

# MATLAB EXPO 2016

## Designing Perception Systems for Autonomous Driving

Avinash Nehemiah  
November 3, 2016



# How Autonomous Driving Impact our Lives

1. Safety
  - Reduce traffic accidents 90%
  - Save 30,000 lives/year ( on-par with modern vaccines)
2. Redefine logistics – substantial fuel savings
  - Reduce cost
  - Environment friendly
3. Save us from our daily commutes

**Engineers in autonomous driving choose MATLAB and Simulink.**

# MATLAB and Simulink in Perception Systems for Autonomous Driving

**Sensor Design**

**Signal Processing**

**Sensor Fusion**

**Decision Logic**



## Delphi

**Radar Sensor Alignment Algorithm for Automotive Active Safety System**

## Continental

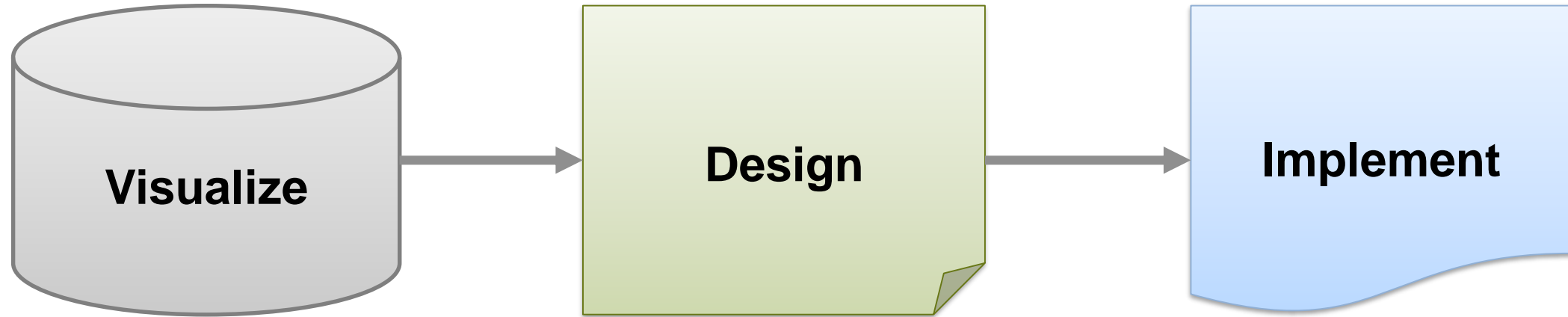
**Traffic Sign Recognition for Driver Assistance Systems**

## Scania

**Model-Based Approach to Resource-Efficient Object Fusion for an Autonomous Braking System**

## Magna

# MATLAB Helps Engineers...



# Test Vehicle Equipped with Various Sensors



## Velodyne LiDAR HDL-32E

- Horizontal FoV: 360°
- Vertical FoV: +10..-30°
- Range: 80..100m
- 100 Mbps Ethernet



## Point Grey Blackfly

- Stand “ice cube” vision camera
- 800x600, 27FPS
- GigE interface



## Mobileye 560

### Mobileye 560

- FoV: 38°x150m
- CAN interface



## XSENS MTI-G-700

- Stable and sensitive
- MEMS-based AHRS
- USB interface



## Delphi ESR

### Delphi ESR

- 76GHz electronically scanning radar
- Dual FoVs, 90°x60m, 20°x174m
- CAN interface

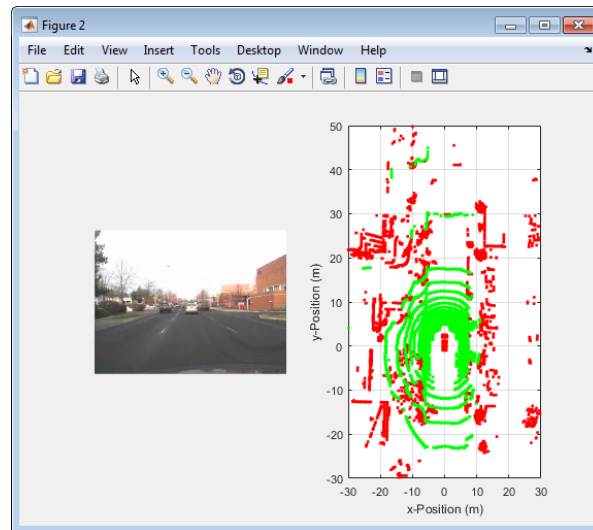
# Two Examples for Today

Sensor  
Design

Signal  
Processing

Sensor  
Fusion

Decision  
Logic



LiDAR Processing



Sensor Fusion  
(Camera + Radar)

# Visualize Data and Gain Insight



**Vision + Radar Sensor Fusion for**

Pick Data: 01\_highway\_lanechange\_25s\_sensor.mat

load Li... mode: MATL... Algorithm Parameters Sensor Configuration

time(sec) frame rep...  
 total: 25.00 500 Play Pause [checked] pau... fcw Reset

curr: 2.90 59

hostSpee: 98.68 kp  
 hostYawRa: 0.26 deg/

Algorithm Process:

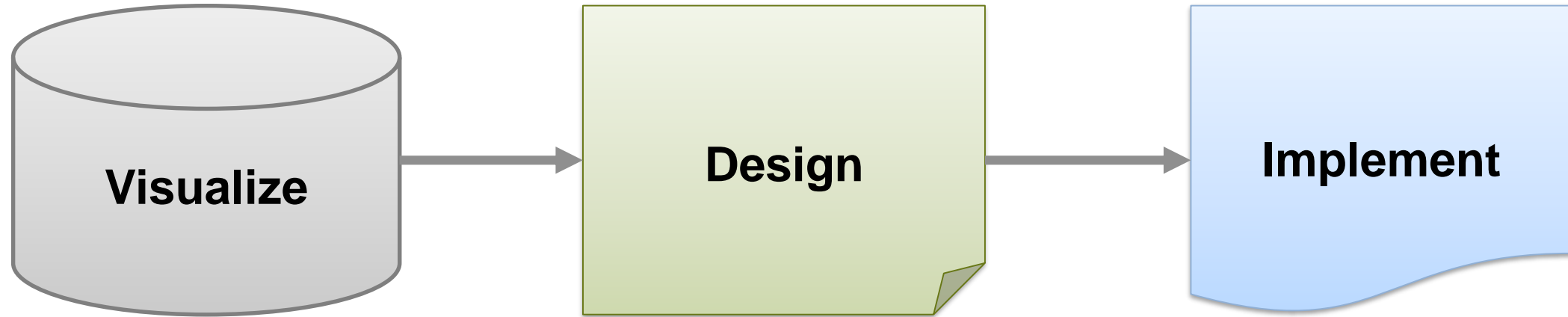
- estimatePath
- findRelevantRadarOb...
- fuseVisionAndRadar...
- trackFusedObjects
- assessThreat
- Select Above All

Plot: X (m) vs Y (m). laneWidth(m): 3.54

Variable Browser:

Variable	Value
1x1 Ax...	1x1 Ax...
1x1 poin...	1x1 poin...
29019x1	29019x1
1x1 poin...	1x1 poin...
29019x1	29019x1
1x1 poin...	1x1 poin...
1x1 plar...	1x1 plar...
25981x3	25981x3
25981x1	25981x1
2599x1	2599x1
1x1 poin...	1x1 poin...
21262x1	21262x1
2599x1	2599x1
600x800	600x800

## MATLAB Helped Us...



- Visualize sensors
  - Cameras
  - LiDAR
  - Radar
- Create custom apps and visualizations



# Design Algorithms with MATLAB



fcwAppWebinar

### Vision + Radar Sensor Fusion for

Pick Data: 02\_city\_c2s\_fcw\_10s\_sensor.mat

load Li... mode: MATL...


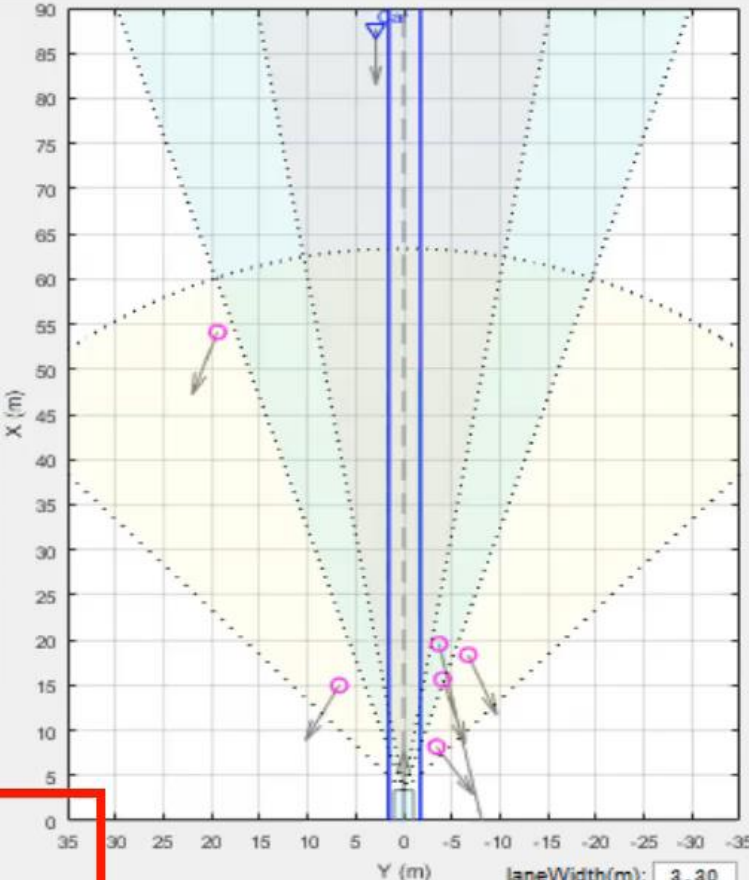
time(sec) frame  rep...  
 total: 10.20 204    pau... fcw

curr: 0.15 4

hostSpee: 56.40 kp  
 hostYawRa: 0.38 deg/

**Algorithm Process:**

- estimatePath
- findRelevantRadarOb...
- fuseVisionAndRadar...
- trackFusedObjects
- assessThreat
- Select Above All

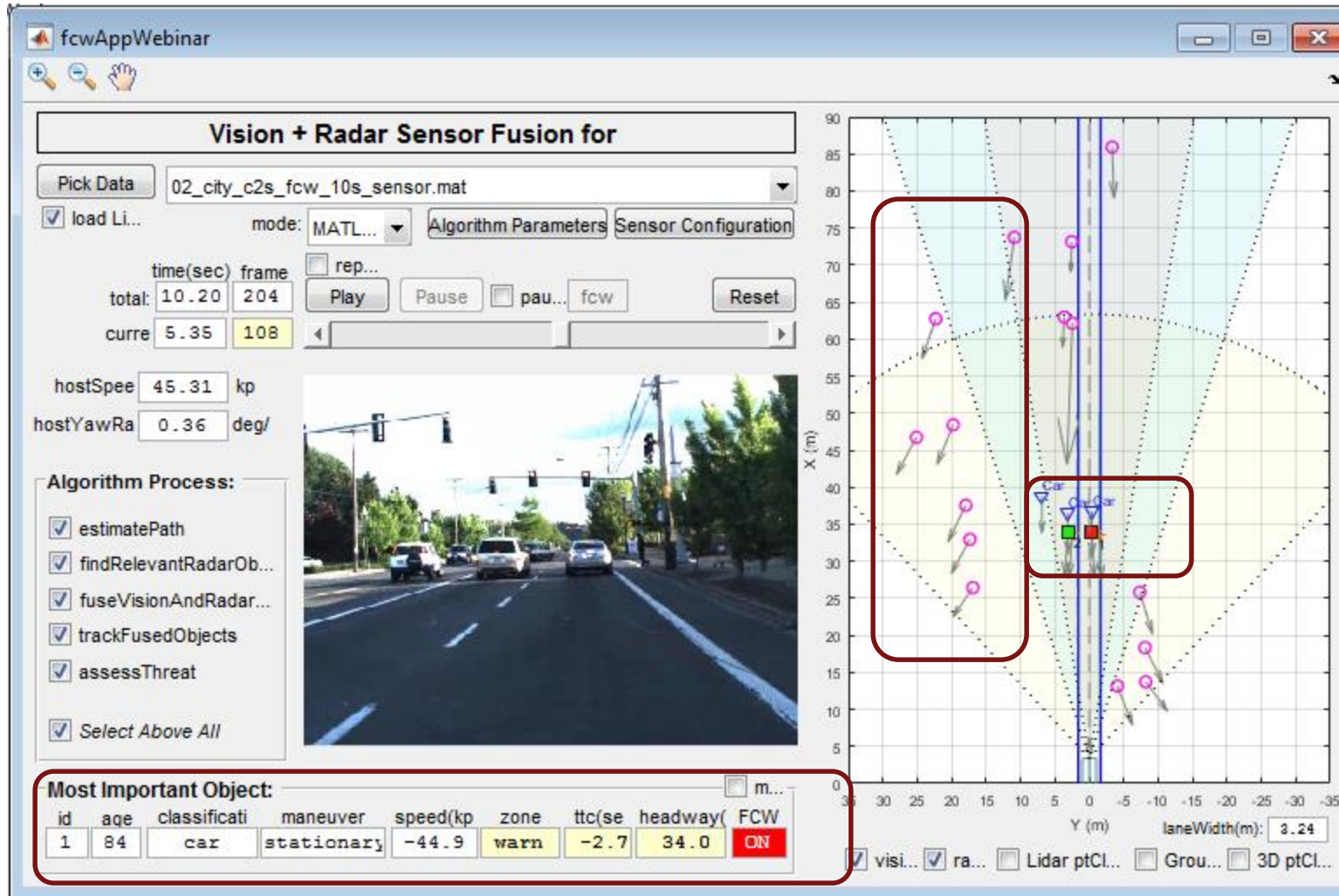
**Most Important Object:**

id	age	classificati	maneuver	speed(kp)	zone	ttc(se)	headway(	FCW
-	-	-	-	-	-	-	-	OFF

laneWidth(m): 3.30

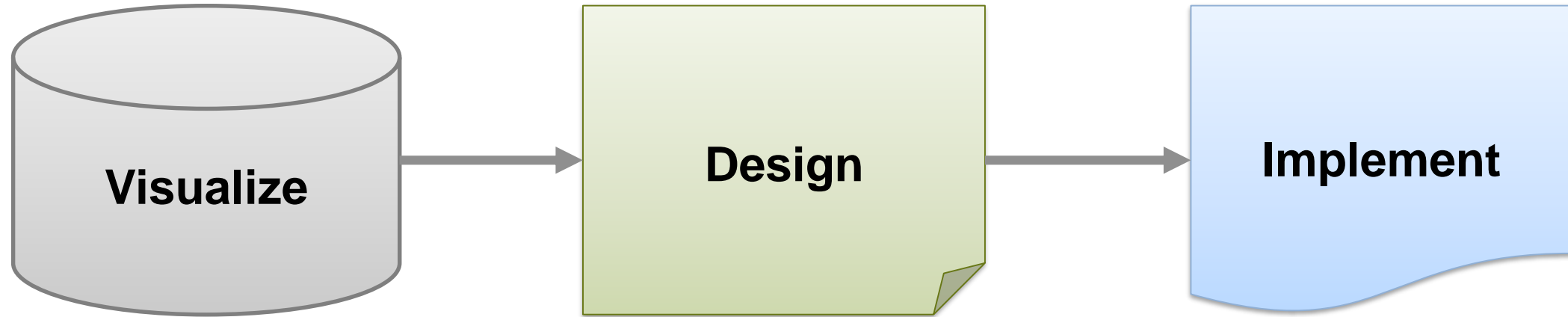
visi...  ra...  Lidar ptCl...  Grou...  3D ptCl...

# Design Algorithms with MATLAB



1. Filter false detections from radar
2. Sensor fusion and tracking using Kalman filters
3. Decision logic

## MATLAB Helped Us...



- Algorithms for multiple domains:
  - Computer vision
  - Radar
  - Sensor fusion
- Toolboxes just work

# Implement Algorithms in C Code



Code Generation Report

MATLAB code   Call stack   C code

Filter

Functions

- [assessThreat](#)
- [assessThreat > assessCollisionRisk](#)
- [cartesianToPolar](#)
- [checkInPath](#)
- [estimatePath](#)
- [filterYawRate](#)
- [findMostImportantObject](#)
- [findRelevantRadarObjects](#)
- [findRelevantRadarObjects > calculateGroundSpeed](#)
- [forwardCollisionWarning](#)
- [fuseVisionAndRadarObjects](#)
- [fuseVisionAndRadarObjects > associateSensors](#)
- [fuseVisionAndRadarObjects > fusionCost](#)
- [objectZone](#)
- [trackFusedObjects](#)
- [trackFusedObjects > calculateCost](#)
- [trackFusedObjects > createNewTracks](#)
- [trackFusedObjects > deleteLostTracks](#)
- [trackFusedObjects > detectionToTrackAssignment](#)
- [trackFusedObjects > getTrackStruct](#)
- [trackFusedObjects > predictNewStateOfTracks](#)
- [trackFusedObjects > updateAssignedTracks](#)
- [trackFusedObjects > updateUnassignedTracks](#)
- [trackVisionObjects](#)
- [trackVisionObjects > calculateCost](#)
- [trackVisionObjects > createNewTracks](#)
- [trackVisionObjects > deleteLostTracks](#)

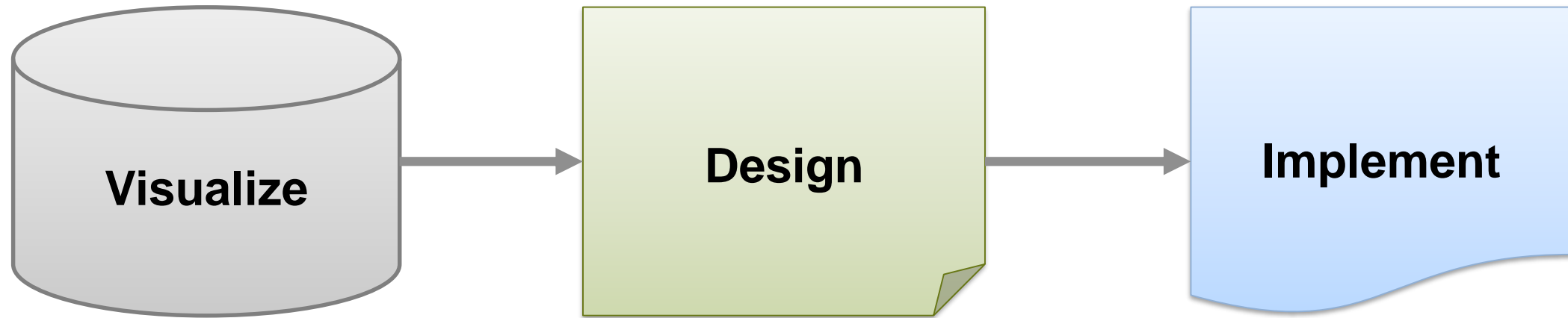
Function: forwardCollisionWarning
Calls: Select a function call:

```

1 function [mostImportantObject, ...
2     egoPath, ...
3     radarObjects, ...
4     visionObjects, ...
5     fusedObjects, ...
6     assessedThreats] = forwardCollisionWarning( ...
7         lane, ...
8         inertialMeasurementUnit, ...
9         visionSensor, ...
10        radarSensor, ...
11        params, ...
12        reset)
13 # forwardCollisionWarning: sensor fusion algorithm for Forward Collision Warning
14 #
15 # Copyright 2016 MathWorks, Inc.
16 #
17 # --- Inputs:
18 #     lane: lane marking data from ME 560
19 #         >> datainterface.lane.default
20 #
21 #     inertialMeasurementUnit: IMU data
22 #         >> datainterface.inertialMeasurementUnit.default
23 #
                
```

Summary	All Messages (0)	Variables	Target Build Log
C source code generated on: 31-Oct-2016 12:20:10			
Coding target:	Static Library		
Number of errors:	<u>0</u>		
Number of warnings:	<u>0</u>		
Number of notices:	<u>0</u>		
<b>Tell Us What You Think</b>			
We value your feedback. Please take a few minutes to answer this short questionnaire regarding the Code Generation Report.			
<a href="#">&gt;&gt;Provide Feedback</a>			

## MATLAB Helped Us...



- Faster iteration by generating C code

# Design LiDAR Processing for Autonomous Driving



The MATLAB R2016a interface displays two figures related to LiDAR processing:

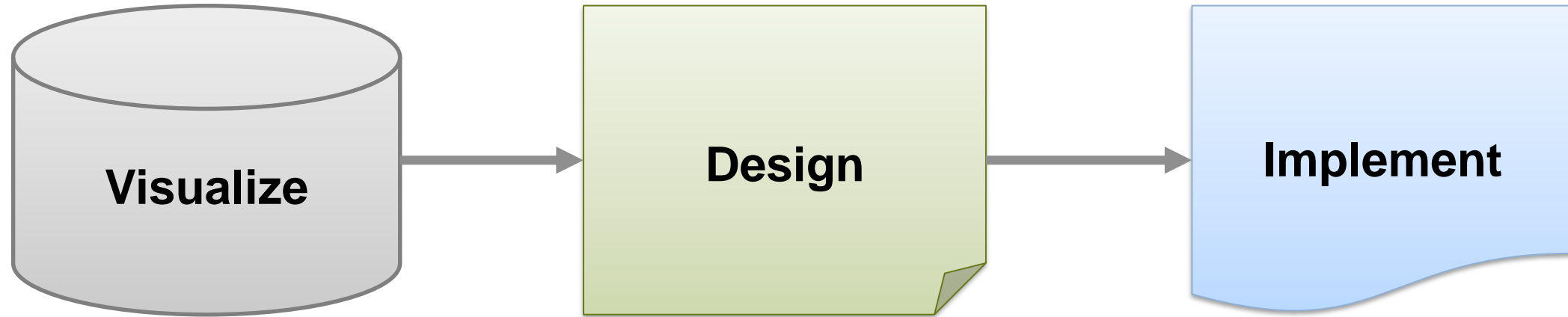
- Figure 1: Point Cloud Player** shows a 3D visualization of a LiDAR point cloud. The axes are labeled X (m), Y (m), and Z (m). The plot is titled "LiDAR 3-D Point Cloud".
- Figure 2** displays a 2D range-finder plot. The x-axis is labeled "x-Position (m)" and the y-axis is labeled "y-Position (m)". The plot shows a street scene with a camera view inset. The range-finder plot uses red and green colors to represent different data points.

# Design LiDAR Processing for Autonomous Driving



```
MATLAB R2016a
HOME PLOTS APPS SHORTCUTS EDITOR PUBLISH VIEW DemoFolder Dock Undock Search Documentation
New Open Save Find Files Compare Go To Comment Insert Breakpoints Run Run and Advance Run and Time
FILE NAVIGATE EDIT BREAKPOINTS RUN
C:\Users\anehemia\OneDrive - MathWorks\Shared with Everyone\LiDARProcessing\LiDAR\v1.2
Editor - C:\Users\anehemia\OneDrive - MathWorks\Shared with Everyone\LiDARProcessing\LiDAR\v1.2\LiDARProcessingWorkflow.m
LiDARProcessing.m x ObstacleDetectionDriveablePath.m x LiDARProcessingWorkflow.m x +
1 - clear;
2 - close all;
3 - load singlePtCloud.mat;
4 - %% Display Image
5 - figure;
6 - subplot(1,2,1); imshow(videoFrame);
7 - title('Input Video Frame');
8 - subplot(1,2,2);
9 - pcshow(lidarPtCloud);
10 - title('Lidar Point Cloud');
11
12
13 - %% Find ground plane
14 - [mdl,inlierIndices,outlierIndices] = ...
script Ln 4 Col 1
```

# MATLAB Helps Engineers...



- Visualize most sensors
  - Cameras
  - LiDAR
  - Radar
- Create custom apps and visualizations
- Algorithms for multiple domains:
  - Computer vision
  - Deep learning
  - Radar
- Toolboxes just work
- Faster iteration by generating C code



**See the demos in person at our booth**