Using Radar and Vision Fusion for Improved Low-light Pedestrian Detection

by Dr. Peter Gulden and Dr. Zorawar Bassi

Abstract

Using multiple sensor modalities is becoming a critical design requirement in automotive safety systems. The reasons are multi-fold, such as having redundancy, robustness to adverse weather conditions, challenging environment scenes, and improved accuracy due to complementary features. The placement and interaction, or fusion, of these sensors, can vary widely. We present some results from combining radar and image (vision) sensors, using low-cost processors that can be placed in various locations throughout the vehicle. This could be at the edge or in some zonal configuration intermediate to a center topology. The sensors are fused for object detection, which is far more robust under various environmental conditions, than if a single modality was used. Further, the unique indie approach for combining several radars leading in a cooperative way to enhance the number of detection points and full vectoral velocity capability is showcased. Our focus for this investigation was pedestrian and bicycle detection in an automotive backup scenario, under low light conditions. While sensor fusion has required powerful computing resources to-date, we show how vision and radar fusion can be accomplished with modest processing power to help bring the safety benefits of multi-modal sensing to mass-market vehicles.

Introduction

Today's vehicles rely on numerous sensors for safety reasons, as well as to accommodate various levels of automation; ultimately this automation is also aimed at increasing safety and providing a more efficient driving experience. Camera image sensors are one of the most widely used types of sensors, given that they most closely resemble human vision – vision being the most fundamental human capacity used for driving. Like human vision, cameras can be easily impaired, whether due to challenging environmental conditions, such as low light or fog, or simply because the lens may be obstructed by dirt. Hence cameras are often paired with other types of complimentary sensors, such as radar, which are immune to those conditions that can impair a camera. Radar sensors have their own strengths over image sensors, such as providing accurate distance and velocity data. The fusion of a camera with radar can enhance a safety system by improving accuracy and robustness, as well as providing redundancy.

Perhaps the most widely used camera-based safety system, and one of the first to be mandated by regulations, is the backup camera. As the development and funding of advanced vehicle AI systems, with the goal of full autonomy, has greatly increased over the last decade, the simple backup camera continues to be the most widely used and available safety system. Its simplicity allows it to be understood and adopted by drivers of all generations. In recent years, the backup camera has expanded to include smarts, or some level of AI, where it can detect objects in its view, which in turn can signal an emergency braking feature. Adding smarts, where the camera provides a decision or 'opinion' on safety, as opposed to simply a picture, greatly increases the need for accuracy and robustness. This is even more true in tough environment conditions (nighttime), where a driver's own vision may be challenged. Being a relatively simple safety system to be fitted inconspicuously in a compact area, a backup camera is required to be low cost, low power, and small size. These requirements often conflict with the smart features, which tend to necessitate large sensors and additional processing ICs. The conflict is exasperated by the requirement of high accuracy and robustness, which in turn needs more 'smarts', i.e., more processing power.

In our paper, we return to this most fundamental safety system, the backup camera, with some basic smarts, though our results are equally applicable to any “smart” viewing camera around the vehicle (e.g. a side mirror camera system). Such smart backup cameras have been available for some time, however, their performance under tough environmental conditions, given their low cost, is often suboptimal, limiting the features (such as audio warning, and emergency braking) that can be activated. We study these cameras, with the smarts of detecting pedestrians and bicycles, under low light (<10 lux), where a human may also have difficulty detecting. We then combine the camera with a radar sensor using a simple fusion approach and show how performance is improved. Extensive work has been done on multi-sensor fusion, including camera and radar fusion; see [1] for a review. The majority of work focuses on the algorithm and architecture, often using complex machine learning models more fit for high autonomy, without consideration of hardware implementation and cost. A key objective here was to keep the system cost low, keeping in mind the low to mid-priced vehicle. Hence our approach is such that all processing can be done on the integrated processor inside the camera, no additional processors are needed for the fusion. Aside from the camera, the only additional hardware is the radar front end. As the trend for vehicles to make use of multiple sensor types grows, many are also including radar systems, such as corner radars. We make use of the same data from any existing vehicle radar system, so as not to add a radar specific for the smart backup application and keep the cost low. Our overall pipeline has components similar to [2], however, those authors are focused on a general framework, treating the camera and radar as independent detectors, making use of a more complex probability model for fusion, independent of any hardware considerations, whereas we start with a small footprint device, and build our architecture accordingly, taking only the vision system as the detector. We also present a new approach for combining several radars leading in a cooperative way to enhance the number of detection points as well as provide full vectoral velocity. The increased detections and full velocity can improve the radar clustering and in turn the fused detection.

Our paper is organized as follows. We begin with a discussion of the camera system, in particular the processor inside the camera on which all computation is done. The right processor and choice of algorithms are critical to keeping system cost/power/size low. Next, we present our camera smarts, namely the vision-based pedestrian and bicycle detector. This is followed by a discussion of the radar processing pipeline in the section, “Radar Pipeline and Clustering”. Our fusion approach is outlined in section, “Simple Fusion Algorithm”, and the results of the fusion showing improved performance are presented in section, “Experiments and Results”. “Cooperative Radar” section covers our unique radar offerings, followed by conclusions and future improvements.

Camera Processor

The standard backup camera consists of a lens, an image sensor, and some form of a system-on-chip (SoC), which we call the camera processor (see Figure 1). The camera processor receives raw data from the image sensor and performs various functions to output a live video stream. indie Semiconductor offers several advanced camera video processors geared towards the automotive industry, and for the purposes of this paper, we have worked with the GW5x SoC and GW6x SoC (currently in development), product lines (see [3] for more information). These processors are highly efficient, low power SoCs, targeted for exterior automotive cameras. Figure 1 shows key internal blocks.

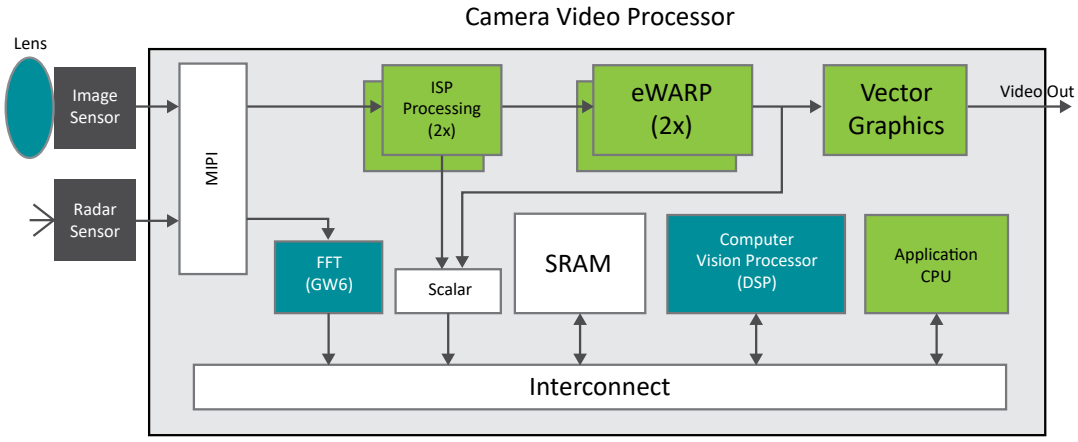


Figure 1: indie Semiconductor’s GW5x/GW6x camera video processor block diagram.

The GW5x does not include the FFT block, all other blocks are common to both GW. x and GW6x, with variations in features and performance. Raw data from the image sensor comes in through the MIPI interface, it is processed by the image sensor processing (ISP) block (this does de-Bayer, noise reduction, HDR, auto-exposure, gamma, etc.) to create RGB data. The

eWARP block is a custom warping engine, which can apply complex geometric transformations to the video, followed by any OSD/drawing through the vector graphics, lastly, the video is output. Internal static random-access memory (SRAM) which ranges between 3 and 6 Mbytes, fulfills all system memory needs. At various points frame(s) can be grabbed, scaled (Scalar) and written to memory. The Computer Vision Processor is a highly parallelized, Single Instruction/Multiple Data (SIMD) type, DSP block, which performs all the computer vision processing, like object detection. A main processor, the Application CPU, performs the control functions, and can also be used for running algorithms. A second MIPI interface allows taking in a raw radar signal, which is processed through a hardware FFT block (in GW6x), the radar cube is then written to SRAM for additional processing by the Computer Vision Processor. The GW6x family includes 2x independent copies of the ISP and eWARP blocks, this allows independent ISP tuning and geometry transforms for the viewing video stream, and the video used for computer vision processing.

The indie processors have a small footprint in all aspects, with a size of ~10 x 10 mm and power ranging from 0.5 to 1.0 W, depending on resolution and features enabled. This makes them ideal for camera solutions on the edge. With the ability to take in a second input, of possibly different modality, they can also support specific fusion applications. All our software algorithms for this project were run either on the GW5x IC, or on the bit accurate simulator for the upcoming GW6x IC. The GW6x family also has a configuration with external DDR support, however we chose not to enable that configuration in order to keep the footprint minimal – our development did not rely on any external memory.

Vision Based Object Detector

In a time where automotive AI ubiquitously uses larger and larger models, rooted in deeper and deeper machine learning, with big neural nets (NN) dominating computer vision, we dared to resort back to a classical non-NN approach. The objective again is to make things as small as possible – whereas gigabytes of memory is common for central computing, we worked with merely 3-6 megabytes for an edge solution. In line with this, our vision-based detector is a GW5x/GW6x hardware-optimized version, based on pre-NN classical approaches of [4, 5]. It makes use of aggregate channel features (ACF), a multi-scale pyramid, combined with cascaded decision trees, trained using AdaBoost, for inference.

The detector is depicted in Figure 2 and runs primarily on the DSP sub-processor.

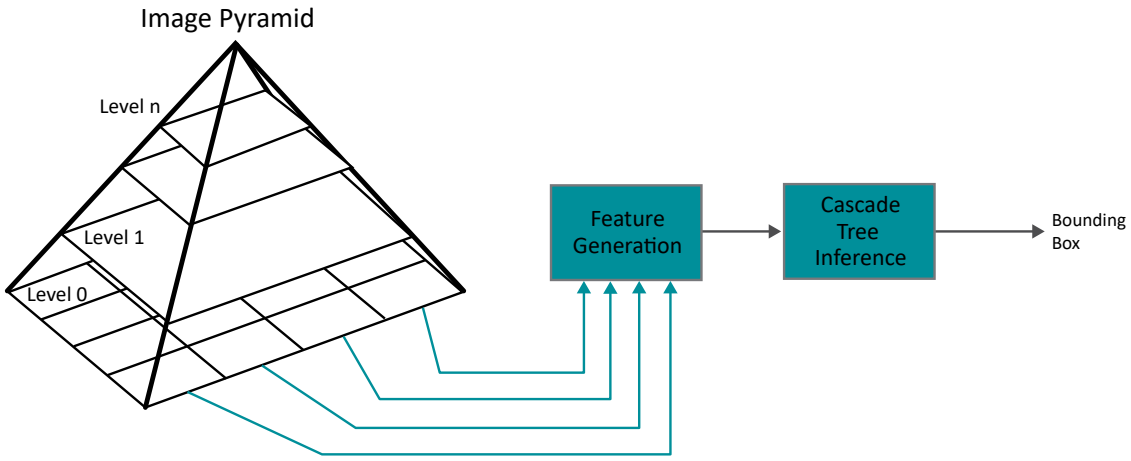


Figure 2: ACF vision detector.

Various optimizations were necessary to ensure all model data and intermediate results would fit within the on-board SRAM. Given all other processing needed, such as implementing the radar pipeline, this meant limiting the detector memory footprint to 1MB! We briefly summarize the main steps and optimizations. Processing of the image made use of memory tiles, which allowed efficient reading/writing of data. Only a subset of pyramid levels was computed, the others being obtained by scaling of the computed levels [6]. All computed levels themselves were processed in three steps, corresponding to the left, middle and right of the image, each step done for all computed levels, before moving to the next step. Subsequent pyramid levels overwrote previous levels to conserve memory. The ACF feature descriptor was limited to 8 channels: Y - the pixel luminance value, |M| - the gradient magnitude, H1 to H6 – a 6 bin histogram of oriented gradients. As for the pyramid, sub-sampling was also used to reduce the ACF feature vector size for an image window. An image window size of 64x128 was used and typically feature subsampling of 2.

The fundamental component of the cascaded inference stage was the binary tree shown below, along with the node descriptor and the operation at a node [7]:

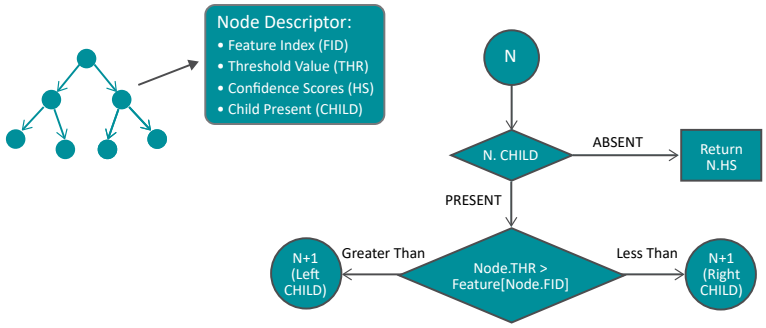


Figure 3: Binary Decision Tree.

At each node, the feature value is compared with the node threshold. A tree is traversed until no child node is present. The final confidence of the tree is compared with a threshold, as well as aggregated across multiple trees. Figure 4 shows the aggregation and the multiple tree structure, note the threshold values are determined during model training.

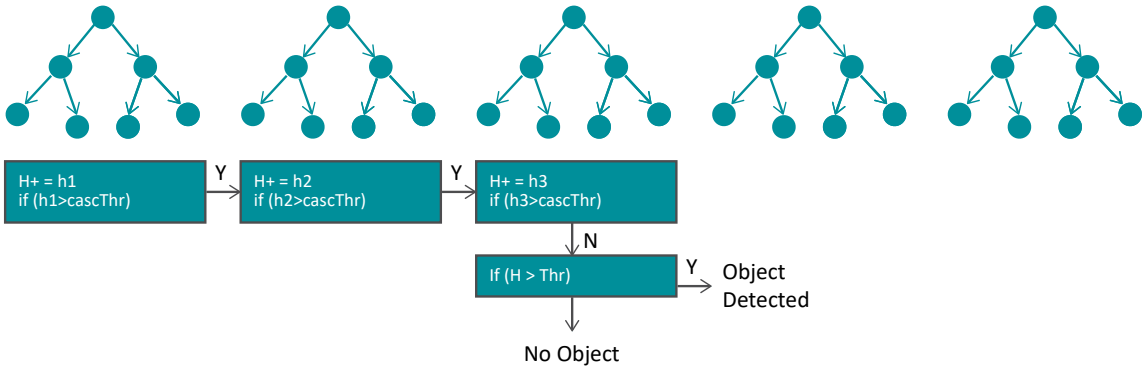


Figure 4: Cascaded inference across multiple trees.

In general, the trees are not of uniform depth. A critical step in parallelizing the inference computation was to use trees all of the same depth. Based on our testing, we settled on a depth of 3, and approximately 2560 trees in total. These were trained with ~40 to 50K images. The output of the inference stage is a set of bounding boxes, often overlapping, and confidence scores. These boxes are reduced in number by using non-maximal suppression to retain only those with the highest confidence. The scores are normalized via sigmoid function to give a confidence value between 0 and 1, upon which a final decision can be made.

Other important steps in the processing include warping, scaling and brightness adjustment. The lens for a backup camera is typically a fisheye lens (~180 degrees), whereas the detector expects perspectively corrected images. The eWARP© block, indie's proprietary geometric transformation technology performs the perspective correction using a proprietary low latency, no frame store, architecture. The camera resolution is also large, in our case 1920x1080, which to conserve memory was scaled and cropped to 640x360, before processing by the computer vision processor. This can be again done by the eWARP block or the Scalar block. The eWARP block can also be used to create some of the pyramid levels, as different windows within a large frame. For detection in low light, experiments showed improved accuracy by S-curve brightness adjustment prior to running the detector. This could be done through the second ISP block for GW6x, without changing the viewing video stream. It can also be implemented on the DSP.

Our detector takes 40-60ms to execute, per frame, depending on content and number of trees, and requires ~1MB of memory.

Radar Pipeline and Clustering

The radar pipeline is the standard set of steps for extracting a point cloud from a Frequency Modulated Continuous Wave Radar (FMCW) radar [8], followed by a clustering stage, and lastly mapping to a common coordinate system. These steps are shown in Figure 5.

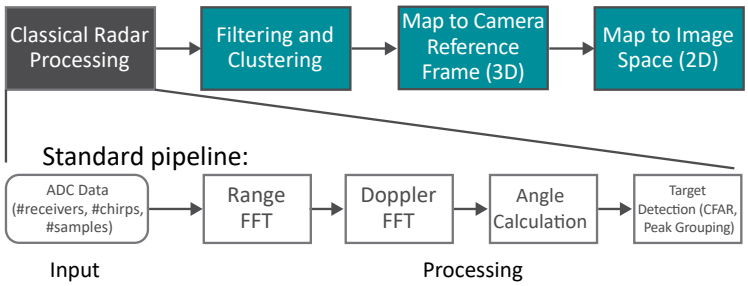


Figure 5: Radar processing and stages.

A 12-channel (4 receive x 3 transmit) FMCW radar was used. The ADC radar data can be brought in through the second MIPI input of the video processor. The most computation and memory-intensive steps here are the FFTs. The GW6x includes an FFT hardware block that performs both the range and Doppler FFTs. On GW5x, the DSP can be used for the FFTs, though this takes away cycles from the vision processing and reduces the overall FPS (frames per second) of the system. A total of ~3MB of memory is used by FFT computation for our system. The "cube" from each FFT stage overwrites the previous stage's data. The range FFT consists of #chirps x #channels 1D FFTs, and the Doppler FFT consists of at most #samples x #channels 1D FFTs. Figure 6 illustrates the FFTs and data cubes. For these sizes, the hardware FFT completes in ~20ms.

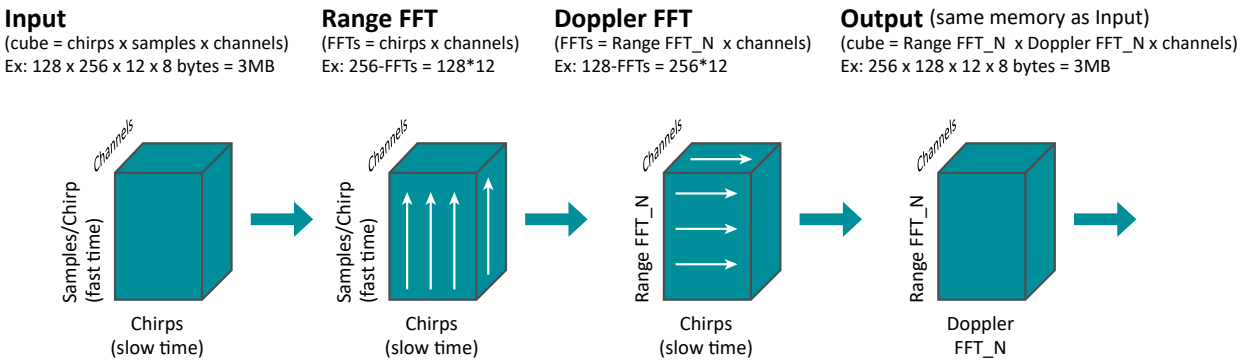


Figure 6: FFT data and processing.

The remaining stages, angle computation and target detection (CFAR, peak grouping) are less intensive, and can be performed on the DSP, without impacting substantially the vision detection processing. For a DSP speed of 1GHz, these stages can run in 5-10ms. The output of the standard pipeline is a cluster of points, each point a 5-vector of position [x, y, z], radial velocity, and intensity.

The point cloud is next filtered and clustered. The filtering eliminates specific points depending on their position, velocity and intensity, which are not likely to be a nearby object of interest. For example, points outside of 25m or >3m above the ground are not retained. The filtered points are clustered using the OPTICS algorithm [9], which is similar to the popular DBSCAN approach, but allows varying density using a variable neighborhood radius. For a typical backup scenario, the number of points to be clustered ranged from 50-100, hence the computational load for this stage is small. As the point vector contains three different units (3D position, velocity and intensity), clustering was done on the three metrics independently, and then combined afterward. The clusters obtained are also assigned labels if they are potential candidates for objects (pedestrians or bicycles) or not – this is a binary value of 1 or 0. Note these labels are not confidence scores, as no learning is done on the radar clusters. They are from an additional screening step, based on heuristics (such as intensity and spatial-related metrics), to further reduce the number of useful clusters. Those with label 0, can be dropped. The clustering stage runs on the DSP (computer vision processor) and completes in <5ms.

The last two stages map the clusters to the correct frames in anticipation of the fusion stage. First, the 3D positions are mapped to the camera's 3D reference frame using 3D rotation and translation, which is obtained from the radar to camera extrinsic calibration. Lastly, to align the radar points with the detector bounding boxes, which sit in the perspective-

corrected image, the 3D cluster points (in the camera's reference frame) are mapped using the same perspective transformation that was applied by the eWARP engine to correct the camera fisheye input. Now the radar points lie in the same pixel image that contains the bounding boxes. As these mappings are simple math functions applied to a small number of points, the computation time is negligible. Combining the times from all stages, the radar processing completes in ~30-40ms on our GW6x processor (on GW5x without the hardware FFT, the time is longer), with a memory footprint of ~3-4MB.

A word on calibration. For fusion, the vision-radar system must be calibrated, meaning the camera's intrinsic parameters are known, and the radar is extrinsically calibrated relative to the camera [10]. Various calibration software routines are available for this purpose. As an example, for camera intrinsic calibration, multiple images of a checkerboard pattern can be fed through the ROS camera calibration process. This will provide the required camera lens distortion model and the projection matrix. For extrinsic, a good radar corner reflector is placed in the overlapping FOV of both camera and radar, and several frames are recorded. The radar point from the reflector is mapped to the camera image, using the camera intrinsics and un-tuned extrinsics (3D rotation and translation) – say an identity. Then the extrinsics are tuned to have the radar reflector point fall on the reflector itself in the camera image. This is illustrated in Figure 7 below.

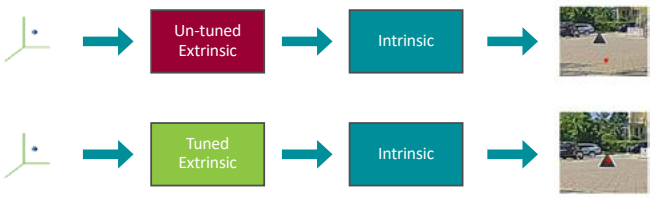


Figure 7: Camera-Radar extrinsic calibration.

The process can be automated by having the camera automatically detect the radar corner reflector. The frames being fused should also be calibrated, or synchronized, in time. In the backup scenario we were studying, there were not any fast-moving scenes or objects. Furthermore, our radar FPS is relatively low (5 fps) compared to that of the camera (60 fps). Hence, we did not perform any temporal calibration, we simply selected camera frames closest in time to radar frames.

Simple Fusion Algorithm

Having obtained the bounding boxes with their confidence levels, and the radar 2D points with their labels, we can now fuse the results. This is a decision-level fusion. Typically, such fusion involves combining two independent probabilistic models, with some common classical approaches including Kalman filter, particle filter and Monte Carlo type methods. However, these methods have high computational complexity. Furthermore, they are Bayesian based, requiring knowledge of prior probabilities, which are often difficult to quantify. Hence, we use the more tractable Dempster-Shafer method to combine the confidence scores [11, 12]. To further simplify matters, we take the non-conventional step of starting with vision-based detection as the main probability and having the radar detection supplement or confirm it. In particular, the radar mass functions are tied to the vision bounding boxes, and the combined scores are the revised confidences for the vision bounding box.

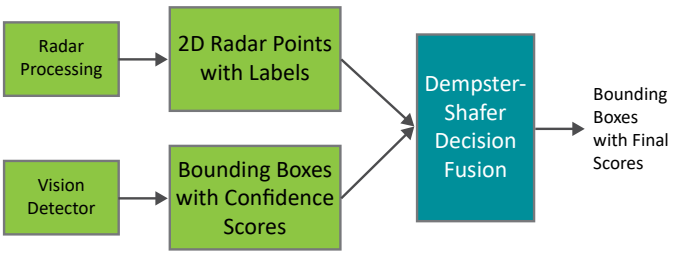


Figure 8: Simple decision-based fusion.

We provide some background of the theory as needed for our application. Dempster-Shafer is a general scheme for expressing uncertainty, whereby a set of propositions, instead of individual propositions, is assigned a probability-like quantity called mass value, ranging between 0 and 1. Let X be the universe of all the possibilities: $X=\{PD, NO\_PD\}$, here PD stands for pedestrian (or any other object trained for) detected, and NO_PD stands for pedestrian not detected. The power set is: $2^X=\{\phi,\{PD\},\{NO\_PD\},X\}$. A mass function m(x) maps elements from the power set to a value in between 0 and 1 $m:2^X \rightarrow [0,1]$ such that $m(\phi) = 0$, $\sum_{A\epsilon 2}{}^X m(A) = 1$. Two different mass functions of the same universe can be

$$m_{1,2}(A) = \frac{\sum_{B\cap C=A} m_1(B)m_2(C)}{1-K} \quad \text{where} \quad K = \sum_{B\cap C=\emptyset} m_1(B)m_2(C)$$

When two mass functions are in conflict, a metric called credibility is used to assign weights to them and combine using the above equation. We do not consider conflicts, as the radar mass function is treated as supplementary to the vision function. We define the mass functions corresponding to the two sensors as follows:

- **Vision mass function:**

  $m_V(\phi) = 0,\ m_V(PD)=\beta_{PD},\ m_V(NO\_PD)=0,\ m_V(X) = 1-\beta_{PD}$

  where β_{PD} is the normalized vision (ACF) detector confidence score.

- **Radar mass function:**

  - When there are no points inside the bounding box:

    $m_R(\phi) = 0,\ m_R(PD)= 0,\ m_R(NO\_PD) = 1-\alpha_{MD},\ \ m_R(X) = \alpha_{MD}$

    where α_{MD} is the miss-detection (false-negative) probability.

  - When there are N points inside the bounding box:

    $m_R(\phi)=0,\ m_R(PD)=1-\alpha_{FD}{}^N,\ m_R(NO\_PD)=0,\ \ m_R(X)=\alpha_{FD}{}^N$

    where α_{FD} is the false-detection (false-positive) probability.

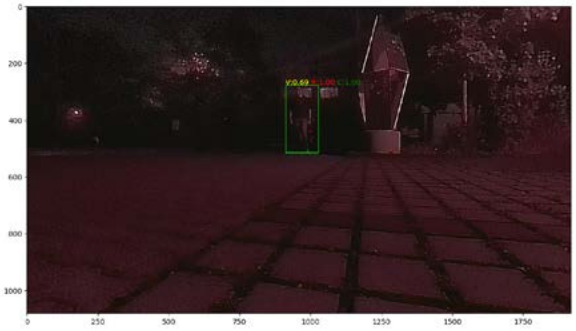
Both α_{MD} and α_{FD} are learnt values – ideally, they should be determined by training with the radar clusters against known object clusters. They may also be estimated to some extent from the cluster binary labels. However, given the limited data we had, this was not feasible, hence we used a conservative value of 0.5. It is important to note that our radar mass function is not truly an independent probability model, as it depends on the vision bounding box result – it is being used to supplement the vision result.

For each bounding box, the vision mass functions are computed using the vision confidence score (β_{PD}). The number of radar points are determined inside the box: if zero, the radar mass functions are computed using the first rule, if $N > 0$, the radar mass functions are computed using the second rule. Last the combined mass function $m_{V,R}(PD)$ is computed using the Dempster rule, to give the combined, or fused, probability that the box contains an object of interest – in our case a pedestrian. This fusion procedure, also running on the DSP, is simple and its impact on the processing time (or memory needs) is inconsequential.

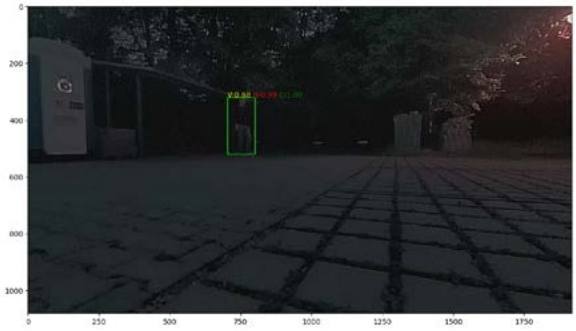
Experiments and Results

Our fusion system was tested under both bright (daytime) and low (primarily nighttime) light conditions. We found during daytime, the vision detector alone was sufficient to detect pedestrians with good confidence (>=0.9) and low false positives, arguably the radar was not required during daytime. Under low light, the results changed considerably, with the vision confidence being insufficient (<0.8) and a higher false positive rate. Hence, we focused our efforts on low light situations. We note that missed detections (false negatives) by the vision detector, cannot be resolved by our fusion approach, as the radar was not independently trained for detection – with the radar mass functions tied to vision bounding boxes, it serves more to strengthen or weaken a vision detection. This can be improved by expanding the radar pipeline into a detector itself and training it for independent detection. The low light scenarios we studied consisted of correct vision detection with low confidence (<=0.8), and with zero or more false positives – the low confidence and false positives then being resolved by radar fusion. Training of our vision detector included both pedestrians, pedestrians with bicycles and bicycles alone, however, the detection window was kept at the same aspect ratio for simplicity. The images in Figure 9 summarize our results. Here the vision bounding boxes are in yellow or green, the radar points in red (a little hard to see), the vision confidence (from the ACF detector) is written above in yellow as V, the radar mass value, $m_R(PD)$, is written in red as R, and the combined score, using Dempster's rule, in green as C. If the combined score is >=0.85, we treat it as a successful detection with good confidence and highlight the bounding box green. As in print the score may be hard to see on the image, they have also been captioned below each image for the green boxes.

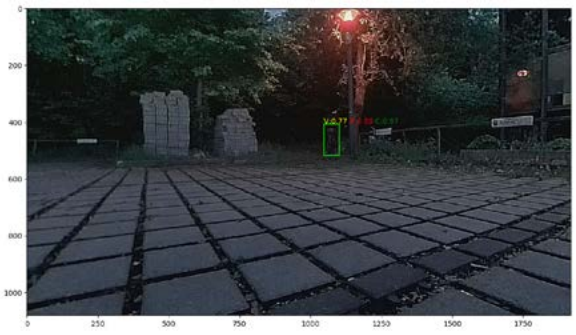
We have retained the full field of view (FOV) of the perspective image for display, to show the challenges in detecting under low light. In some images, as in 9a and 9b, the pedestrian is difficult to discern even with our natural human vision. Figures 9a to 9c show common cases where the vision detector locates an object, but the confidence score is low (<0.8). Once fused with the radar cluster points (small dots in red), any ambiguity is resolved, and we have a strong detection. It's hard to see but 9c is a person on a bicycle, in a difficult background – it is also successfully detected by our system. In 9a points from another cluster (white dots in top left half) are also seen, but these are rejected by our filtering during clustering. Figures 9d to 9f show the correct detection along with multiple false positives detections, all of which have the same low vision confidence score. The radar fusion strengthens the correct detection scores to successful detection, and eliminates the false detections, as sufficient valid cluster points are not present in those bounding boxes. Figures 9g and 9h show similar results for a person on a bicycle. Here we have multiple correct detections with low scores, due to the extended nature of the bicycle (recall we use the same aspect detection box), all of which become successful detections with the fusion, and the radar fusion also eliminates the false positives.



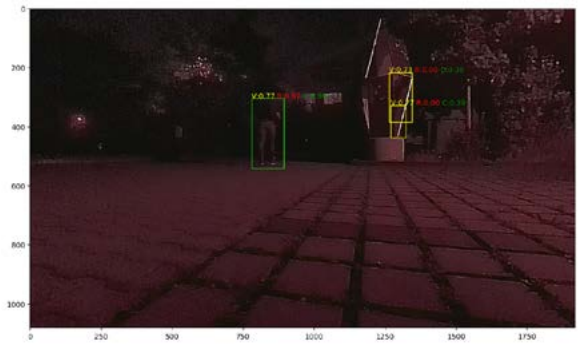
(a): V=0.69, R=1.00, C=1.00



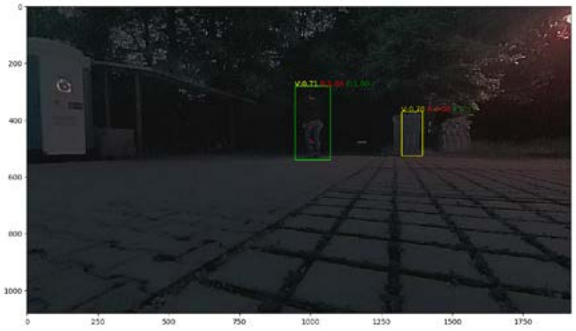
(b): V=0.68, R=0.99, C=1.00



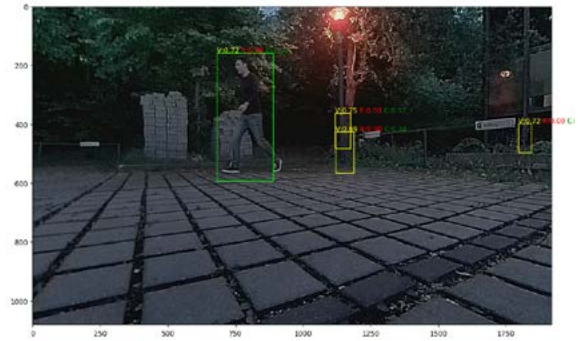
(c): V=0.77, R=0.88, C=0.97



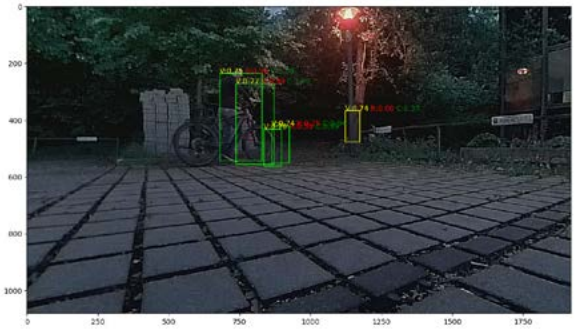
(d): V=0.77, R=0.97, C=0.99



(e): V=0.71, R=1.00, C=1.00



(f): V=0.72, R=0.98, C=1.00



(g): 0.74<=V<=0.78, 0.5<=R<=0.99, C>0.89

(h): 0.72<=V<=0.74, 0.50<=R<=0.97, C>0.84

Figure 9: Nighttime images showing vision detector bounding boxes, individual and fused confidence scores, all ambiguous results have been resolved by the fusion process.

All computation for our system was done on the GW5x or GW6x processors, which are low power, small area SoCs, with no external DDR. As mentioned above, the vision detector uses ~1MB memory and takes ~40-60ms to process a frame, and the radar pipeline uses ~3-4MB memory, taking ~30-40ms (on GW6x). This gives a total memory requirement of ~4-5MB, a very modest amount in an age where gigabytes are often used, and a fused frame rate of ~10fps (~1/100ms), which is sufficient for low-speed scenarios (such as backup or parking). With a reasonable increase in compute capability, higher fps can be achieved for faster speed applications such as side e-mirrors.

Cooperative Radar

Radar and vision are key sensing modalities for ADAS and automated driving [13]. While vision provides great lateral resolution, radar inherently provides depth and radial velocity information. Lateral resolution though is the key weakness of a radar sensing modality. Further, due to the long wavelength, radar signals are reflected in a different manner. In typical drive scenarios most of the optical reflections are diffuse, while radar signals contain significantly more specular reflections. This leads to a much sparser point cloud output of the radar. For sensor fusion, the orthogonal nature of the two modalities is a great benefit. However, it also brings up the problem of aligning and matching the input of the different sources. This task becomes significantly easier when the number of detections and the lateral resolution of the radar increases and gets

closer to the optical detections. In theory, a very large radar or using a very large number of radar sensors can achieve this. This approach though is impractical due to excessive hardware requirements and the associated cost, power consumption and mounting space requirements. The work at hand thus focuses on extracting more information from existing radar sensors with a “cooperative radar” approach [14, 15].

Cooperative radar has two or more radar sensors operating in a loosely synchronized manner following the method in [15, 16]. In a separate processing step the received radar signals are then aligned in such a way that the paths in between the sensors can be used. Figure 10 illustrates the principle for two sensors. The resulting additional path between the two radars is equivalent to using a third radar in the middle of the two radars with two times the resolution.

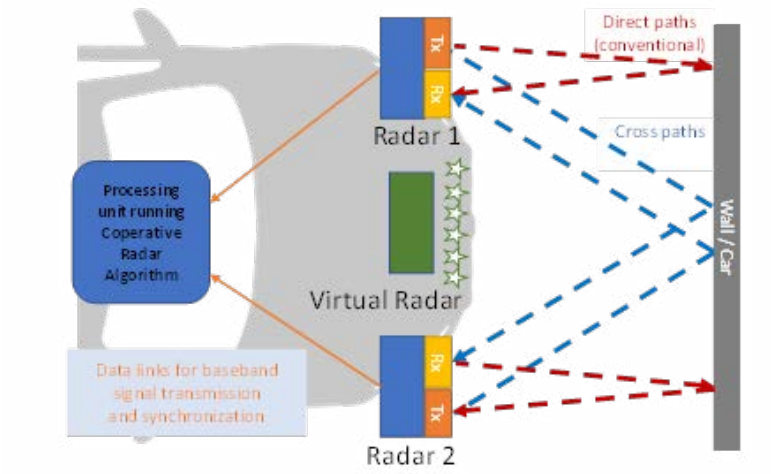


Figure 10: Secondary radar principle of operation and resulting virtual radar.

Practical drive tests have been done with two radars, each having 6 transmit and 8 receive channels. The radar units were mounted in the front of a test vehicle. The resulting multi-perspective radars provided significantly higher point cloud density for extended objects and provided information missed by the single radars. Sample scenes depicted in Figure 11, highlight the achieved benefits.

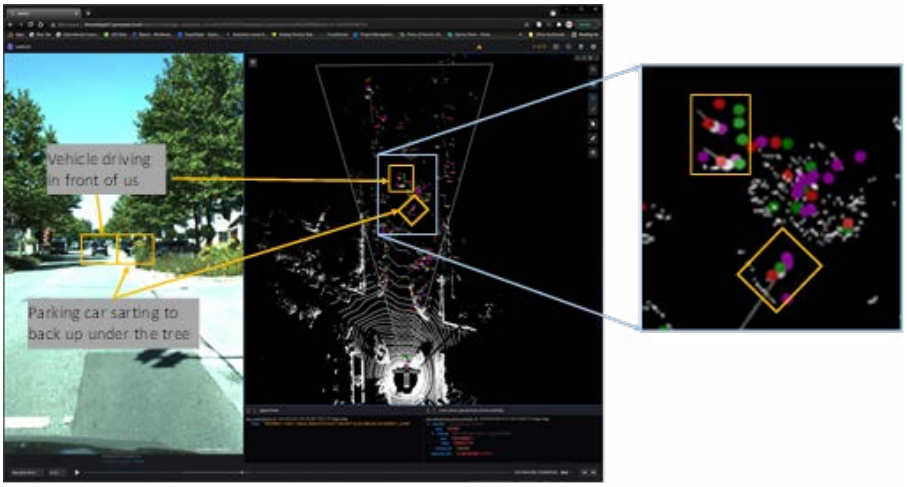


Figure 11: Enhanced point cloud in test drives: Ground truth data and camera data shown, radar reflections from left radar in red, right radar in green and virtual radar in magenta.

In addition to the resolution benefit the loose coupling also allows extraction of the tangential velocity. Previously, only the radial velocity component was estimated. With the larger base between the two radars, the tangential velocity can be estimated as well. This results in faster conversion of tracking filters, reducing the latency for new objects in the field of view. Figure 12 shows a sample scene with a pedestrian crossing.

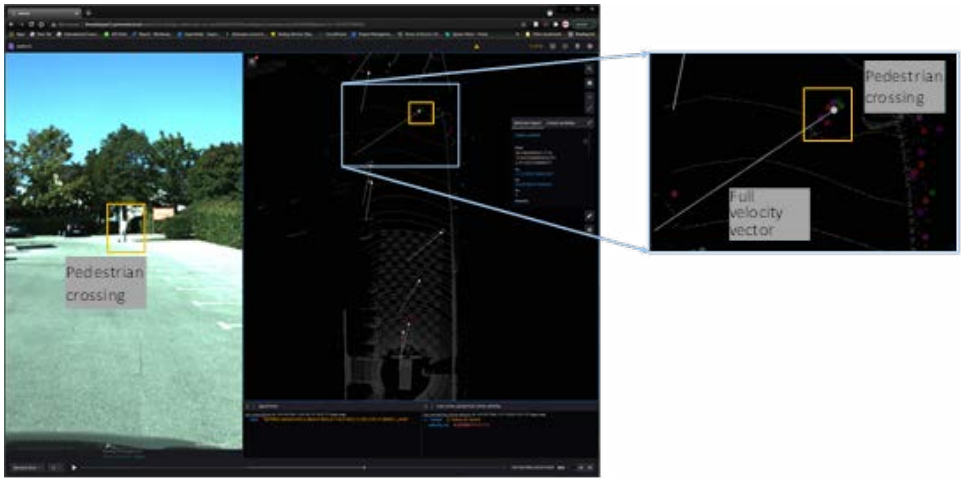


Figure 12: Pedestrian crossing while driving, full velocity vector displayed with white line.

Conclusions and Improvements

We have presented a small footprint vision-radar fusion system for detection. Our system was specifically developed with slow speed, smart yet simple, automotive applications in mind. We focused on smart backup, and showed how pedestrians and bicycles, under challenging low light conditions, can be successfully detected with our approach and hardware implementation. As all processing was done inside the camera video processor, such a system can be placed at any location on the vehicle that can accommodate a camera – this makes our system well-suited for edge or zonal configurations. Further improvements can readily be made by modestly increasing the hardware resources (processing capability and memory) and expanding the radar/fusion pipeline. The radar sub-system can be improved by converting it to a detector, similar to the vision detector. This means extracting features from the radar data and training them to build a classifier. This will give us an independent radar detector and subsequent mass function, implying missed detections by the vision detector, could possibly be independently detected by the radar. Instead of decision level fusion, more sophisticated feature level fusion of vision and radar data could be considered. Temporal tracking can also be added to further enhance performance. All these improvements are well understood and well used in the industry, when deployed in large complicated automotive AI systems. We have deliberately stayed away from neural networks for processing/memory requirements, but using CNNs and other modern machine learning architectures allows one to handle much more complicated fusion applications. We also hope to study vision fusion with our cooperative radar, while maintaining a small footprint. With the denser points, vector velocities, and adding a radar classifier, this should give further increases in detection accuracy and robustness.

References

1. Z. Wei, F. Zhang, S. Chang, Y. Liu, H. Wu and Z. Feng , “MmWave Radar and Vision Fusion for Object Detection in Autonomous Driving: A Review”, Sensors 2022 22(7), 2542.
2. M. Bouain, D. Berdjag, N. Fakhfakh and R. B. Atitallah, “Multi-Sensor Fusion for Obstacle Detection and Recognition: A Belief-based Approach”, 2018 21st International Conference on Information Fusion (FUSION).
3. indie Semiconductor, www.indiesemi.com.
4. P. Dollar Z. Tu, P. Perona and S. Belongie, “Integral Channel Features”, BMVC 2009.
5. P. Dollar, S. Belongie and P. Perona, “The Fastest Pedestrian Detector in the West”, BMVC 2010.
6. P. Dollár, R. Appel, S. Belongie and P. Perona, “Fast Feature Pyramids for Object Detection,” in IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 36, no. 8, pp. 1532-1545, Aug. 2014.
7. R. Appel, T. Fuchs, P. Dollár and P. Perona, “Quickly Boosting Decision Trees – Pruning Underachieving Features Early”, Proceedings Of the 30th International Conference on Machine Learning, Atlanta, Georgia, USA, 2013.
8. Y. Zhou, L. Liu, H. Zhao, M. López-Benítez, L. Yu and Y. Yue, “Towards Deep Radar Perception for Autonomous Driving: Datasets, Methods, and Challenges”, Sensors 2022, 22(11), 4208.

References (continued)

9. M. Ankerst, M. Breunig, H. Kriegel and J. Sander, “OPTICS: Ordering Points To Identify the Clustering Structure“, Proceedings of the 1999 ACM SIGMOD International Conference on Management of Data. ACM Press.
10. H. Junlong and D. Feng, “Unified Calibration Method for Millimeter-Wave Radar and Machine Vision“, IJERT Vol. 7 Issue 10, October 2018.
11. A. P. Dempster, “A generalization of Bayesian inference”, Journal of the Royal Statistical Society, Series B 30 205-247, 1968.
12. G. Shafer, “A Mathematical Theory of Evidence”, Princeton University Press 1976.
13. S. Patole, M. Torlak, D. Wang and M. Ali, “Automotive radars: A review of signal processing techniques”, IEEE Signal Process. Mag., vol. 34, no. 2, pp. 22-35, Mar. 2017.V. S.
14. Chernyak, Fundamentals of Multisite Radar Systems: Multistatic Radars and Multiradar Systems, New York, NY, USA:Gordon & Breach, 1998.
15. M. Gottinger, M. Hoffmann, M. Christmann, M. Schütz, F. Kirsch, P. Gulden and M. Vossiek, “Coherent Automotive Radar Networks: The Next Generation of Radar-Based Imaging and Mapping”, IEEE Journal of Microwaves, vol. 1, no. 1, pp. 149 - 163, January, 2021
16. M. Gottinger, P. Gulden and M. Vossiek, “Coherent Signal Processing for Loosely Coupled Bistatic Radar”, IEEE Transactions on Aerospace and Electronic Systems, vol. 3, no. 57, pp. 1855-1871, June, 2021