

# Sound Classification and Localization by Acoustic Sensing on Raspberry Pi 4

Jaidon Lybbert, University of Washington

January 24, 2024

## **Abstract**

In this article a network of distributed acoustic sensing nodes is used to localize a sound signature using a time-difference-of-arrival (TDoA) algorithm. Accurate wall-clock times are obtained by GPS at each node, and a frame of acoustic data is recorded through a microphone on each node. Time differences are determined by correlation, and the resulting system of equations solved by the Chan-Ho algorithm to determine location. In a concurrent process, each node performs classification inference on a data frame using a pre-trained convolutional neural network (CNN). The resulting class identification is used to assign a process model to the tracked object for use in an Extended Kalman Filter (EKF) applied to the Chan-Ho localization result to make the final estimate of object position and trajectory.

## **1 Introduction**

This document is organized as follows. In Section 2 5 6 7. I conclude in 8 by discussing the overall takeaways from my experiments and things I would like to try in the future.

## 2 Background

## 3 TensorFlow Lite

## 4 Kalman Filters

The Extended Kalman Filter (EKF) is a well-described algorithm, where discrete samples from one or more sensors is combined with an analytic model of a system to estimate system state. A block diagram of the Kalman Filter from *Kalman Filters for Beginners* by Phil Kim [1] is shown in Fig 1.

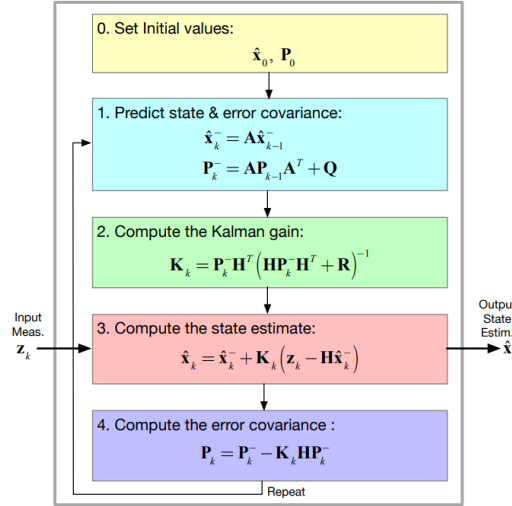


Figure 1: Block diagram of the Kalman Filter algorithm.  $\hat{\mathbf{x}}_0$  represents the initial state.

## 5 Methods

### 5.1 Environment Setup

For classification, I will be using TensorFlow Lite

### 5.1.1 GPS

I used the Adafruit Ultimate GPS USB breakout board to get GPS data through `gpsd` [3].

```
sudo apt install gpsd gpsd-clients
sudo systemctl stop gpsd.socket
sudo systemctl disable gpsd.socket
sudo killall gpsd
sudo gpsd /dev/ttyUSB -F /var/run/gpsd.sock
```

To test the device I used the command `cgps -s`, which prints out GPS status information to the console, as shown in Fig. 2.

		Seen 17/Used 5				
Time:	2024-02-29T09:13:29.000Z (0)	GNSS	PRN	Elev	Azim	SNR Use
Latitude:	47.67300667 N	GP 10	10	28.0	102.0	28.0 Y
Longitude:	122.30096333 W	GP 28	28	50.0	133.0	21.0 Y
Alt (HAE, MSL):	44.500, 61.800 m	GP 31	31	31.0	165.0	24.0 Y
Speed:	0.87 km/h	GP 32	32	53.0	58.0	23.0 Y
Track (true, var):	137.1, 15.4 deg	GL 8	72	34.0	58.0	21.0 Y
Climb:	0.00 m/min	GP 2	2	56.0	275.0	16.0 N
Status:	3D FIX (27 secs)	GP 4	4	3.0	238.0	0.0 N
Long Err (XDOP, EPX):	3.08, +/- 46.2 m	GP 8	8	5.0	226.0	0.0 N
Lat Err (YDOP, EPY):	1.42, +/- 21.3 m	GP 12	12	4.0	48.0	0.0 N
Alt Err (VDOP, EPV):	0.97, +/- 22.3 m	GP 17	17	11.0	325.0	0.0 N
2D Err (HDOP, CEP):	2.87, +/- 54.5 m	GP 25	25	6.0	77.0	0.0 N
3D Err (PDOP, SEP):	3.03, +/- 57.6 m	GL 1	65	68.0	325.0	0.0 N
Time Err (TDOP):	3.97	GL 2	66	26.0	267.0	0.0 N
Geo Err (GDOP):	6.67	GL 17	81	38.0	317.0	0.0 N
ECEF X, VX:	n/a n/a	GL 22	86	17.0	185.0	0.0 N
ECEF Y, VY:	n/a n/a	GL 23	87	54.0	233.0	0.0 N
ECEF Z, VZ:	n/a n/a	GL 24	88	37.0	317.0	0.0 N
Speed Err (EPS):	+/- 332 km/h					
Track Err (EPD):	n/a					
Time offset:	0.517937753 s					
Grid Square:	CN87uq31					

Figure 2: Terminal display of GPS information from the `cgps -s` command.

## 6 Results

## 7 Discussion

## 8 Conclusion

**Note** *All my code and project files for this and future reports, can be found on my GitHub repository [2].*

## References

- [1] P. Kim and L. Huh. *Kalman Filter for Beginners: With MATLAB Examples*. CreateSpace Independent Publishing Platform, 2011.
- [2] Jaidon Lybbert. Classwork for embedded and real-time systems, university of washington, winter 2024. <https://github.com/jaidonlybbert/EEP522A-W24>, 2024. Accessed: 2024-01-23.
- [3] Kevin Townsend. Adafruit ultimate gps with gpssd. <https://learn.adafruit.com/adafruit-ultimate-gps-on-the-raspberry-pi/setting-everything-up>, 2023. Accessed: 2024-02-29.