https://doi.org/10.2352/ISSN.2470-1173.2018.17.AVM-256

Multi-sensor fusion for Automated Driving: Selecting model and optimizing on Embedded platform

Shyam Jagannathan, Mihir Mody, Jason Jones*, Pramod Swami and Deepak Poddar,
Automotive Processor Business – Texas Instruments Inc (India,* USA),
Email : { shyam.jagannathan, mihir, jasonatjones, pramods, deepak.poddar}@ti.com

Abstract

Automated Driving requires fusing information from multitude of sensors such as cameras, radars, lidars mounted around car to handle various driving scenarios e.g. highway, parking, urban driving and traffic jam. Fusion also enables better functional safety by handling challenging scenarios such as weather conditions, time of day, occlusion etc. The paper gives an overview of the popular fusion techniques namely Kalman filters and its variation e.g. Extended Kalman filters and Unscented Kalman filters. The paper proposes choice of fusing techniques for given sensor configuration and its model parameters. The second part of paper focuses on efficient solution for series production using embedded platform using Texas Instrument's TDAx Automotive SoC. The performance is benchmarked separately for "predict" and "update" phases on for different sensor modalities. For typical L3/L4 automated driving consisting of multiple cameras, radars and lidars, fusion can supported in real time by single DSP using proposed techniques enabling cost optimized solution.

Introduction

In order to let a car run autonomously, first it has to sense the external environment/surroundings; process the data and act by making meaningful decisions. In this sense, process and act chain, the sensing part of the external environment is taken care by sensors like camera, radar, LIDAR and referred as surround sensors in rest of the paper. Apart from surround sensors, other sensors like vehicle odometry sensors and actuators are also important to feed the information to decision-making block. For example, the steering wheel angle and wheel speed is important data for a car to make the right decision along with surrounding information. So broadly we would divide sensors in be below three categories,

- Surround sensors: These are mounted on the external/internal surface of the car and useful to provide surrounding information. Example: Camera, radar, Lidar, ultrasonic, infrared camera, IMU, GPS and digital map etc.
- Vehicle odometry sensors: These sensors capture the information about vehicle motion. Example: wheel speed, acceleration, yaw rate, steering wheel angle etc.
- Actuators: These are the sensors which translate the human/machine actions. Example: Break Torque, Engine Torque, restraint actuators, wheel spring etc.

Car makers have been using different sensors mainly lidar, radar, camera and ultrasonic for safety features like ACC (Automatic Cruise Control), LKA (Lane Keep Assist), blind spot detections, forward collision warning, and very recently for active safety features like AEB (Auto-Emergency Brake) as well. In recent past, industry has seen the usage of more sensor/information like satellite information, vehicle and infrastructure (V2V and V2x) and Lidar to improve the robustness of these safety features. There is significant overlap of the information provided by these sensors. At the same time, their degree of reliability varies. For example, radar and camera both can identify the distance of an object but the reliability of information from a radar sensor is higher as compared to a camera. Autonomous driving systems need to provide the highest degree of reliability and would require a good overlap of information from different sensors to make a confident decision using fusion process as shown in Figure 1.

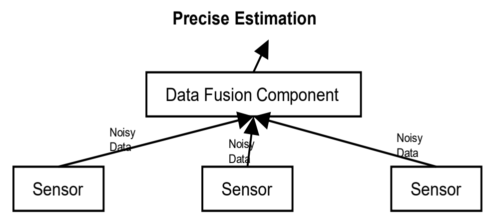

***Figure 1**. Basics of fusion process*

The first urban autonomous vehicle demonstration in DARPA 2007 had 18 sensors (9 Lidar, 5 Radar, 2 vision and 2 GPS/IMU) having redundant information for vehicle path planning. With redundant information measurement precision can be enhanced and in addition, the fault tolerance of the overall system increases, as the failure of one sensor does not necessarily result in the failure of the system as a whole [1][6]. Figure 2, provides a pictorial view of how multiple sensors can help to cover different fields of view and basic functions for autonomous driving [2].

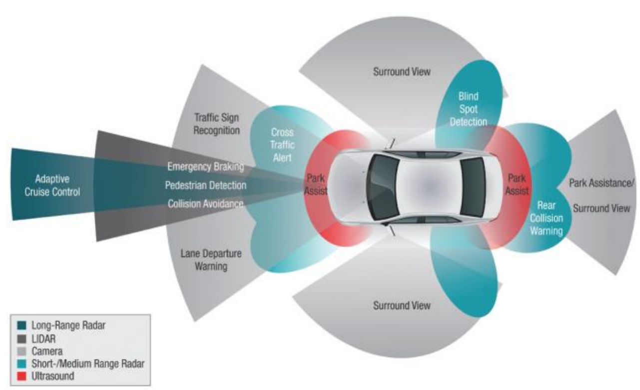

***Figure 2**. Surround sensors coverage area and applications*

Texas Instruments (TI) TDA3 platform is designed to cater to multiple computer vision markets [4][5]. The TDA3x SoC is a heterogeneous and scalable architecture that includes dual core of ARM® Cortex®-M4, dual core of C66x DSPs and single core of Embedded Vision Engine (EVE) for vector processing as shown in Figure 3. It integrates hardware for camera capture, Image Signal Processor (ISP) and Display Sub System (DSS) resulting in better image quality at lower power. It also contains large on-chip RAM, rich set of input/output (I/O) peripherals for connectivity, and safety mechanism for automotive market resulting in lower system cost.

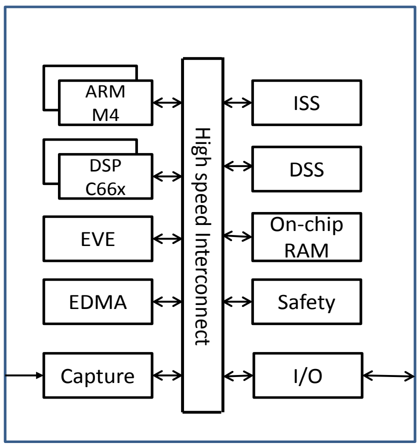

***Figure 3**. Block Diagram of TDA3x SoC*

Fusion System

The fusion system are classified in 3 categories as shown in Figure 4 namely high level fusion, mid-level fusion and low level fusion.

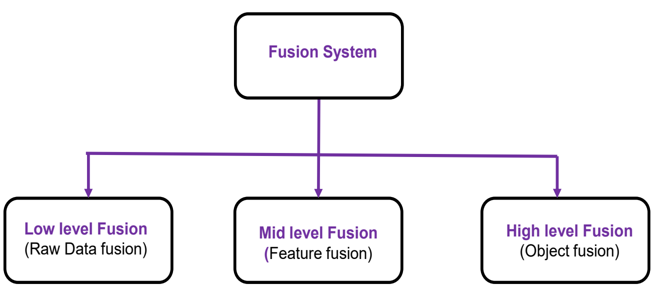

***Figure 4**. Classification of fusion systems*

The High level fusion system typically fuses object level data (e.g. type, distance, velocity etc.) across modalities (e.g. radar, camera and radar) to give more realizable picture above object level data. The low level fusion happens across raw data across sensors e.g. merging point cloud across radar & lidar, finding depth using stereo cameras, viewing 360^0 around cars using surround view cameras etc. The mid- level fusion is merging features across sensors (e.g. features from camera and radar is used to gather to detect object). The paper is going to focus mainly on high level fusion systems.

High level fusion techniques

In high level sensor fusion data from each sensor is independently processed to detect objects at the node, followed by object tracking using Kalman filters. Several techniques are available to detect objects using camera, lidar and radar data. It is assumed that object detection is already applied on each sensor data and best estimates on object position and velocity is readily available for sensor fusion. Kalman filters are a popular choice for performing high level sensor fusion because of its robustness to noise [10]. A state is defined which models object motion in terms of 2D or 3D position and 2D velocity. The state of each object being tracked is predicted and updated when data from different sensors arrive at different time instant. Basic flow of high level fusion is shown in Figure 5.

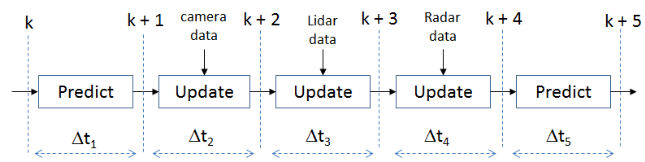

***Figure 5**. High level object data from camera, lidar and radar is fused using Kalman filter predict and update stages at different time instant k.*

Kalman Filters

Kalman filters belong to the family of Bayesian filters which comprise of two steps, predict and update [3]. The state vector models the object parameters such as 2D position information $\mathbf{p_x}$, $\mathbf{p_y}$ and 2D velocity information $\mathbf{v_x}$, $\mathbf{v_y}$ depending on the choice of object model such as constant velocity, acceleration, turn rate etc. Given a prior object state $\mathbf{x}$ at time k we can predict the state $\mathbf{x'}$ at time k+1 using state transition function $\mathbf{F}$ as shown in Eq. (1). While the state represents the mean position estimate, object covariance $\mathbf{P}$ Eq. (2) represents the uncertainty. Kalman filters also model noise such as state transition noise $\mathbf{u}$, process noise covariance $\mathbf{Q}$, measurement noise covariance $\mathbf{R}$ all of which affects the state and process covariance. When a measurement arrives from a sensor, the measurement function $\mathbf{H}$ maps the state to the actual measurement value of the sensor (3). By computing the difference in actual measured value $\mathbf{z}$ and predicted state, the Kalman gain $\mathbf{K}$ Eq. (5) is computed which is used to refine the state Eq. (6) and process covariance Eq. (7) once again. This cycle repeats till the Kalman filter is able to track the object precisely with less uncertainty [7].

Kalman Filter Predict Equations

$$x' = Fx + u \tag{1}$$

$$P' = FPF' + Q \tag{2}$$

Kalman Filter Update Equations

$$y = z - Hx' \tag{3}$$

$$S = HP'H^T + R \tag{4}$$

$$K = P'H^T S^{-1} \tag{5}$$

$$x = x' + Ky \tag{6}$$

$$P = (I - KH)P' \tag{7}$$

While the prediction step remains largely the same, the update step varies depending on the type of sensor, camera, lidar or radar. Camera and lidar provides data in Cartesian coordinate system so the measurement function **H** a linear function as shown in Figure 6. But radar provides data in polar coordinate system which makes **H** a non-linear function.

$$\begin{pmatrix} p_x \\ p_y \end{pmatrix} \xleftarrow{h(x')} \begin{pmatrix} p_x' \\ p_y' \\ v_x' \\ v_y' \end{pmatrix} \quad \text{where, } h(x') = \begin{pmatrix} 1\,0\,0\,0 \\ 0\,1\,0\,0 \end{pmatrix}$$

***Figure 6**. Measurement function h(x) for lidar*

Extended Kalman Filter

The standard Kalman filter equations are derived assuming a Gaussian distribution of data affected by white noise. The measurement function **H** for radar data as shown in Figure 7 converts the state in Cartesian coordinates $\mathbf{p_x}$, $\mathbf{p_y}$, $\mathbf{v_x}$, $\mathbf{v_y}$ to polar coordinates **ρ, φ, ρ'** where **ρ** is range, **φ** is bearing and **ρ'** is radial velocity.

$$\begin{pmatrix} \rho \\ \varphi \\ \dot{\rho} \end{pmatrix} \xleftarrow{h(x')} \begin{pmatrix} p_x' \\ p_y' \\ v_x' \\ v_y' \end{pmatrix} \quad \text{where, } h(x') = \begin{pmatrix} \sqrt{p_x'^2 + p_y'^2} \\ \arctan(p_y' / p_x') \\ \dfrac{p_x' v_x' + p_y' v_y'}{\sqrt{p_x'^2 + p_y'^2}} \end{pmatrix}$$

***Figure 7**. Measurement function h(x) for converting state in Cartesian format to measurement in polar format for radar data*

When the measurement function **H** is non-linear, the resulting distribution is not Gaussian, so the Kalman Filter equations do not hold. The EKF approach tries to address this problem by doing a linear approximation of the non-linear function [8]. The approximation is derived by taking first-order approximation of Taylor series expansion as shown in Figure 8 where $\mathbf{H_j}$ is (3x4) Jacobian matrix. The measurement function **H** is replaced by the Jacobian $\mathbf{H_j}$ in the update equation. Remaining equations are the same as standard Kalman filter.

$$Hj = \begin{bmatrix} \dfrac{\delta\rho}{\delta p_x} & \dfrac{\delta\rho}{\delta p_y} & \dfrac{\delta\rho}{\delta v_x} & \dfrac{\delta\rho}{\delta v_y} \\[2ex] \dfrac{\delta\varphi}{\delta p_x} & \dfrac{\delta\varphi}{\delta p_y} & \dfrac{\delta\varphi}{\delta v_x} & \dfrac{\delta\varphi}{\delta v_y} \\[2ex] \dfrac{\delta\rho'}{\delta p_x} & \dfrac{\delta\rho'}{\delta p_y} & \dfrac{\delta\rho'}{\delta v_x} & \dfrac{\delta\rho'}{\delta v_y} \end{bmatrix}$$

***Figure 8**. Jacobian of measurement function **H** for radar data and constant velocity model*

Extended Kalman filters are mostly used in automatic cruise control applications with safe distance keeping where linearizing a non-linear function is acceptable. Common models used in such applications are constant velocity or constant acceleration models which help define the state of the object being tracked.

For urban scenarios where vehicles turn at intersections or cases where a car driving on a highway turns to take an exit, the constant velocity model will overshoot the actual measurements. This is because while turning the vehicle will slow down and the velocity will reduce. The Constant-Turn-Rate-Velocity or CTRV model is a better fit to such scenarios.

Unscented Kalman Filter

Unscented Kalman Filter uses a technique known as sigma points to model non-linearity [9]. As shown in Figure 9, Sigma points are a set of N points chosen around mean and within the covariance ellipse of the object state. These points are passed through the state transition function **F** and from the resulting response a new mean and covariance is computed, this completes the prediction step. The same sigma points are reused and passed through the measurement function H with results in a new mean and covariance, which completes the update step.

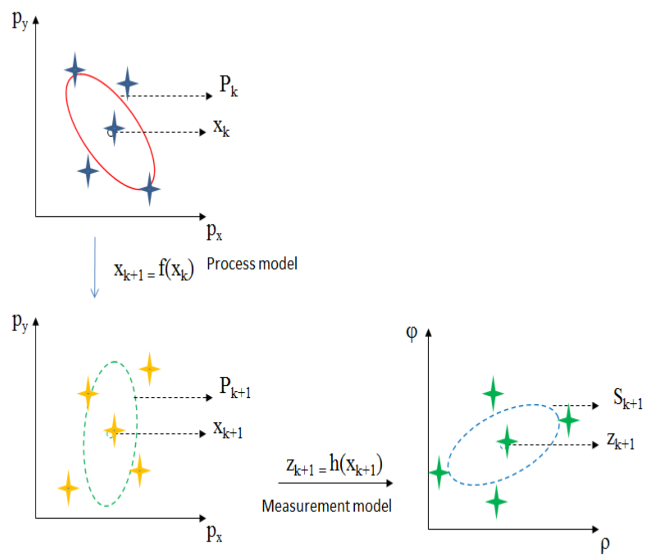

***Figure 9**. Using sigma points, predict the state at k+1 by passing through the non-linear state transition function f(x) followed by measurement function h(x) to successfully convert from $\mathbf{p_x}$, $\mathbf{p_y}$ to ρ, φ*

Optimizing on DSP

The TDA3x SoC contains dual core of C66x DSP [4][5] clocked up-to 750 MHz for each core. The C66x DSP shown in Figure 10 is a floating point VLIW architecture with 8 functional units (two multipliers and six arithmetic units) that operate in parallel as shown in Figure 9. It comprises of 64 general purpose 32-bit registers shared by all eight functional units. There are four arithmetic units .L1/.L2, .S1/.S2, two multiplier units for .M1/.M2 and two data load-store units .D1/.D2. Each C66x DSP core has 32KB of L1 data cache, 32KB of L1 instruction cache and 288KB of unified L2 data/instruction memory.

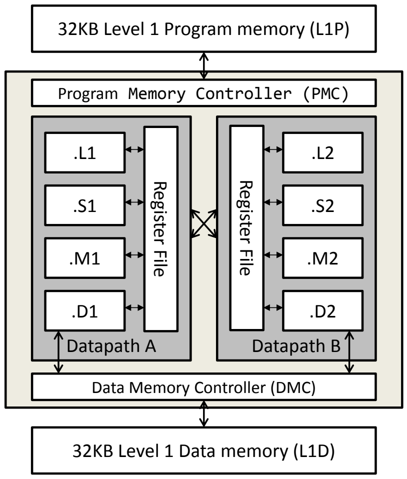

***Figure 10**. TI's C66x DSP Architecture showing functional units, data path and memory*

With dual 64-bit data path TI's C66x DSP can work on 4 single precision values in parallel. The .M1/.M2 multiplier units can multiply 8 single precision values which accelerate the matrix multiply operations of Kalman filter. Special instructions such as reciprocal **1/x** (RCPSP) and reciprocal of square root $1/\sqrt{x}$ (RSQRSP) speed up the division and finding square root operations. Optionally precision can be refined using Newton-Raphson techniques. As an example configuration, tracking of single-object with radar, lidar and camera data is as shown in Table 1. The common prediction step and radar update step is done using UKF approach whereas lidar and camera update steps can be done using standard KF or EKF method. The state x is defined to be a 4x1 vector which 2D position $\mathbf{p_x}$, $\mathbf{p_y}$ and 2D velocity $\mathbf{v_x}$, $\mathbf{v_y}$.

Table 1: Example configuration of radar, lidar and camera

Stages	Sensor	Method
Predict	NA	UKF
Update	Radar	UKF
Update	Lidar	KF/EKF
Update	Camera	KF/EKF

The prediction stage of UKF involves steps as shown in Table 2. Sigma point generation involves compute intensive steps of finding square root of covariance matrix which can be done using Cholesky decomposition. The generated sigma points is augmented from N to 2N + 1 points before passing through the state transition function in the prediction of sigma points stage. Passing points through non-linear function involves table-lookup operations, division and few arithmetic operations. Using the new sigma points a new state **x** and covariance **P** is computed which involves single precision multiplies and arithmetic operations.

Table 2: Example configuration of radar, lidar and camera

Steps	Description
Step 1	Generate sigma points
Step 2	Predict sigma points
Step 3	Predict mean and covariance

When a radar sample arrives, the UKF update stage is triggered which comprises of steps as shown in Table 3. The first stage involves incorporating the noisy measurement to the predicted state by passing through the measurement function **H**. It involves finding square-root, division, table-lookup and arithmetic operations. This is followed by update step where cross correlation function **Tc** is computed which is required for finding Kalman gain K, this involves mostly arithmetic operations and matrix multiply operations. Using Kalman gain, the state and covariance matrix is updated to new values. Steps are similar to prediction stage which involves arithmetic operations and matrix multiply operations.

Table 3: Update step for radar, UKF method

Steps	Description
Step 1	Predict state using measurement
Step 2	Compute measurement covariance – S
Step 3	Compute cross correlation matrix – Tc
Step 4	Compute Kalman gain – K
Step 5	Compute mean and process covariance (x, P)

When a lidar or camera measurement arrive the update step using standard KF or EKF is triggered. Operations mostly involve matrix multiply operation and arithmetic operations and 2x2 matrix inverse operation to find Kalman gain as shown in Table 4.

Table 4: Update step for lidar and camera, standard KF method

Steps	Description
Step 1	Predict state using measurement
Step 2	Compute measurement covariance – S
Step 3	Compute Kalman gain – K
Step 4	Compute mean and process covariance (x, P)

Tracking multiple objects involves maintaining separate state for each object. This also means separate predict and update stages. This poses a challenge of associating measurements for one object with another. Common practice is to apply Normalized Cross Correlation (NCC) on object features. Object feature could be anything which uniquely identifies and object. NCC mostly involves fixed point multiplication and computing the reciprocal of square-root for normalization.

Results

Core kernel benchmarks for standard KF, EKF and UKF for predict and update stages is provided in Table 5. C66x DSP cycles taken for the example configuration are shown in the Table 6. Assuming radar samples arrive at 60 fps, camera or lidar samples at 30fps and the maximum number of objects as 64 the total cycles taken on 750Mhz C66x DSP on TDA3x is as shown in Table 7. It shows that with a 35% load of a single DSP, the fusion processing can be completed.

Table 5: Core kernel DSP cycles for KF, EKF and UKF

CAMERA/LIDAR			
	KF	**EKF**	**UKF**
Predict	3,620	3,620	NA
Update	7,600	3,000	NA
RADAR			
	KF	**EKF**	**UKF**

Predict	NA	3,620	20,490
Update	NA	12,300	10,650

Table 6: Example configuration DSP cycles for tracking 1 object

Stages	Sensor	Method	DSP Cycles
Predict	NA	UKF	20,490
Update	Radar	UKF	10,650
Update	Lidar	KF/EKF	3,000
Update	Camera	KF/EKF	3,000
		Total	**37,140**

Table 7: Performance on 750Mhz C66x DSP

Total Kalman Filter (KF, EKF, UKF) cycles	131,097,600
Object association cycles	88,875,600
Control code overhead – 20%	43,991,040
Total DSP cycles for 30fps	**263, 946, 240**

Conclusion

The paper gives overview of sensor fusion in context of automated driving and explains fusion techniques namely Kalman filters and its variation e.g. Extended Kalman filters and Unscented Kalman filters. The paper proposes choice of fusing techniques for given sensor configuration and its model parameters. The paper also devolves on optimized solution on DSP for series production using embedded platform using TI's TDA series of ADAS processors. The proposed solution enables fusion of multiple cameras, radars and lidar in a single DSP for cost-effective solution.

References

[1] Michael Aeberhard, Nico Kaempchen, "High-level sensor data fusion architecture for vehicle surround environment perception," Proc. 8th Int. Workshop Intell. Transp., 2011.

[2] Mihir Mody, Jason Jones, Kedar Chitnis, Rajat Sagar, et.al, "Understanding Vehicle E/E Architecture Topologies for Automated Driving : System Partitioning and Tradeoff Parameters", AVM track, Electronic Imaging, 2018.

[3] Sebastian Thrun, Wolfram Burgard and Dieter Fox , "Probabilistic Robotics", ACM book, August 2005.

[4] Mihir Mody, Pramod Swami, Kedar Chitnis, Shyam Jagannathan, et.al, "High Performance Front Camera ADAS Applications on TI's TDA3X Platform", IEEE 22nd International Conference on High Performance Computing, HiPC, 2015

[5] Prashanth Viswanath, Kedar Chitnis, Pramod Swami, Mihir Mody, et. al, "A Diverse Low Cost High Performance Platform for Advanced Driver Assistance System (ADAS) Applications ",Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, CVPR, 2016

[6] Kedar Chitnis, Mihir Mody, M G Biju, Sivaraj R, et. al, "Enabling Functional Safety ASIL Compliance For Autonomous Driving Software Systems", AVM track, Electronic Imaging, 2017

[7] R. E. Kalman, "A new approach to linear filtering and prediction problems", Journal of Fluids Engineering, vol. 82, no. 1, pp. 35-45, 1960

[8] K. Fujii, "Extended kalman filter", Refernce Manual, 2013

[9] H. Zhang et al., "Unscented kalman filter and its nonlinear application for tracking a moving target", Optik-International Journal for Light and Electron Optics, vol. 124, no. 20, pp. 4468-4471, 2013

[10] Qiang Li, Ranyang Li, Kaifan Ji, Wei Dai "Kalman Filter and Its Application", IEEE 8th International Conference on Intelligent Networks and Intelligent Systems, 2015

Author Biography

Shyam Jagannathan is Senior Technical Lead for Automotive Business at Texas Instruments (TI) Incorporated. His domains of interest are perception, sensor fusion, localization, path planning and next generation DSP's. He received his Bachelor of Engineering (BE) from R. V. College of Engineering, India in 2005 and his master's degree in signal processing and communications from IIT Chicago, Illinois in 2013.

Mihir Mody is Senior Principal Architect for Automotive business at Texas Instruments (TI) Incorporated. His domains of interest are image processing, computer vision, video coding & machine/deep learning algorithms with a focus on embedded HW & SW solution. He received Master of Engineering (ME) degree from Indian Institute of Science (IISc) in 2000 and Bachelor of Engineering (BE) from GCOEP in 1998.

Jason Jones is the Principal Systems Architect for Automotive Business at Texas Instruments (TI) Incorporated, responsible for current and future roadmap solutions encompassing Digital Cockpit, ADAS, and Vehicle Gateway applications. He holds Bachelor of Science (BS) degree from Texas A&M University in 1993.

Pramod Swami is Principal Engineer with Automotive group at Texas Instruments (TI). His domains of interest are hardware and software architecture, design and implementation of signal processing for video compression and computer vision. He received Bachelor of Engineering degree from MNIT, Jaipur.

Deepak Poddar is Technical Lead with Automotive group at Texas Instruments (TI). His domain of expertise is embedded software development and optimization for automotive applications. He received Bachelor of Engineering from NIT, Warangal in 2004.