Fusion Technology of Radar and RGB Camera Sensors for Object Detection and Tracking and its Embedded System Implementation

Jian Xian Lu[1], Jia Cheng Lin[2], Vinay M.S.[3] Po-Yu Chen[4], Jiun-In Guo[5]

[1,2,3,5]Department of Electronics Engineering, National Chiao Tung University, Taiwan R.O.C.

E-mail: [1]maverick99876@gmail.com, [2]ego878gem104@gmail.com, [3]vinay.ms23@gmail.com, [5]jiguo@nctu.edu.tw

[4]Mediatek Inc., Taiwan R.O.C.

E-mail: marvin.chen@mediatek.com

***Abstract*— This paper proposes a Camera and Radar sensor fusion algorithm combining Radar and RGB camera for object detection. The proposed design detects the type of the object with images/videos inputs and tracks the object followed by using a radar object detection and recognition to provide the actual type and distance of the object from the radar.**

Utilizing cameras, the deep learning model is employed to identify the objects in the image by applying Unscented Kalman Filter (UKF) and Kalman filter to track the objects. After projecting the radar tracking points in images, the radar tracking points and the image tracking points are regarded as the input to the Track-to-Track system to generate more stable tracking points. Finally, Track-to-Track points are input to the next image tracking to stabilize the labeling of the objects in the image. The average accuracy of the proposed method is around 95%, with 15% higher compared to only using deep learning model. The proposed sensor fusion method is developed on a desktop computer and implemented on the Nvidia Xavier embedded system yielding about 10 FPS with 77GHz radar input and 640x360 image input.

***Keywords*—** *Depth sensor, Object tracking, Pedestrian detection, Radar, Sensor Fusion*

I. Introduction

Nowadays, the Advanced Driving Assistance Systems (ADAS), composed of many sensors such as radars, cameras, ultrasonic sensors etc., is the standard aid for the vehicles and drivers to understand the surrounding situation and make the correct decision to achieve safe driving. Thus, it aids in avoiding driving in the dangerous scenarios.

Camera is the most common sensor in our daily life. It has many advantages like cheaper price, higher resolution and higher frame rate. Since the deep learning technology has started gaining popularity, objects detection using cameras is widely adopted in many applications particularly in surveillance, automotive, and self-driving vehicles. However, the cameras are greatly influenced by the varying light intensities. To overcome this drawback, radars with advantages such as high update rate and no interference by the varying lighting and weather conditions and stable in offering front objects' real distance and relative speed are used. Thus, they become the best choice to assist the cameras.

In sensor fusion technology, sensors compensates for the disadvantages of the other sensors and together overcome the worse situations by improving the detection efficiency. For ADAS applications, the fusion of camera and radar sensors is more reliable and safer compared to using a single sensor, either a camera or a radar.

The fusion of radar and camera sensors is the best solution for the surveillance and ADAS system because of the good performance of object detection in the camera and stable object tracking in the radar. Fig. 1 shows an example of the surveillance system using both radar and camera sensors.

Fig.1 Surveillance system with sensor fused with radar at the top and camera at the bottom

II. The Proposed Algorithm

The flowchart of the proposed Radar and RGB Camera Sensor Fusion architecture is as in Fig.2. The proposed method initially combines radar's and camera's tracking result followed by the Track-to-Track algorithm to generate the stable tracking result. In the end, the tracking result of the Track-to-Track algorithm is accompanied with camera trackers to generate more stable trackers. The following sections details the steps involved in the proposed algorithm.

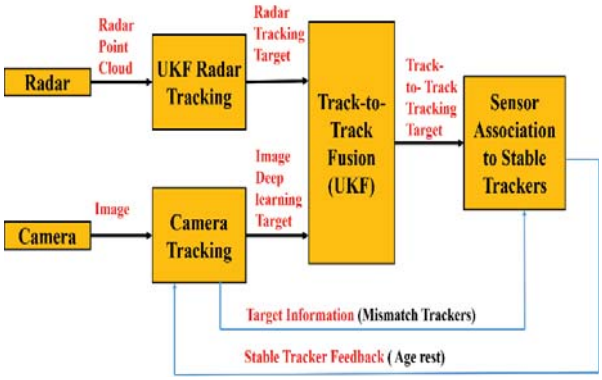


Fig.2. Flowchart of the proposed algorithm

A. Radar/Camera Calibration

Fig. 3 shows the setup of the radar put at different angles. The horizontal angle is set to zero as in [1] with reference to which the other angles are calculated as in [2]. The relation of camera and radar coordinates are as in Fig. 4. The camera coordinates are set as the real world coordinate. The calibration sequence is as below:

Radar coordinate (x_r, y_r, z_r) → radar world coordinate (x_{rw}, y_{rw}, z_{rw}) → camera world coordinate (x_{cw}, y_{cw}, z_{cw}) → camera coordinate (x_c, y_c, z_c) → image coordinate (x_p, y_p).

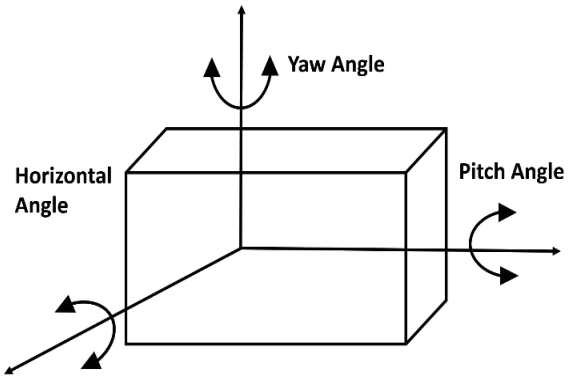


Fig.3 Radar installation angle diagram

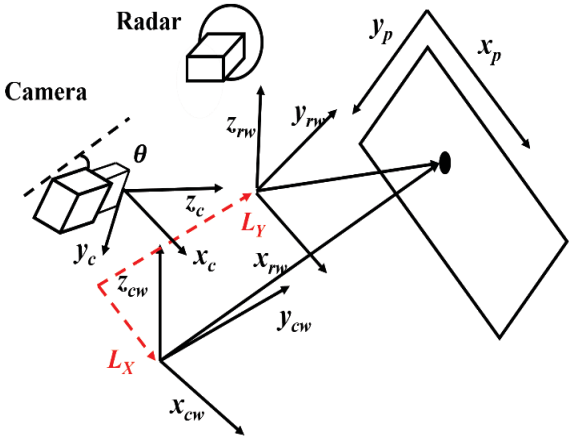


Fig. 4 Relation of radar and camera coordinates

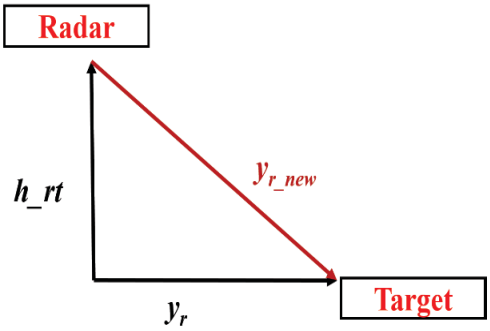


Fig.5 Relation between the radar and a target

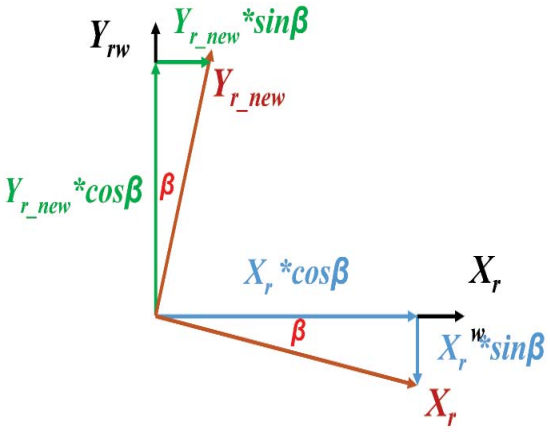


Fig.6 Relation of yaw angle and the radar coordinates

Then the radar coordinates are transformed into the radar world coordinates. The radar yaw angle and height difference between the radar and a target are considered to calculate the projected depth distance, y_{r_new}. Fig. 5 shows the relation between the radar and a target where h_rt is the height difference from a target to the radar. The function to generate projected y_{r_new} is given by (1). Fig. 6 and (2) shows that radar coordinate (x_r, y_{r_new}) through yaw angle β to generate real radar world coordinates (x_{rw}, y_{rrw}).

$$y_{xnew} = \sqrt{(y_y)^2 - (h_{xz})^2} \tag{1}$$

$$\begin{cases} y_y = y_{r\_new} * cos\beta - x_y * sin\beta \\ x_{rw} = y_{r\_new} * sin\beta + x_y * cos\beta \end{cases} \tag{2}$$

The transform from radar world coordinates to camera world coordinates is given by (3), where L_x and L_y are the horizontal and vertical distances between the radar and the camera followed by transferring the camera world coordinates considering the pitch angle effect as in (4). The pitch angle α and the camera height H are added to the equation.

$$\begin{cases} x_{cw} = x_{rw} - L_x \\ y_{cw} = y_{rw} + L_y \end{cases} \tag{3}$$

$$\begin{cases} X_c = X_{cw} \\ y_c = -y_{cw} * sin \propto -Z_{cw} * cos \propto +H * cos \propto \\ Z_c = y_{cw} * cos \propto -Z_{cw} * sin \propto +H * sin \propto \end{cases} \tag{4}$$

The second step is similar to the radar calibration. Fig.7 shows the effect of a camera yaw angle β to the coordinate transform. The function of coordinate transform from camera coordinate to new camera coordinate is given by (5). Finally, a new camera coordinates to image coordinates are projected as in [3]. The focal length (f_x, f_y) and principal point (ox, oy) are calculated as in [3] and final calibration equation [4] is as in (6).

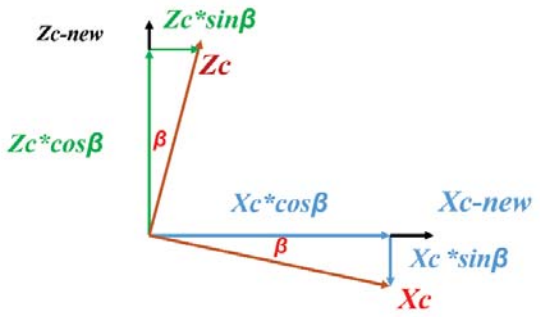


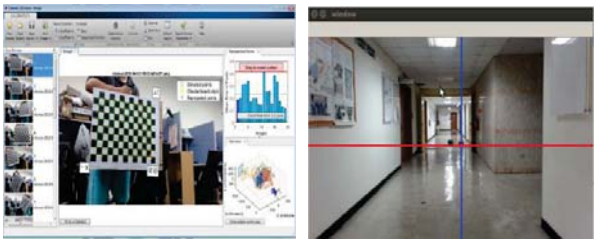
Fig. 7 Camera yaw angle "$\beta\beta$" effect the camera coordinate

$$\begin{cases} X_{c_{new}} = Z_c * sin\beta + X_c * cos\beta \\ Y_{c\_new} = Y_c \\ Z_{c\_new} = Z_c * cos\beta - X_c * sin\beta \end{cases} \quad (5)$$

$$\begin{cases} X_p = \frac{X_{c\_new}}{Z_{c\_new}} * f_x + O_x \\ Y_p = \frac{Y_{c\_new}}{Z_{c\_new}} * f_y + O_y \end{cases} \quad (6)$$

Firstly, the MATLAB camera calibration toolbox is input with about thirty pictures of different orientations and with the distance of a 2D marker to find the camera instinct parameters as in Fig. 8(a). With these, the four camera instinct parameters namely f_x, f_y corresponding to focal length and o_x, o_y corresponding to principle points of the camera, respectively in horizontal and vertical directions are obtained.

Then, the real distance of the pixel in an image is found by adjusting the intersection points of the blue and red lines to match the ground of target's center as in Fig. 8(b) with which the distance of a target, and the height of camera and radar are measured as the initial parameters.



(a) (b)

(c)

Fig.8 (a) MATLAB camera calibration toolbox with different picture inputs, (b) MATLAB camera calibration toolbox with different picture inputs, (c) Result of candidate sweeping pitch angle $\alpha\alpha$ and yaw angle $\beta\beta$

Lastly, the pitch angle and yaw angle of radar and camera are calculated by sweeping the pitch angle α in the range of -90° to +90° and yaw angle β ranging from -90° to +90° to match the target pixel as shown in (7). Fig. 8(c) shows the results of sweeping angle.

$$\begin{cases} Pitch\ angle \propto: -90° \leq \propto \leq 90° \\ Yaw\ angle\ \beta: -90° \leq \beta \leq 90° \end{cases} \quad (7)$$

The calibration accuracy is then evaluated with the ground truth using a laser range finder as shown in Fig. 9(a). The calibration error respectively ranges from 2.8-6.1cms for the distances ranging from 2.9-9.4 meters. Fig. 9(b) shows the results of the total calibration from radar to camera. The red points are the projected radar points within 5 meters, and the green points are the radar points beyond 5 meters.

(a) (b)

Fig.9 (a) Calibration result of 10 meter (Left), and laser rangefinder groundtruth (Right) (b) Result of calibration from radar to camera

The errors of calibration are less than 0.1 meters in each range of distance as shown in Table 1.

TABLE 1. CALIBRATION ERROR FOR DISTANCE RANGING FROM 3M TO 10M

Camera Distance (m)	Real Distance (m)	Error (m)
3	2.972	0.028
4	3.972	0.028
5	4.924	0.076
6	5.987	0.013
7	6.938	0.062
8	8.035	0.035
9	8.950	0.050
10	9.939	0.061

B. Radar Clustering

Radar cluster is achieved by DBSCAN clustering [5]. The two parameters in the DBSCAN range: (i) Minimum points: higher the points, more the points filtered. (ii) Minimum distance: higher the range, more points in each range causes less clustering points. Through the experimental field test, the minimum points of the DBSCAN is set to 3 and range as 0.40 meters. Fig. 10(a-b) shows the differences before and after the clustering.

(a) (b)

Fig.10 (a) Before clustering, (b) After clustering

C. Radar Tracking

Radar tracking is conducted using the Unscented Kalman Filter (UKF) tracking algorithm [6] as it has non-linear path of radar points and it is of lower complexity.

(a) (b)

Fig.11 UKF Radar tracking results (a) Before the object turning around, (b) After the object turning around.

Initially, the data is associated to tracker with the correct radar clustering point. In data association, Mahalanobis distance [7] is used as the value of tracker and as input whereas, the Hungarian algorithm [8] is used for the data assignment. When the key parameter of UKF, the updated weight coefficient *dt* is low, the tracking of the object is slow. Wide range of tests were carried for different values of *dt* = 0.5~1.4. In the proposed algorithm, *dt* = 0.7 is used as it is experimentally proved to yield the better tracking results. Fig. 11 shows the results of the UKF Radar Tracking with *dt* = 0.7 where the green, red and yellow points indicate the radar points, the clustering points and the tracking points, respectively.

D. Camera Tracking

The object types such as humans, cars and bikes and their corresponding bounding boxes are chosen as the input to the trackers are all obtained by the deep learning processes. Initially, the data association is carried out in order to assign the correct information to the tracker. In data association, the value between the input bounding box and tracker are defined. The *"IoU"* which is the overlap area of the total area is given by (8). The total area is the sum of orange area and green area, and the overlap area is the green area and *bbox* denotes the bounding box as in Fig.12.

$$IoU = \left(\frac{Overlap\ area}{Total\ area}\right) \tag{8}$$

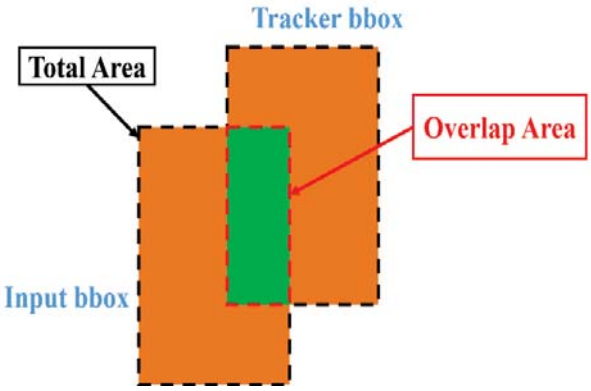
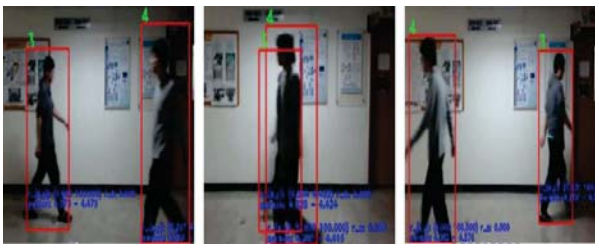


Fig.12 Graphical description of "IoU" function

The input parameters of a bounding box are its width, height and its center point *(x, y)*. The path of an object in a camera is usually linear. Therefore, the Kalman filter [9] is employed as the tracking method. Each tracker has an age parameter to define its life-time. The formula of age trackers is given by (9). When the age of trackers is longer than the maximum pre-defined age, the tracker deletes to ensure the ghost trackers do not exist too long. Camera tracking with Kalman filter enables the tracking of the objects in motion as well as the interleaving objects as in Fig.13 (a-c).



(a) (b) (c)

Fig.13 (a) Multi-person tracking (before interleaving), (b) Multi-person tracking (interleaving), (c) Multi-person tracking (after interleaving)

$$Age\ of\ tracker = \begin{cases} 0, if\ currect\ tracker\ updated \\ 1, if\ current\ tracker\ not\ updated \end{cases} \tag{9}$$

E. Track-to-Track Fusion System

Track-to-Track fusion system is as shown in Fig. 14. The track-to-track fusion block is similar to the one used in [11, 12]. The Radar/Camera calibration is used to project the radar-tracking target with respect to the ground of the image. After generating the bounding boxes from the deep learning model, the points at the bottom of each bounding boxes' center are set as the ground point of each camera tracking a target in an image. Once, both the targets in the image are found, the two local systems output to the main system that uses UKF tracking method. Finally, it outputs the tracking targets of the Track-to-Track fusion system.

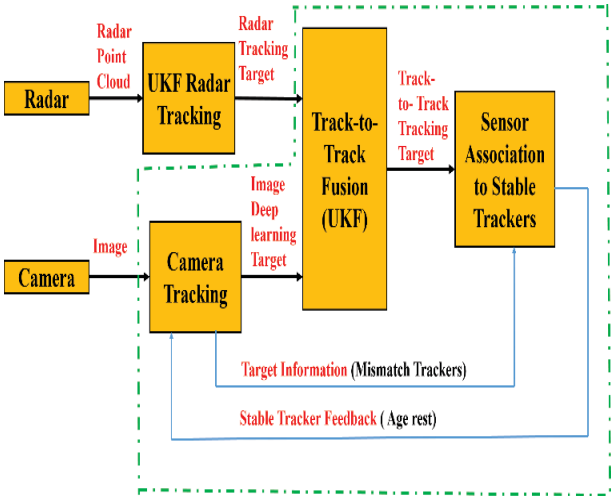


Fig.14 The proposed track-to-track fusion system in green dotted line area

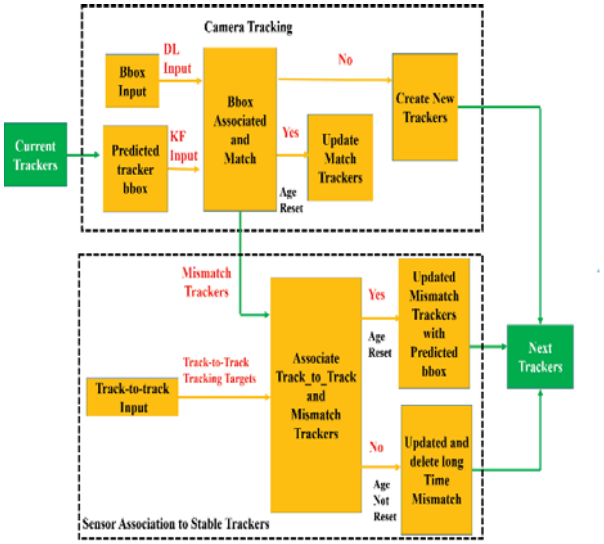


Fig. 15 The detailed flowchart of the proposed track-to-track fusion system

There exists a strong correlation between the *Sensor Association to Stable Trackers* block and *Camera Tracking* block have a strong correlation as shown in Fig. 15. The current trackers are sent to the camera-tracking block to generate the predicted bounding boxes, and deep learning model gives the current detected bounding boxes. Then, both of them are sent to the data association block to decide which predicted bounding boxes from trackers are matched or mismatched with the current detected bounding boxes. After data association, there are three different output conditions generated as follows:

(i) Mismatch of current detected bounding boxes: the new tracking targets are detected in the image during which new trackers are created for the next trackers.

(ii) Matched bounding boxes of both predicted and current detected bounding boxes: the current trackers can be updated with the matched and currently detected bounding boxes to the next trackers and the next trackers are reset.

(iii) Mismatch of current trackers: If only the camera tracking is used, the age of these trackers will be +1. Since the information of Track-to-Track tracking targets can be further improvised, the mismatched trackers from the *Sensor Association to Stable Trackers* block are considered and track-to-track tracking targets as fed as the0 input. Then, both of them are sent to step (i) and (ii).

When mismatched trackers match successfully, the age of the trackers are reset. On the contrary, when the mismatched trackers are still in the mismatched case, the age of the mismatched trackers are allowed to be to set to +1.

F. Optimization

The optimization of the proposed track-to-track fusion system is carried out by the Association Gate method. The purpose of association gate is to filter the faraway Track-to-Track point from the mismatched trackers as in Fig.16. The black rectangle is the bounding box of "mismatch tracker". The Track-to-Track points in green inside the blue area are the legal points and the points in red outside the blue area are the illegal points. The bounding black rectangle is the *mismatch tracker*, *h-ratio* and *w- ratio* are the height and width of the association gate, respectively.

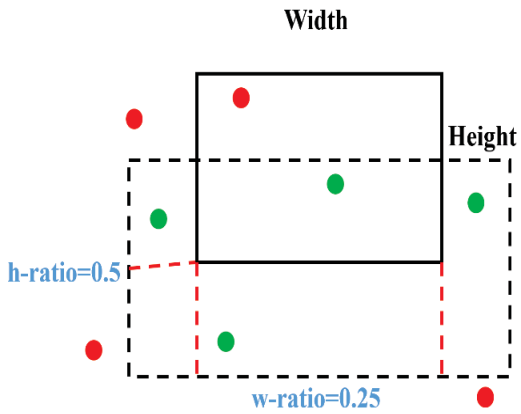


Fig.16. Legal track-to-track (green) points and illegal track-to-track (red) points

For example, if *h-ratio*=0.5, the height of the association gate is half of the bounding box's height from above and below the bottom of the bounding box whereas if *w-ratio* =0.25, the width of the associated gate is extended by 1/4th of the bounding box's width from left and right of the bottom of the bounding box.

The lifetime of each trackers mentioned in section D is a fixed value. However, the fixed lifetime of trackers results in two problems viz.

(i). Ghost trackers occur when the trackers leave the image, or the tracker is too far to be detected by deep learning model. If we use fixed lifetime value, the time of ghost tracker will be on the image for at least the fixed lifetime.

(ii). False positive trackers are generated by the false detection of deep learning model. The fixed lifetime causes the time of false positive trackers become longer.

In order to curb the impact of the above problems, we propose "Track Lifetime Value" (TLV) to replace the fixed lifetime. The goal of TLV function given by (10) is to keep the tracker existing for a long time and to eliminate the short time tracker existence. In TLV function, "$track_{since\_update}$" is the value increasing from the time of tracker updated. The "$track_{since\_update}$" is the number of times the tracker is updated, like the age of the tracker mentioned in the Camera Tracking section. The "$track_{period}$" is the user-defined parameter to let users to choose the lifetime of the TLV. In the proposed method, we set 0.4 as the threshold of the tracker-eliminated value, implying the trackers eliminates the TLV values less than 0.4.

$$TVL = \left(1 - \frac{track_{since\ update}}{track_{period}}\right) * \left(\frac{track_{period} + track_{times}}{2 * track_{period}}\right) \quad (10)$$

Fig. 17 shows a comparison between TLV and fixed frame in the case of the "*Ghost Tracker*" and "*False Positive Tracker*". We set a value of 30 in the "$track_{period}$" of TLV and 30 as the age of the "fixed frame" tracker. In the "*Ghost Tracker*", when tracker is not updated from the time of 50 frames, we eliminate tracker in the 63rd frame using TLV, and in the 80th frame using fixed frame. The error in tracker number is reduced by 17 from 80 to 63 in case of the TLV. In the "*False Positive Tracker*", we eliminate tracker in 3 frames using TLV, and in 30 frames using fixed frame. Thus, the error tracker number is reduced by 27 from 30 to 3 in TLV. Therefore, we use *"Track Lifetime Value"* function to replace

the original "fixed frame" function for the better tracking performance.

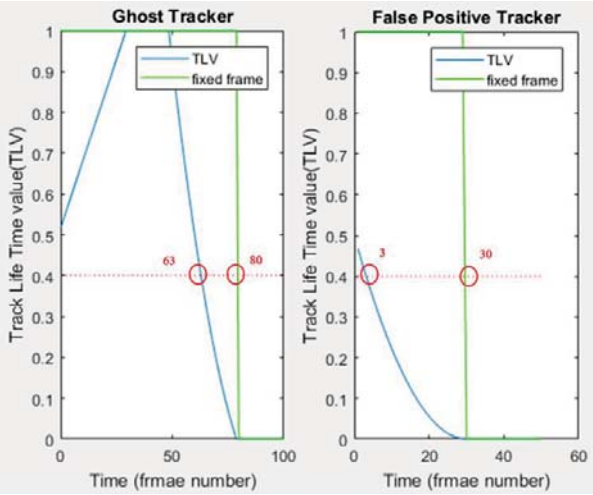


Fig.17. The comparison between TLV (blue line) and fixed frame (green line) in case of "Ghost Tracker" (left) and "False Positive Tracker" (right)

G. Output of Detection and Tracking Result

Through the whole fusion architecture, the final information of the target in the image is as shown in the Fig.18 where *r_(x, y)* is the relative position of *(x,y)* and *r_s* is the relative velocity (v_x, v_y).

Fig.18 Final output of detection and tracking *w*

III. Experimental Results

In the proposed Radar and RGB Camera sensor fusion method, Texas Instruments' *Ti AWR1642BOOST FMCW* radar and Logitech C920 camera are used and the corresponding parameters are respectively listed in table 2 and table 3. Fig. 19 shows the mechanism of the sensor fusion device, in which the radar is at the top and camera at the bottom of the radar. The sensor fusion device is on a retractable bracket, whose height can be adjusted from 1.2m to 2.8m and the angle can be adjusted ranging from -90° to +90°.

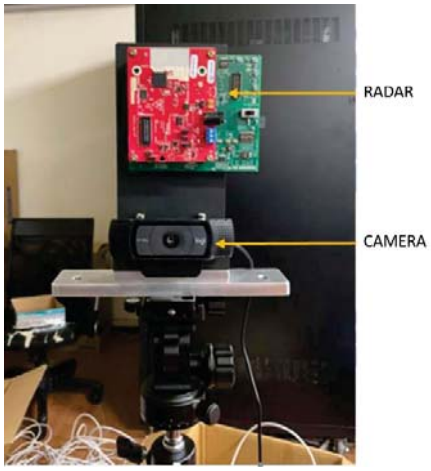


Fig.19 The mechanism of the proposed sensor fusion device

TABLE 2. TI AWR1642BOOST FMCW Parameters

Frequency	**No. of Receivers**	**No. of Transmitters**	**Max. Sampling rate**	**IF Bandwidth**
76-81 GHZ	4	2	12.5 Msps	5 MHz
Processor	**Memory**	**RF Bandwidth**	**Interfaces**	
ARM Cortex R4F 200 MHz C674x DSP 600 MHz	1.5 MB	4 Ghz	CAN, CAN-FD, I2C, QSPI, SPI, UART	

TABLE 3. Camera Specifications

Diagonal View (FOV)	78°
Horizontal View (FOV)	70.42°
Vertical View (FOV)	43.3°
Resolution	360p, 480p, 720p, 1080p
Frame Rate	30 FPS

Table 4 lists the performance parameters of the AWR1642BOOST FMCW radar in Ultra Short Range Radar (USRR) mode. The update rate of the radar is about 15-20 fps, and each update concludes for about 30-40 radar points. Every radar point has the information of relative position, relative velocity, and peak value of the object.

TABLE 4. TI AWR1642BOOST FMCW Radar in USRR Mode

Parameter	**USRR**
Max Range	20 m
Range Resolution	4.3 cm
Max Velocity	36 km/h
Velocity Resolution	0.32 m/s

The training processes of the proposed method is carried out on a PVANET-lite as well as on a SSD-lite Multi-head 512x256 models on a system with NVIDIA GTX1080Ti and parameters are as in Table 5. The classes of training data are humans, cars, and bikes.

TABLE 5. SPECIFICATIONS OF THE SYSTEM USED FOR TRAINING

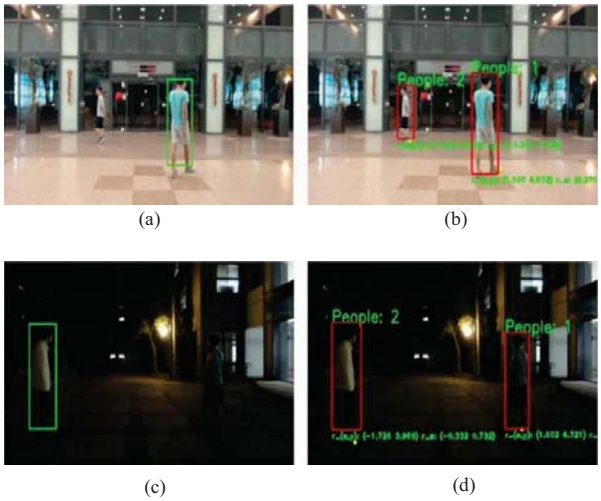
CPU	Intel(R) Core(TM) i7-7700 CPU @ 3.60GHz
GPU	NVIDIA GTX1080Ti
DRAM Memory	64GB
CUDA Version	CUDA-9.0
Operating System	Ubuntu 16.04(64 bits)

The proposed algorithm is also implemented on an embedded system as per the prerequisite of real-time applications. NVIDIA Jetson Xavier, the embedded system that is pre-flashed with linux environment, enables the proposed method to be implemented in a low-power system with the high computational performance and makes it suitable for the real-time ADAS applications. The specifications of NVIDIA Jetson Xavier is shown in table 6. It has achieved about 10 FPS with 77 GHz radar input and 640 X 360 image input.

TABLE 6. SPECIFICATIONS OF THE NVIDIA JETSON XAVIER SYSTEM USED FOR TESTING

CPU	8-Core ARM v8.2 64-Bit CPU, 8 MB L2 + 4 MB L3
GPU	512-Core Volta GPU with Tensor Cores
DRAM Memory	16 GB 256-Bit LPDDR4x | 137 GB/s
CUDA Version	CUDA-10.0
Operating System	Linux4Tegra (basically Ubuntu 18.04 with pre-configured drivers), 64-bit
I/O	HDMI x 1, USB3.0 x 1, RJ45 x 1…

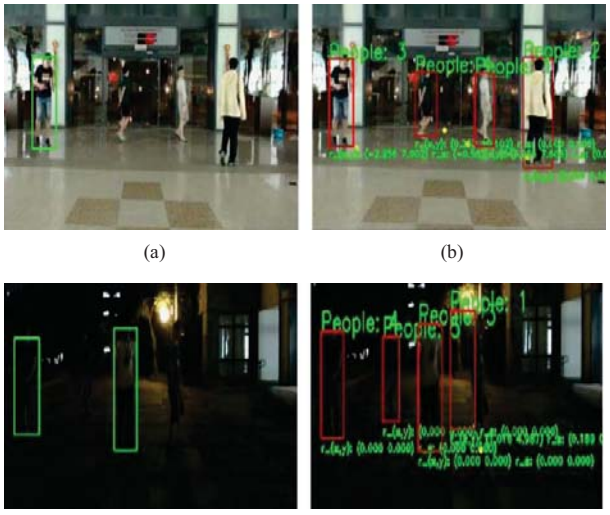
The experiments were conducted in two conditions such as day and night hours with (i) one person, (ii) two people, (iii) three people, and (iv) four people, all moving in free style. The proposed method is implemented using SSD-lite and PVANET-lite methods. The yellow points are the radar association points whereas *r_(x, y)* is relative position and *r_s(x, y)* is relative velocity reported from radar, respectively.



(a) (b)

(c) (d)

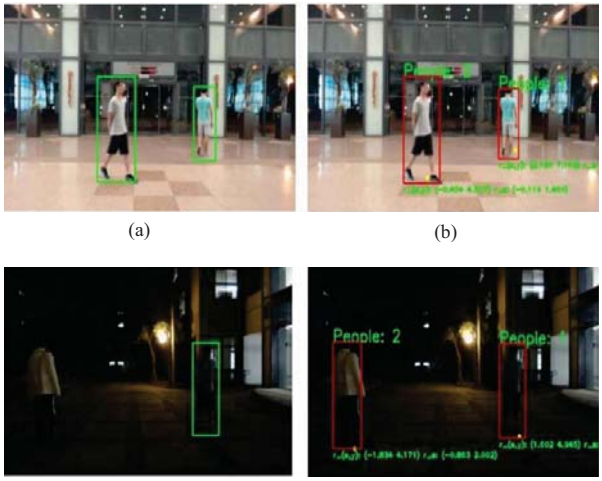
Fig. 20. Two people (a) SSD-lite (bright scene) (b) Proposed method (bright scene) (c) SSD-lite (dark scene) (d) Proposed method (dark scene)



(a) (b)

(c) (d)

Fig. 21. Four people (a) SSD-lite (bright scene) (b) Proposed method (bright scene) (c) SSD-lite (dark scene) (d) Proposed method (dark scene)



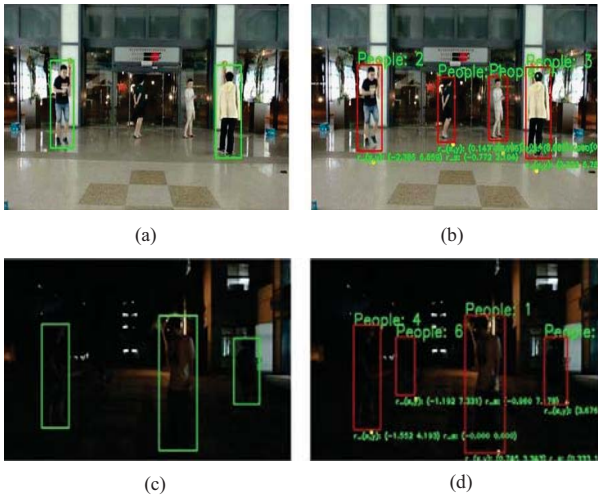
(a) (b)

(c) (d)

Fig. 22. Two people (a) PVANET-lite (bright scene) (b) Proposed method (bright scene) (c) PVANET -lite (dark scene) (d) Proposed method (dark scene)

The results of the proposed method in comparison to the SSD-lite method are as in Fig.20 (a-d) and Fig. 21 (a-d) for two and four people, respectively. Similarly, the results of the proposed method in comparison to the PVANET-lite method are as in Fig.22 (a-d) and Fig. 23 (a-d) for two and four people, respectively.

The comparison of the proposed method with that of the SSD-lite model and PVANET-lite model is as tabulated in the table 7 and table 8, respectively. Compared to the pure SSD-lite, our proposed method with SSD-lite improves accuracy by 19% in bright scene, and by 28% in dark scene with a little precision drop. Likewise, corresponding to the pure PVANET-lite, our proposed method with PVANET-lite improve 14% accuracy in bright scene, and 29% in dark scene with a little precision drop.



(a) (b) (c) (d)

Fig. 23. Four people (a) PVANET-lite (bright scene) (b) Proposed method (bright scene) (c) PVANET -lite (dark scene) (d) Proposed method (dark scene)

TABLE 7. COMPARISON OF THE PROPOSED SENSOR FUSION ALGORITHM WITH THAT OF SSD-LITE MODEL

Method	SSD-Lite	Proposed Method (SSd-Lite)	SSD-Lite	Proposed Method (SSd-Lite)
Scenario	Bright Scenes		Dark Scenes	
Total Frames	9609		8412	
Precision (%)	99.63	98.84	99.25	95.21
Accuracy (%)	78.75	97.71	65.46	93.60

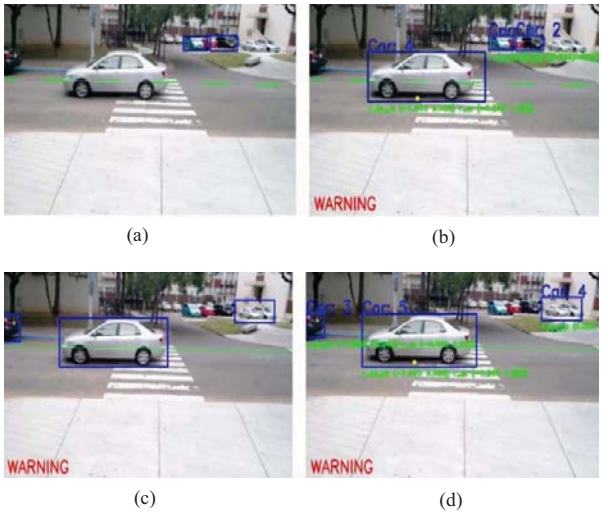
TABLE 8. COMPARISON OF THE PROPOSED SENSOR FUSION ALGORITHM WITH THAT OF PVANET-LITE MODEL

Method	PVANET-lite	Proposed Method (PVANET-Lite)	PVANET-lite	Proposed Method (PVANET-lite)
Scenario	Bright Scenes		Dark Scenes	
Total Frames	9609		8412	
Precision (%)	99.98	99.11	100	98.47
Accuracy (%)	84.11	98.23	66.41	95.91

Fig. 24 shows the comparison of SSD-lite, PVANET-lite and the proposed method to detect the vehicles in the real environment.

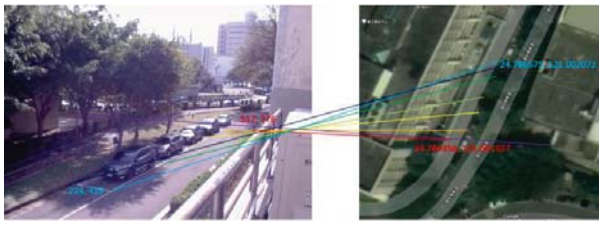
Furthermore, the proposed method can also be employed in estimating the speed of the objects in motion by computing a homography matrix between camera coordinates and GPS coordinates as represented by (11) and fig.25. Fig. 26 is an example of the computation of the object speed moving at 5.56m/s, which is estimated as 6.19 m /s with an error rate of 0.63 m /s. The estimation of different speeds of objects by the proposed sensor fusion method is tabulated in table 9.

$$\begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} H_{00} & H_{01} & H_{02} \\ H_{10} & H_{11} & H_{12} \\ H_{20} & H_{21} & H_{22} \end{bmatrix} * \begin{bmatrix} lat \\ lon \\ 1 \end{bmatrix} \qquad (11)$$



(a) (b) (c) (d)

Fig.24 Vehicle warning (a) SSD-lite (b) Proposed method with SSD-lite (c) PVANET-lite (d) Proposed method with PVANET-lite



(a) (b)

Fig.25 (a) Camera coordinates (pixel location x, y), (b) Bird-view coordinates (GPS in decimal degree)

Fig. 26 Estimation of the speed of the object moving at 20 km/h (5.56m/s) by the proposed sensor fusion method.

TABLE 9. COMPUTION OF THE OBJECT SPEEDS BY THE PROPOSED SENSOR FUSION METHOD

Avg. speed (Ground Truth)	Avg. speed (estimated)	Absolute error Abs [col 1 – col 2]	Error %age [(Col 3/Col 1]X100]
20km/hr (5.56m/s)	6.19m/s	0.63m/s	11.38%
25km/hr (6.94m/s)	7.93m/s	0.99m/s	14.30%
30km/hr (8.33m/s)	8.45m/s	0.12m/s	1.39%
35km/hr (9.72m/s)	9.62m/s	0.10m/s	1.06%
40km/hr (11.11m/s)	11.47m/s	0.36m/s	3.25%
45km/hr (12.5m/s)	13.12m/s	0.62m/s	4.98%
50km/hr (13.89m/s)	14.88m/s	0.99m/s	7.12%

IV. Conclusion

The proposed method in this paper focuses on the enhancing the accuracy of detection and tracking of objects by sensor fusion method that combines each sensor's positive features. The proposed method improves the efficiency of the object detection and tracking by achieving an overall efficiency of around 95% accuracy and 97% precision as in table 10. Thus, it increases the efficiency by about 15% accuracy with about only 2% precision loss compared to deep learning models. Our method also offers additional object numbering, relative position and relative velocity. Since the radar sensor is not affected by light intensities, our experiments have proved that the sensor fusion enables the object tracking and detection even in dark scene.

TABLE 10. Overall Comparision of the Results

Method	**SSD-lite**	**Proposed method (SSD-lite)**	**PVANET-lite**	**Proposed method (PVANET-lite)**
Total Frames	27278			
Precision (%age)	99.49	96.57	99.99	98.74
Accuracy (%age)	79.00	94.22	82.26	96.27

Acknowledgment

This work is partially supported by the *"Center for mmWave Smart Radar Systems and Technologies"* under the *"Featured Areas Research Center Program"* within the framework of the Higher Education Sprout Project by the Ministry of Education (MOE), Taiwan R.O.C. It is also partially supported under Ministry of Science and Technology (MOST), Taiwan R.O.C. projects with grants MOST 108-3017-F-009-001 and MOST 109-2634-F-009-017 through Pervasive Artificial Intelligence Research Labs (PAIR Labs) in Taiwan, R.O.C.

References

[1] Gao, Dezhi, et al. "A method of spatial calibration for camera and radar." 2010 8th World Congress on Intelligent Control and Automation. IEEE, 2010.

[2] Luo Wei, Yao Yuan, and Zhang Jinchang, "A method for joint calibration of millimeter wave radar and camera." Journal of Tsinghua University: Natural Science Edition 3 (2014): 289-293.

[3] Zhang, Zhengyou. "A flexible new technique for camera calibration." IEEE Transactions on pattern analysis and machine intelligence 22 (2000).

[4] Caprile, Bruno, and Vincent Torre. "Using vanishing points for camera calibration." International journal of computer vision 4.2 (1990): 127-139.

[5] Ester, Martin, et al. "A density-based algorithm for discovering clusters in large spatial databases with noise." Kdd. Vol. 96. No. 34. 1996.

[6] Wan, Eric A., and Rudolph Van Der Merwe. "The unscented Kalman filter for nonlinear estimation." Proceedings of the IEEE 2000 Adaptive Systems for Signal Processing, Communications, and Control Symposium (Cat. No. 00EX373). IEEE, 2000.

[7] Taguchi, Genichi, and Rajesh Jugulum. The Mahalanobis-Taguchi strategy: A pattern technology system. John Wiley & Sons, 2002.

[8] Kuhn, Harold W. "The Hungarian method for the assignment problem." Naval research logistics quarterly 2.1 - 2 (1955): 83-97.

[9] Ljung, Lennart. "Asymptotic behavior of the extended Kalman filter as a parameter estimator for linear systems." IEEE Transactions on Automatic Control 24.1 (1979): 36-50.

[10] Kim, Kyeong-Eun, et al. "Sensor fusion for vehicle tracking with camera and radar sensor." 2017 17th International Conference on Control, Automation and Systems (ICCAS). IEEE, 2017.

[11] Fröhle, Markus, et al. "Multisensor Poisson Multi-Bernoulli Filter for Joint Target Sensor State Tracking." arXiv preprint arXiv:1712.08146 (2017).