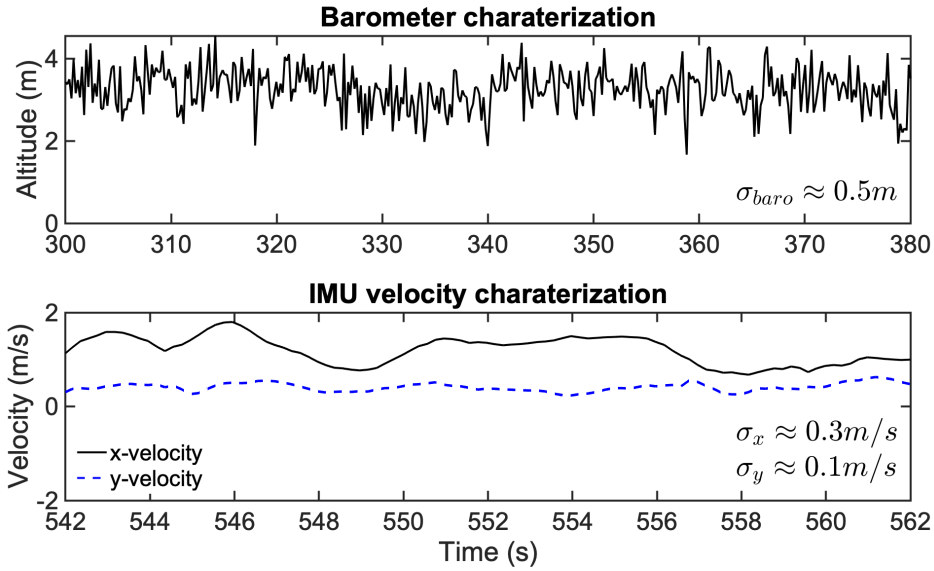



Figure 2: Results from a baseline sensor noise characterization run. Data from this run was used to estimate standard deviation and covariances of each sensor to specify the measurement and process noise matrices (R and Q) in the Kalman filter implementation.


(a) FAST features visual odometry is tracking shown by colored circles.

(b) Trajectory estimate of visual odometry.

Figure 3: Unsatisfactory results of implmementing visual odometry.

We further attempted to add monocular visual odometry for our position solution. We implemented an open-source monocular visual odometry solution that tracks FAST Features to compute optical flow and uses that to estimate the relative pose of the camera [9] [10]. We calibrated the camera on our commercial drone and tested the visual odometry framework on the video from our test flight. Fig. 3 shows the results from our visual odometry implementation. Fig. 3(a) shows that we were able to successfully track keypoints throughout the video. We created an adaptive threshold throughout so that there were always keypoints for the odometry to track, but not too many keypoints that tracking became too computationally expensive. Fig 3(b) shows that although the visual odometry seemed to perform well to estimate the translational movement of the drone, this particular visual odometry framework was not well suited to estimate the rotation of the drone. We flew the drone in roughly a rectangle, but the visual odometry estimates that the drone flew in a straight line. While other monocular visual odometry solutions are widely available, we found that reformatting them to be compatiable with our previous code was outside of the scope of this paper. For these reasons, we did not include visual odometry in our final position solution.

**Algorithm 2** Altitude Sensor Fusion EKF

**Inputs:**
    barometer measurements
**Initialize:**
    $\mathbf{T} \leftarrow$ concatenated and sorted timesteps
**for** $t \in \mathbf{T}$ **do**
    **Predict:**
        $\hat{\mu}_{t|t-1} = F\hat{\mu}_{t-1|t-1}$
        $P_{t|t-1} = F_t P_{t-1|t-1} F_t^T + Q$
    **Update Barometer:**
        $z_t \leftarrow$ barometer measurements
        $\tilde{y}_t = z_t - h(\hat{\mu}_{t|t-1})$
        $K_t = P_{t|t-1} H_t^T (R + H_t P_{t|t-1} H_t^T)^{-1}$
        $\hat{\mu}_{t|t} = \hat{\mu}_{t|t-1} + K_t \tilde{y}_t$
        $P_{t|t} = (I - K_t H_t) P_{t|t-1} (I - K_t H_t)^T + K_t R K_t^T$
**end for**

**Algorithm 3** Latitude & Longitude Sensor Fusion EKF

**Inputs:**
    GNSS pseudoranges
    Satellite positions
    IMU odometry
**Initialize:**
    $\mathbf{T} \leftarrow$ concatenated and sorted timesteps
**for** $t \in \mathbf{T}$ **do**
    **if** odometry exists at $t$ **then**
        **Predict:**
            $v_t \leftarrow$ imu odometry
            $\hat{\mu}_{t|t-1} = F\hat{\mu}_{t-1|t-1} + Bv_t$
            $P_{t|t-1} = F_t P_{t-1|t-1} F_t^T + Q$
    **end if**
    **if** GNSS exists at $t$ **then**
        **Update GNSS:**
            $z_t \leftarrow$ GNSS measurements
            $\tilde{y}_t = z_t - h(\hat{\mu}_{t|t-1})$
            $K_t = P_{t|t-1} H_t^T (R + H_t P_{t|t-1} H_t^T)^{-1}$
            $\hat{\mu}_{t|t} = \hat{\mu}_{t|t-1} + K_t \tilde{y}_t$
            $P_{t|t} = (I - K_t H_t) P_{t|t-1} (I - K_t H_t)^T + K_t R K_t^T$
    **end if**
**end for**

## 5. RESULTS

### 5.1 Flight Test Experimental Procedure

Flight tests served as the main avenue to collect data for input into the positioning algorithm. The flight test process also allowed collection of DJI flight logs which contained all relevant IMU sensor readings as well as the DJI-proprietary position solution. The experimental procedure for flights tests was as follows:

1. A desired flight trajectory was established for repeatability. Four markers were placed in a field to draw out roughly a 29 m by 25 m square, the corners of which represented 90-degree turning points. See Fig. 4.


Figure 4: Flight test route located at Stanford University's Roble Field. Difference between flight tests was the altitude at which the route was flown.

2. A Google Pixel 3XL was instsalled on the top of the DJI Mavic Pro. Care was taken to ensure clearance between the phone and the Mavic's blades. See Fig. 5.

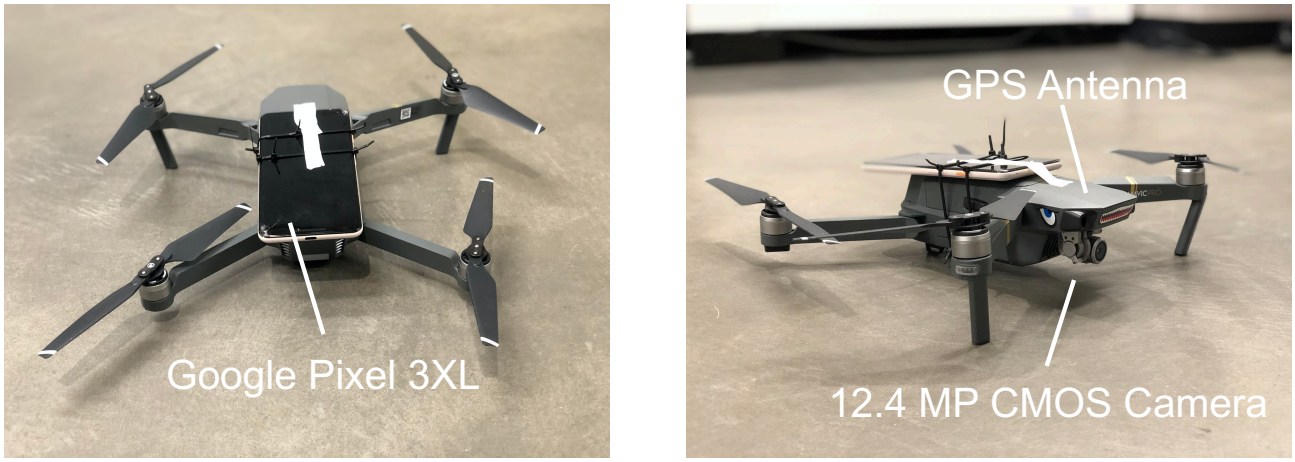


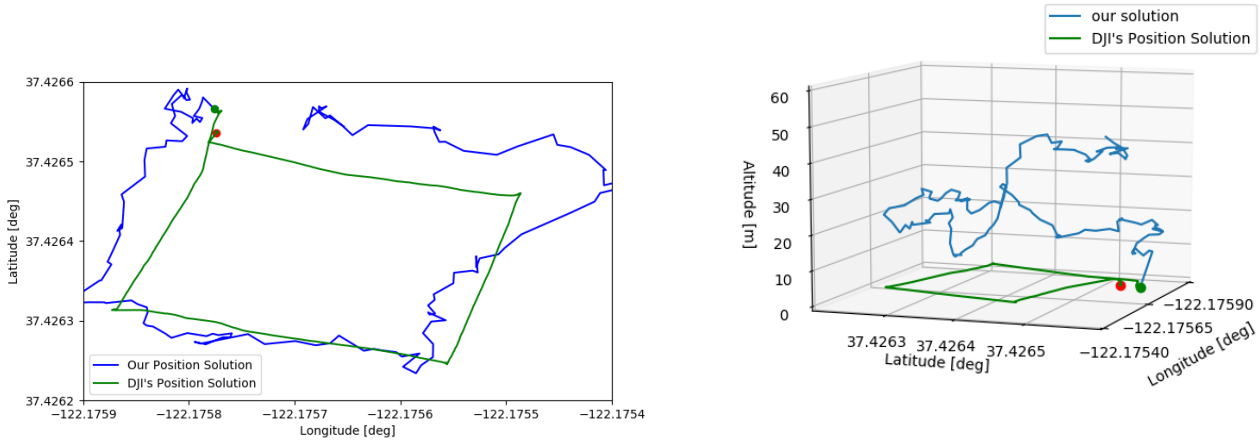

Figure 5: Flight test experimental configuration with DJI Mavic Pro and Google Pixel 3XL Android phone

3. The Mavic was powered on, automatically starting recording of internal flight logs. The GNSS Logger app on the Pixel was similarly set to begin recording GNSS satellite measurements [1].
4. Three separate flights along the specified test route were performed for indicated altitudes of 2, 10, and 22 m above the ground.
5. During each flight, forward-facing video was recorded at 2704x1520 resolution at 30 frames per second for visual odometry.

Fig. 4 exhibits the route designated for each flight test. The location was Roble Field at Stanford University, CA (37.426526$^o$, -122.175780$^o$). Note that due to increased wind velocity at higher altitudes, we limit subsequent discussion of results to the first flight test. Flight tests two and three were performed at considerably higher altitudes (10 and 22 m, respectively), leading to large variations in altitude, which were deemed unfavorable for presentation purposes and baseline comparisons. We note, however, that our proposed algorithm is expected to work equally as well at these altitudes.

### 5.2 GNSS Only EKF

In this section, we demonstrate the performance of the GNSS only extended Kalman filter discussed in the approach section and outlined in Algorithm 1. Fig. 6(a) shows the top-down view of our calculated trajectory overlaid on the consumer drone's proprietary position solution. Fig. 6(b) also shows both our position solution the proprietary solution, but displays the trajectory in three dimensions. Fig. 7 shows the accumulated error over time. As defined previously, we define the latitude, longitude, and altitude error in equations 1, 2, and 3 by the difference between our position solution and the proprietary solution. The latitude and logintude error are relatively low, but the altitude error using only GNSS is extremely high. By the end, the GNSS only EKF is estimating the drone is flying over 40m above the ground when in reality we were only a couple meters off of the ground. These poor altitude results demonstrated the need for sensor fusion.




(a) Latitude and longitude position solution comparison.

(b) 3D position solution comparison.

Figure 6: Position solution using only psuedorange in the extended Kalman filter.

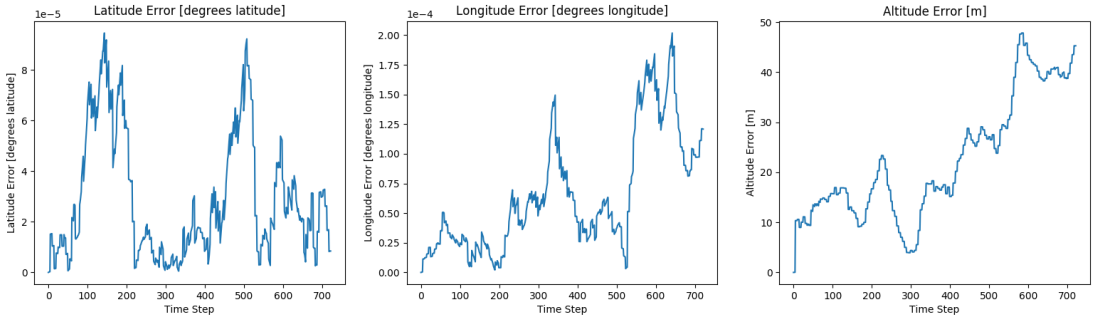
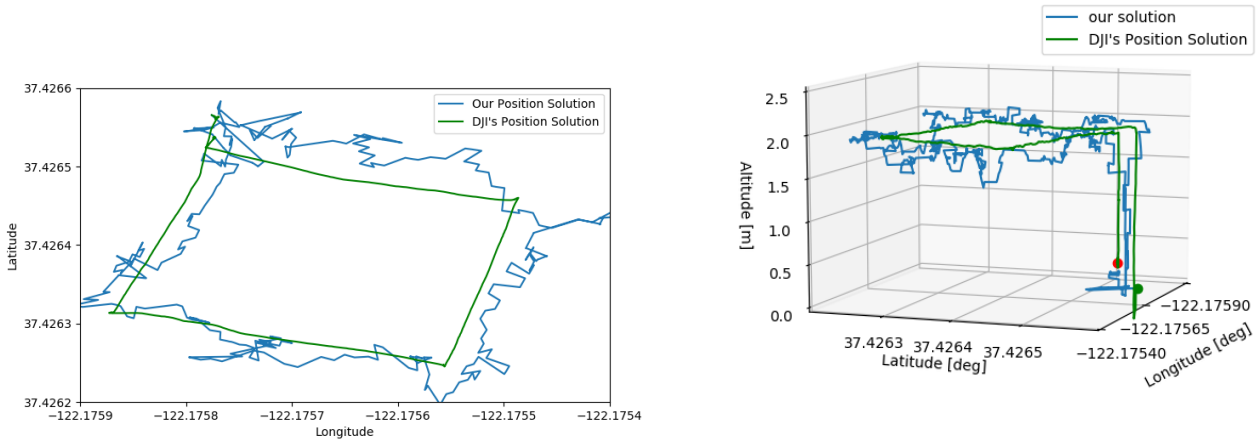



Figure 7: Position error using only psuedorange in the extended Kalman filter.

### 5.3 Sensor Fusion EKF

In this section, we demonstrate the performance of the sensor fusion extended Kalman filter discussed in the approach section and outlined in Algorithms 3 and 2. Fig. 8(a) shows the top-down view of our calculated trajectory overlaid on the consumer drone's proprietary position solution. Fig. 8(b) also shows both our position solution the proprietary solution, but displays the trajectory in three dimensions. Fig. 9 shows the accumulated error over time. As defined previously, we define the latitude, longitude, and altitude error in equations 1, 2, and 3 by the difference between our position solution and the proprietary solution.




(a) Latitude and longitude position solution comparison.

(b) 3D position solution comparison.

Figure 8: Position solution using psuedorange, IMU, and barometer in the extended Kalman filter.

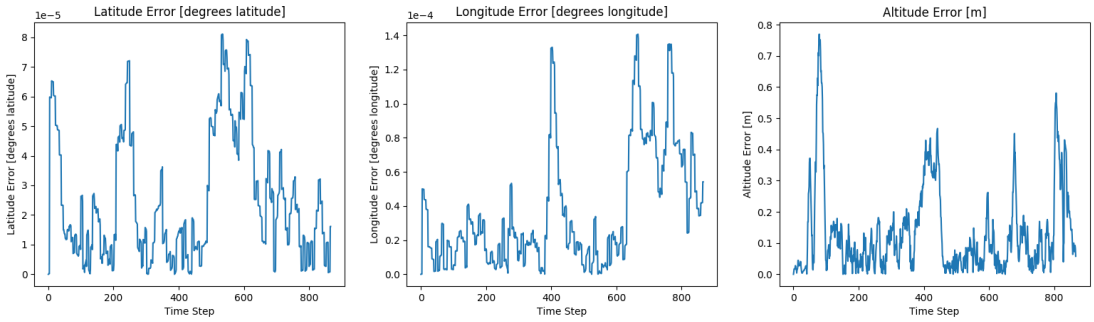



Figure 9: Position error using psuedorange, IMU, and barometer in the extended Kalman filter.

As defined previously in equations 4,5, and 6, we use average latitude, longitude, and altitude error respectively to quantify the performance of our extended Kalman filters against the proprietary position solution. Table 1 compares the average errors between the GNSS only EKF and the sensor fusion EKF. From this table, one can see that adding the IMU and barometer measurements slightly improved the latitude and longitude error and dramatically improved the altitude error. The altitude error decreased by two orders of magnitude by using the barometer measurements instead of only the GNSS raw measurements.

Table 1: Average errors compared with proprietary position solution.

| | GNSS Only | GNSS + IMU + Barometer |
|---|---|---|
| Latitude Error, $\bar{\epsilon}_{lat}$ [deg] | 2.94E-5 | 2.51E-5 |
| Longitude Error, $\bar{\epsilon}_{long}$ [deg] | 6.55E-5 | 3.68E-5 |
| Altitude Error, $\bar{\epsilon}_{h}$ [m] | 22.3 | 0.133 |

## 6. CONTRIBUTIONS OF TEAM MEMBERS

Both authors positively contributed to this paper and the underlying project research. Fig. 10 shows the respective responsibilities and contributions provided by each team member.

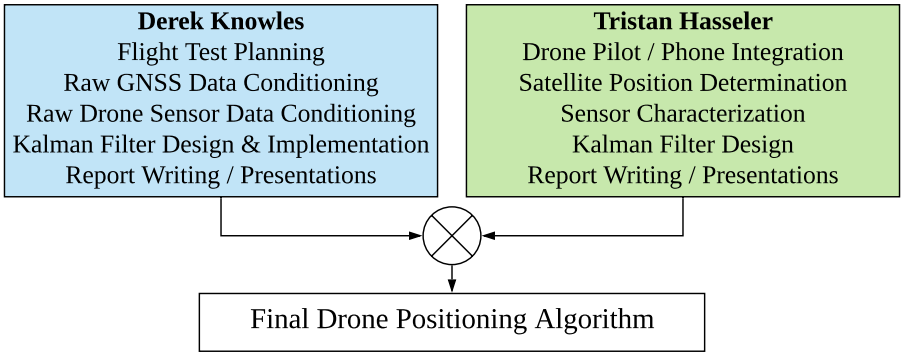



Figure 10: Team structure and key contributions to the final product.

## 7. CONCLUSIONS

In this paper, we developed and tested a simple and extensible sensor fusion algorithm for consumer drone positioning. We used an extended Kalman filter to fuse raw GNSS psuedoranges, IMU, and barometer measurements in order to create a comparable solution to current state of the art proprietary positioning solutions. We demonstrated that our sensor fusion position solution is more precise than a position solution using only GNSS psuedoranges, especially for altitude positioning. We note that while our solution is noisier than the DJI solution, it is likely sufficient for a number of consumer applications. Our solution could be refined by incorporating additional sensors or developing a higher-fidelity motion and dynamics model.

Our final sensor fusion algorithm produced latitude and longitude estimates that were on the order of $10^{-5}$ degrees different than the DJI solution. Additionally, after incorporating IMU sensors into our algorithm, we found that the mean altitude positioning error with respect to the DJI solution was reduced from 22 meters to 0.13 m, showing that GNSS altitude determination is greatly supplemented by additional sensors (in fact, the DJI doesn't even use GNSS height measurements). Finally, we demonstrated a preliminary application of visual odometry using on-board cameras. However, we found that the visual odometry package we used was not sufficiently robust to the rapid yawing motion associated with quad-copter turning.

We believe that with the capabilities of sensor fused position estimates, drones will be able to safely navigate increasingly cluttered environments with simple and lightweight open-source positioning algorithms such as the one developed here. For increased positioning accuracy, further work can be done to incorporate larger numbers of sensors.

## FUTURE DIRECTIONS

In the present work, we have demonstrated the use of our GNSS sensor fusion algorithm to provide satisfactory positioning for consumer drone systems using interfaced accelerometer and barometer data. Future work should focus on further extension of the algorithm to accommodate additional sensors and inputs. Of particular interest, we suggest that emphasis should be placed on developing a suitable extension for visual odometry leveraging the built in camera system present on most consumer drones. This addition could add substantial accuracy the results presented in this work. To accomplish this, work should focus on implementing a camera vision algorithm that can effectively handle rapid yawing maneuvers.

## ACKNOWLEDGMENTS

This work was supported by the National Natural Science Foundation of Shubh (Grant No. 1234) [11]. We would like to thank our Professor Dr. Grace Gao [12] for passionately teaching us about global positioning systems.

**REFERENCES**


[1] [Online]. Available: https://developer.android.com/guide/topics/sensors/gnss

[2] R. E. Kalman, "A new approach to linear filtering and prediction problems," *Journal of Fluids Engineering, Transactions of the ASME*, vol. 82, no. 1, pp. 35–45, mar 1960.

[3] S. Shen, N. Michael, and V. Kumar, "Autonomous multi-floor indoor navigation with a computationally constrained MAV," in *Proceedings - IEEE International Conference on Robotics and Automation*, 2011, pp. 20–25.

[4] S. Lynen, M. W. Achtelik, S. Weiss, M. Chli, and R. Siegwart, "A robust and modular multi-sensor fusion approach applied to MAV navigation," in *IEEE International Conference on Intelligent Robots and Systems*, 2013, pp. 3923–3929.

[5] S. Shen, Y. Mulgaonkar, N. Michael, and V. Kumar, "Multi-sensor fusion for robust autonomous flight in indoor and outdoor environments with a rotorcraft MAV," in *Proceedings - IEEE International Conference on Robotics and Automation*. Institute of Electrical and Electronics Engineers Inc., sep 2014, pp. 4974–4981.

[6] A. Nemra and N. Aouf, "Robust INS/GPS sensor fusion for UAV localization using SDRE nonlinear filtering," *IEEE Sensors Journal*, vol. 10, no. 4, pp. 789–798, 2010.

[7] P. et al, "Forensic analysis of unmanned aerial vehicle to obtain gps log data as digital evidence," *International Journal of Computer Science and Information Security*, vol. 15, 2017.

[8] [Online]. Available: https://github.com/betaBison/gnss-sensor-fusion

[9] [Online]. Available: https://github.com/avisingh599/mono-vo

[10] [Online]. Available: https://github.com/yoshimasa1700/mono_vo_python

[11] [Online]. Available: https://www.linkedin.com/in/shubhgupta1996

[12] [Online]. Available: https://profiles.stanford.edu/gracegao