Tracking of Multiple Pixel Targets Using Multiple Cameras

Jin Zhou and Chiman Kwan(✉)

Signal Processing, Inc., Rockville, MD, USA
ferrryzhou@gmail.com, chiman.kwan@signalpro.net

Abstract. Simultaneous tracking of multiple and small/pixel targets is important for many applications such as ship, aircraft, and vehicle tracking. This paper presents a practical and efficient framework for the above problem. The key ideas in the framework include 3D modeling of multiple cameras and the application of multiple Extended Kalman Filters (EKF) for multi-target tracking. Many practical issues such as multiple small targets, measurement noise, false targets, partial or missing target observations, multiple cameras, etc. have been addressed. Extensive experiments demonstrated the feasibility of the proposed framework.

Keywords: Target tracking · Kalman Filter · Pixel target

1 Introduction

There are many applications that require the tracking of small/pixel targets using multiple sensors. Image sensors (EO/IR) have many advantages over radars such as better mobility and lower cost. Hyperspectral imagers is another potential sensor that can be used for target detection and tracking.

In this project, it is assumed that a set of unmanned air vehicles (UAVs) or other platforms are used to detect and track multiple targets (vehicles or aircraft), which appear as pixel targets from long ranges in cameras. Each UAV carries one or more infrared cameras to capture heat emission from remote objects/aircraft. Figure 1 shows a sample scenario of target tracking using two cameras. In this scenario, two cameras are watching a remote site, where three targets are just taking off. In general, there can be many targets at the same or different times and we have many cameras to track them simultaneously.

Our pixel target tracking system consists of several key components. First, pixel targets are detected in each camera. Many algorithms can be used here, including anomaly detection algorithms [1–6] and foreground/background separation using sparsity [7]. Second, targets in multiple cameras are associated via a 3D camera geometric model. This model is very important for dealing with multiple targets in multiple cameras. Third, although sophisticated tracking algorithms such as [8–13] can be used, a simple and efficient target tracking algorithm based on Extended Kalman Filter (EKF) [16–18] is used to track the targets. Fourth, a number of practical algorithms have been utilized to address many practical issues such as false targets, partial or missing observations, coordination between multiple cameras, etc.

T. Huang et al. (Eds.): ISNN 2018, LNCS 10878, pp. 484–493, 2018.
https://doi.org/10.1007/978-3-319-92537-0_56

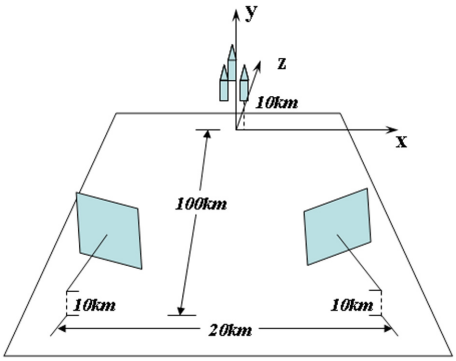


Fig. 1. Tracking multiple targets using multiple cameras.

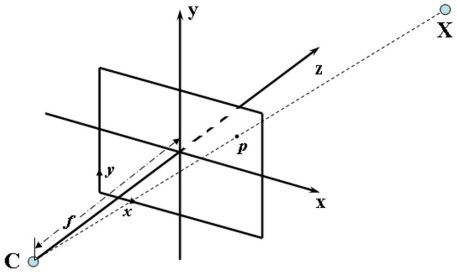


Fig. 2. Geometric model of imaging.

Our tracking system is fully automatic, robust, flexible, scalable, efficient and practical for multiple pixel targets tracking. With all these capabilities, we believe that our multiple camera based tracking system is able to work in real world situations and provides a better solution than radar based tracking system in some complex scenario.

In the following, we first present the geometric model of camera and imaging process in Sect. 2. In Sect. 3, we then describe the geometric model based EKF for 3D target tracking from 2D camera observations. We combine multiple camera observations into a single observation vector so that all information can be used simultaneously. In this way, information from partial frames are effectively utilized. We further describe how we perform automatic new target state initialization based on triangulation. In addition, we will also present the strategy for multi-target and multi-group tracking, which involves automatic camera adjustment and multi-camera coordination. In Sect. 4, we demonstrate the capability of our tracking system with two simulation experiments, which includes observation noise, missing targets, false alarms, non-linear trajectory, multiple target groups, multiple sensors, and long video sequences. Finally, some remarks will be given in Sect. 5.

2 Geometric Models

Figure 2 shows the camera model. The symbols x, y and z denote the three axes of the camera. The letter C is the camera location. X is a 3D point and p is the corresponding image point. x and y are the image coordinate axes; the origin (0, 0) is at bottom left. f is the focal length of the camera.

Given a 3D point X, its projection on image is p which can be computed by the following equation [14]

$$\hat{p} = KR[I| - C]\hat{X} \tag{1}$$

where $\hat{X}$ is the homogeneous coordinates of X and $\hat{p}$ is the homogeneous coordinates of p. That is,

$$p = \hat{p}(1:2)/\hat{p}(3) \text{ and } X = \hat{X}(1:3)/\hat{X}(4) \tag{2}$$

K is the camera internal matrix. R is the rotation matrix. For simplicity, we can use normalized image coordinates

$$\bar{p} = K^{-1}\hat{p}. \tag{3}$$

As a result,

$$\bar{p} = R[I| - C]\hat{X} = R(X - C). \tag{4}$$

In our simulation and tracking studies, we will use the above normalized image coordinates.

3 Multi-camera Tracking Based on Extended Kalman Filters

3.1 Tracking Using Extended Kalman Filter

We use Extended Kalman Filter (EKF) for its simplicity to perform the 3D object tracking based on 2D observations. The state vector is represented as

$$\mu = [\,x \quad y \quad z \quad v_x \quad v_y \quad v_z\,]^T \tag{5}$$

where $(x, y, z)^T$ is target's 3D location and $(v_x, v_y, v_z)^T$ is the speed vector. We use a linear motion model represented by

$$\mu_t = A\mu_{t-1}$$

where

$$A = \begin{bmatrix} 1 & 0 & 0 & \Delta t & 0 & 0 \\ 0 & 1 & 0 & 0 & \Delta t & 0 \\ 0 & 0 & 1 & 0 & 0 & \Delta t \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}.$$

The observation model is given by

$$\bar{p} = R(X - C) \tag{6}$$

where $X = \mu(1:3)$

$$p(i) = \bar{p}(i)/\bar{p}(3) = \frac{R_{i:}(X - C)}{R_{3:}(X - C)}$$

where $i = 1, 2$. $R_{i:}$ is the i^{th} row of R and $[p(1), p(2)]$ are the normalized image coordinates.

For M cameras, we combine all the observations together to form a single measurement vector

$$Z = [p_1(1) \quad p_1(2) \quad p_2(1) \quad p_2(2) \quad \ldots \quad \ldots \quad p_M(1) \quad p_M(2)]^T. \tag{7}$$

In practice, due to missing pixels/frames or partial observations, it is possible that only a subset of the sensors have observations of the target. Therefore, the observation vector (Z) only contains the available information from a subset of the sensors [15].

3.2 Multi-target Tracking

To track multiple targets simultaneously, we maintain an EKF tracker for each target independently. To update each tracker with new observations, we use nearest neighbor strategy, i.e. for each image, the point which is closest to prediction position is assigned as new observation. To cope with missing pixels, we set a distance threshold. If the closest distance is larger than the threshold, we assume there is no new observation. If a tracker does not have any new observations for a long time, its covariance matrix will be large and we drop/stop it.

3.3 Automatic State Initialization for New Targets

The EKF tracker assumes there is an initial state to begin with. For image sensors, the 2D observations do not equal to the 3D positions and cannot be directly used as initial states. To initialize states, we use triangulation method to generate all possible 3D points based on the observations from multiple cameras. When there are many cameras and many false alarms, the number of all possible 3D points could be huge. We use re-projection error to eliminate a majority of the false targets.

3.4 Tracker Pruning

In the previous paragraph, we mentioned that even with initial tracker elimination based on re-projection error, there may still be a lot of false tracks. One way to get rid of false tracks is to use error covariance to measure the confidence of a tracker. If the covariance is large, the confidence is low. We use the inverse sum of the diagonal elements of the covariance matrix to indicate the confidence level. If the confidence level is below a threshold, we terminate the track and its associated EKF tracker. Usually, if a tracker cannot gain enough observation updates, its covariance will grow and finally terminate. Using the above technique, most of the false tracks and their corresponding trackers are eliminated as time goes on.

With enough time, all the survived tracks will merge to true targets. This leads to another issue: many tracks may become identical to each other. In practice, we do not need all of them. For each true target, we only need one tracker. To prune these redundant trackers, we measure similarity of the state vectors and if two trackers are too close, the one with lower confidence is terminated.

With tracker pruning, the number of total trackers will gradually converge to the number of true targets and the system speed will become very fast.

4 Simulations and Tracking Experiments

In this section, we use two simulations to demonstrate the effectiveness of our tracking system. The first simulation is a simple one, i.e. linear trajectory and single target group, and no need for camera adjustment. This simulation is used to demonstrate the basic functions of the tracking system such as robustness to noise, false alarms, missing target and the capability of automatic initialization. The second simulation is very challenging: non-linear trajectory, multiple groups, long time (say, 700 frames). This simulation is used to demonstrate advanced feature of the tracking system such as partial observation, active tracking, automatic multi-camera coordination, non-linear trajectory and tracker pruning. In both simulations, we used a simple target detection algorithm (Reed-Xiaoli Detector [2]) to detect the targets. Other sophisticated algorithms could be used as well where low false alarms can be achieved. However, the objective of these simulations is to demonstrate the proposed tracking system rather than target detection.

In the first simulation, we placed 4 cameras and generated 3 targets. The 3 targets are moving at a constant velocity. Figure 3 shows the simulation setup and the tracking process. Figure 4 shows the images highlighting the detected target pixel's locations. We simulated 30 frames and frame 1, 11, 16, and 26 are shown in Fig. 3. In each image, there are a random number (from 0 to 10) of false alarms. In addition, each target has a probability of 0.1 of missing in an image. The pixel locations are disturbed with random Gaussian noise with mean of 0.5 pixel. In Fig. 3, the camera is represented by 3 axes: red is z axis, blue is x axis and green is y axis. Image plane is in the end of z axis. Green circles represent target observations on images, which include true target's image and false alarms. Blue circles are 3 real targets in 3D space. Red crosses are estimated 3D targets in EKF tracker. Red circles represent the diagonal mean of EKF covariance matrix. In the beginning, there are 6 EKF trackers, only two of them are correct. In frame 11, one of the false target's track (the most right one in frame 1) disappears. Two false targets' tracks close to true targets merged to true target's location. Starting from frame 16, all false target trackers disappear and only 3 true target trackers exist and overlap with ground truth position.

In the second simulation, the experimental setup is as follows:

(1) There are 3 targets moving in circles. One target's speed is different from another two.
(2) There are 7 cameras.
(3) False alarms range from 0 to 5 for each image.
(4) There are 700 frames.

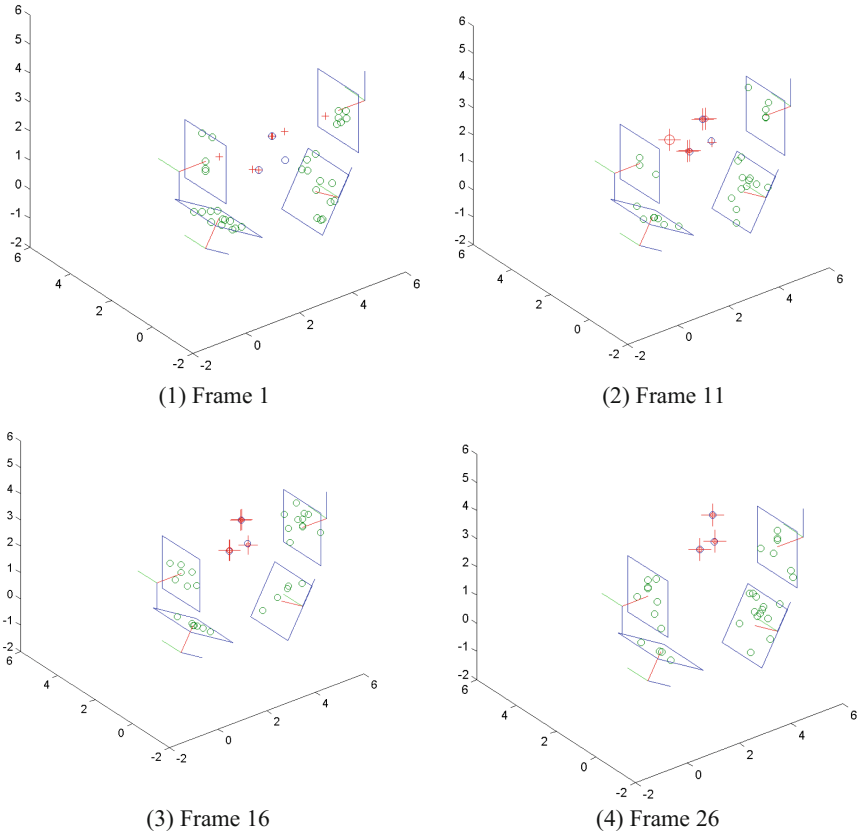


Fig. 3. 3D illustration of multi-target multi-camera simulation and tracking.

Other configuration parameters are the same as the previous simulation. Figure 5 shows trajectories of ground truth targets (in black circles) and EKFs (in lines). We can observe that all three real targets' trajectories are successfully captured. There are some false tracks. Most of them are terminated after a short time. Only 1 false track lasts more than 200 frames. Note that for one of the targets, there are two color lines, i.e. cyan and blue. This means that there are two corresponding tracks. In the beginning, the cyan track is tracking the target. However, after several hundred frames, it loses the target. Fortunately, our system can automatically recover the failed case by automatically initialize new targets. After several frames of failed tracking, a new blue tracker is created to track the target.

Figure 6 shows a realistic rendering of frame 500. The UAV network are coordinated to cover all targets. Note that for visualization, we highlight the target pixel as well as false alarms in the image. In reality, they are all single pixels.

Figure 7 shows a set of sample frames. In the beginning, there are many (more than 400) trackers that are initialized. After 10 frames, some trackers' confidence level becomes low (see big circles in the image). In frame 70, most false trackers are

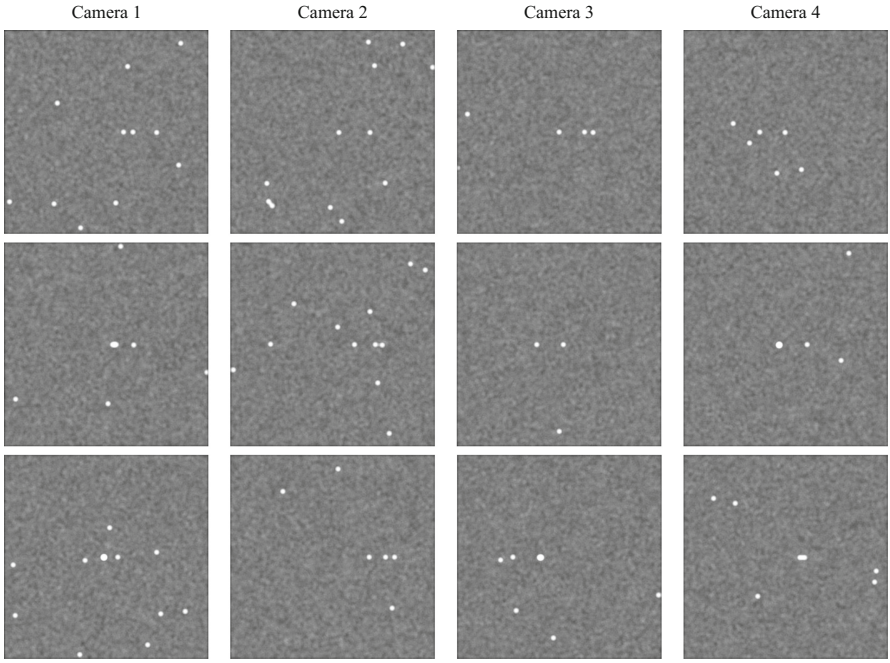


Fig. 4. Simulated images showing the locations of targets as well as false alarms. Columns are images from different cameras. Rows are images from time frame 1, 11, and 21.

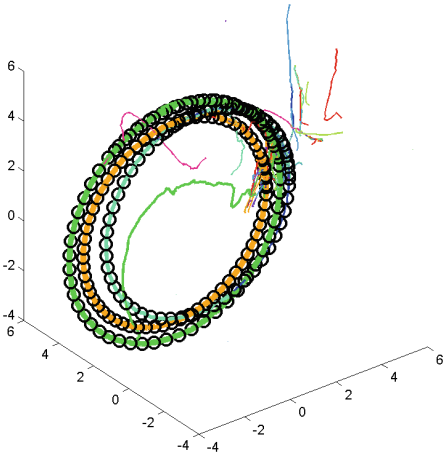


Fig. 5. Ground truth targets trajectory (in black circles) and EKF trackers' trajectory (in lines). (Color figure online)

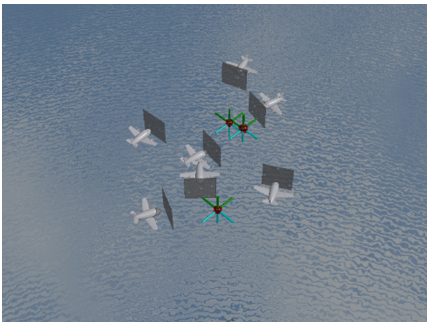
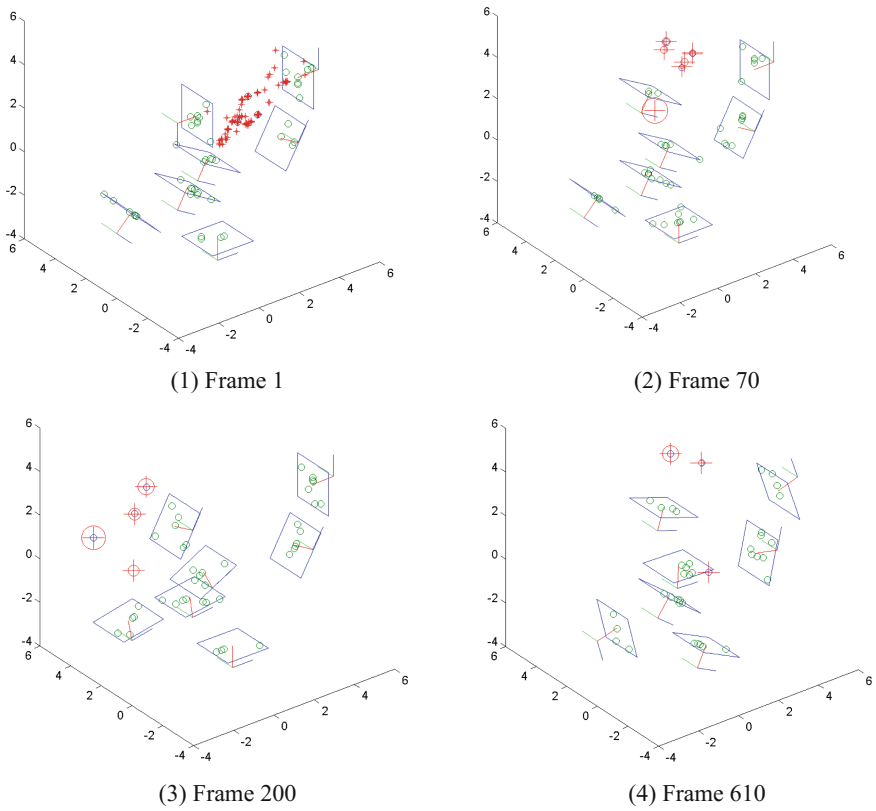


Fig. 6. Frame 500 with realistic rendering. Three axes are the targets and red balls are EKF trackers. (Color figure online)



(1) Frame 1 (2) Frame 70

(3) Frame 200 (4) Frame 610

Fig. 7. 3D illustration of simulation and tracking.

eliminated. Note that in frame 70, the top camera changed its view angle to track the targets. In frame 200, most cameras changed their view angle. The top camera has changed at least twice. In frame 500, two target groups are totally separated due to different running speed. In this frame, both groups received enough attention. The two left cameras are watching the bottom group and other cameras are watching another group. Actually, the two right cameras can see both groups. So the bottom target is covered by 4 cameras and another two targets are covered by 5 cameras. Frame 610 has similar situation and all of the targets get enough coverage.

5 Conclusions

In this paper, we present a general framework for tracking multiple pixel targets using multiple cameras. Many practical issues such as missing pixels, partial observations, measurement noise, multiple targets, etc. have been addressed. Extensive simulations clearly demonstrated the performance of the framework.

Acknowledgments. This research was supported by MDA under contract # HQ0147-12-C-7883. Distribution Statement A. Approved for public release (17-MDA-9402 (31 Oct 17)); distribution is unlimited.

References

1. Ayhan, B., Kwan, C., Li, X., Trang, A.: Airborne detection of land mines using mid-wave infrared (MWIR) and laser-illuminated-near infrared images with the RXD hyperspectral anomaly detection method. In: Fourth International Workshop on Pattern Recognition in Remote Sensing, Hong Kong (2006)
2. Zhou, J., Kwan, C., Ayhan, B., Eismann, M.: A novel cluster Kernel RX algorithm for anomaly and change detection using hyperspectral images. IEEE Trans. Geosci. Remote Sens. **54**(11), 6497–6504 (2016)
3. Ayhan, B., Dao, M., Kwan, C., Chen, H., Bell, J.F., Kidd, R.: A novel utilization of image registration techniques to process mastcam images in mars rover with applications to image fusion, pixel clustering, and anomaly detection. IEEE J. Sel. Top. Appl. Earth Obs. Remote Sens. **10**, 4553–4564 (2017)
4. Wang, W., Li, S., Qi, H., Ayhan, B., Kwan, C., Vance, S.: Identify anomaly component by sparsity and low rank. In: IEEE Workshop on Hyperspectral Image and Signal Processing: Evolution in Remote Sensor (WHISPERS) (2015)
5. Li, S., Wang, W., Qi, H., Ayhan, B., Kwan, C., Vance, S.: Low-rank tensor decomposition based anomaly detection for hyperspectral imagery. In: IEEE International Conference on Image Processing (ICIP) (2015)
6. Qu, Y., Guo, R., Wang, W., Qi, H., Ayhan, B., Kwan, C., Vance, S.: Anomaly detection in hyperspectral images through spectral unmixing and low rank decomposition. In: International Geoscience and Remote Sensing Symposium (IGARSS), pp. 1855–1858 (2016)
7. Dao, M., Kwan, C., Ayhan, B., Tran, T.: Burn scar detection using cloudy MODIS images via low-rank and sparsity-based models. In: IEEE Global Conference on Signal and Information Processing, pp. 177–181 (2016)
8. Zhao, Z., Chen, H., Chen, G., Kwan, C., Li, X.: Comparison of several ballistic target tracking filters. In: Proceedings of American Control Conference, pp. 2197–2202 (2006)
9. Zhao, Z., Chen, H., Chen, G., Kwan, C., Li, X.: IMM-LMMSE filtering algorithm for ballistic target tracking with unknown ballistic coefficient. In: Proceedings of SPIE, Signal and Data Processing of Small Targets, p. 6236 (2006)
10. Kalal, Z., Mikolajczyk, K., Matas, J.: Tracking-learning-detection. IEEE Trans. Pattern Anal. Mach. Intell. **34**(7), 1409–1422 (2012)
11. Mei, X., Ling, H.: Robust visual tracking and vehicle classification via sparse representation. IEEE Trans. Pattern Anal. Mach. Intell. **33**(11), 2259–2272 (2011)
12. Yang, M.H., Zhang, K., Zhang, L.: Real-time compressive tracking. In: European Conference on Computer Vision (2012)
13. Ma, C., Huang, J.B., Yang, X., Yang, M.H.: Hierarchical convolutional features for visual tracking. In: Computer Vision (ICCV) (2015)
14. Hartley, R., Zisserman, A.: Multiple View Geometry in Computer Vision. Cambridge University Press, Cambridge (2003)
15. Song, B., Ding, C., Kamal, A., Farrell, J., Roy-Chowdhury, A.: Distributed camera networks: integrated sensing and analysis for wide area scene understanding. Sig. Process. Mag. **28**(3), 20–31 (2011)

16. Lewis, F.L.: Optimal Estimation. Wiley, Hoboken (1986)
17. Kwan, C., Chou, B., Kwan, L.M.: Comparison of deep learning and conventional object tracking approaches for low quality videos. In: 15th International Symposium on Neural Networks (2018)
18. Kwan, C., Lewis, F.L.: A note on Kalman filtering. IEEE Trans. Educ. **42**(3), 225–227 (1999)