Filtering for 3D Time-of-Flight Sensors

Larry Li

Time-of-Flight (TOF) sensors are becoming increasingly popular in 3D imaging applications such as simultaneous location and mapping (SLAM) for robotics navigation, 3D scanning and printing, occupancy detection, and auto-dimensioning. For robust operations of these applications, the TOF sensor must output relatively clean measurements. There are several ways raw TOF measurements are cleaned up, but they generally fall within two categories: system settings and filtering. This paper discusses the system settings that impact measurement, and several applicable filtering techniques.

Contents

List of Figures

1 TOF Basics

TOF sensors work by emitting modulated light at the target and measuring the relative phase between the light and its reflection. TI's TOF sensor uses a low-cost CMOS array sensitive to near-IR (850 nm) light to measure the relative phase between the emitted light and reflected light using a continuous-wave of pulses, and computes depth for each pixel in the image sensor.

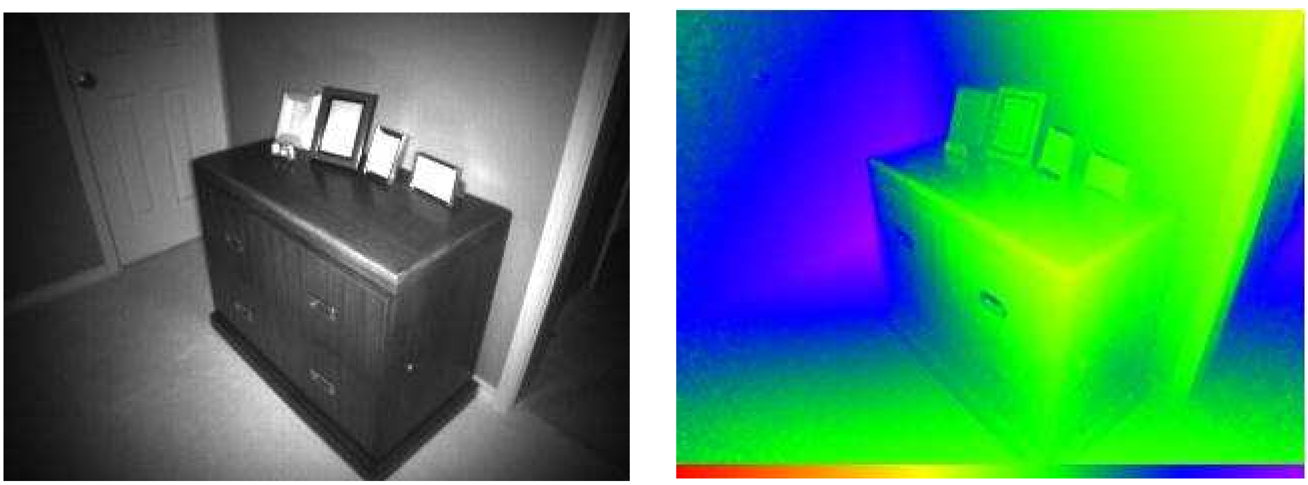
Most TOF sensors produce a phase map and an amplitude map for each frame captured. The amplitude map represents the amount of reflected light received by each pixel; the phase map represents the relative phase measured at each pixel. The phase map is easily converted to a depth map using the following formula:

$$d = \frac{c}{4 \pi f} \varphi \tag{1}$$

where

- c is the speed of light constant
- f is the modulation frequency

The depth map is then filtered and converted to point clouds by applying proper geometric transforms based on the camera’s optical properties such as the field-of-view (FOV), focal length, and lens intrinsics which accounts for optical distortion. Figure 1 shows an example of grayscale amplitude map (a) and a color-coded depth map (b).



(a) Typical Amplitude Map (b) Typical Depth Map

Figure 1. Examples of Typical Maps

Figure 2 shows the point clouds rendered from the same depth map and the amplitude map. Each pixel in the depth map is transformed to a Cartesian vertex in the 3D virtual space, and "painted" with greyscale color from the same pixel in the amplitude map.

Once these vertices or point clouds are drawn, they can be rotated, translated, meshed, and textured using modern 3D rendering techniques and OpenGL graphics accelerators available on most computers.

For more a complete description of the TOF operation, please consult the white papers referenced in Section 7.

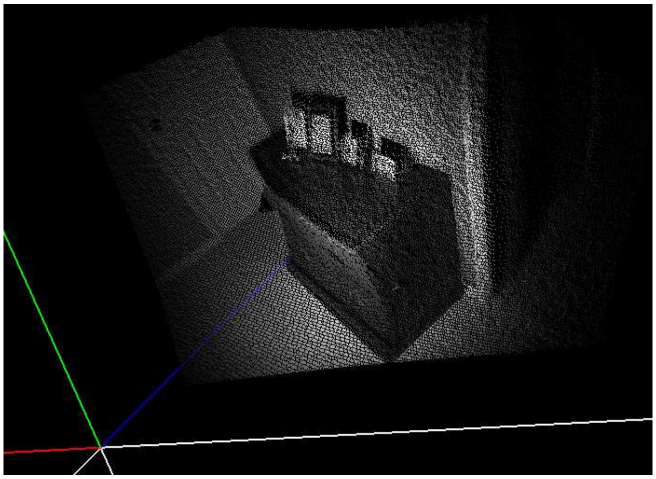


Figure 2. Point Cloud Associated With Figures 1a and 1b

2 System Settings

In general, the expected depth precision from the TOF sensor can be approximated with Equation 2:

$$\sigma \cong \frac{c}{4\sqrt{2}\,\pi f} \times \frac{\sqrt{A+B}}{c_d A} \tag{2}$$

where

- c is the speed-of-light constant
- f is the modulation frequency
- $\mathbf{c_d}$ is the modulation contrast
- **A** is the reflected amplitude
- **B** is the lumped system offset

Of these, only the modulation frequency and the illumination can be dynamically controlled. The maximum modulation frequency is limited by the type of light source used and the maximum illumination is limited by the system design and the power budget. The integration time (exposure) also affects **A**, and should be maximized. Reducing frame rate permits longer integration time, therefore, also reducing noise. With the illumination power maximized, any residual noise must be reduced using software filters, which is usually applied to the depth map before the depth map is used to construct the point clouds.

3 Filtering

Noise can be dealt with using temporal filters and spatial filters. Two common temporal filters are ***infinite-pulse-response*** filter (IIR), and ***median-average*** filter.

The *first-order* IIR filter has a very simple form:

$$x_{new} = (1 - \alpha)x_{old} + \alpha x_{in}$$

where

- α determines the response of the filter, the higher the α, the faster the output tracks to the input, therefore, high frequency noise is attenuated by lowering α. (3)

If availability of memory and CPU bandwidth is abundant, then higher-order IIR filters and FIR filters can be deployed to improve filter performance.

The *median-average* filter is a nonlinear filter that buffers multiple samples, sorts them, and then takes the median, middle value or the average of the middle values of the ordered series as the output. Bubble Sort or Quick Sort is used to sort the buffers. These sorting methods constitute the bulk of the computation. Typically, 3 to 5 samples per pixel can be buffered with acceptable computational penalty. Both IIR filter and median-average filter are done per-pixel.

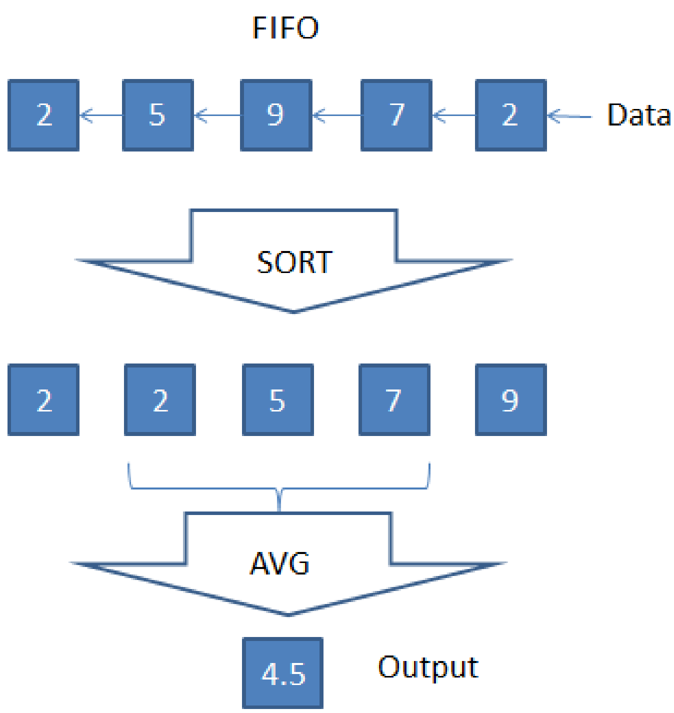


Figure 3. Median Average Filter Operation

The main downside of temporal filter is motion blurs, making them unsuitable for application with highly dynamic scenes. To handle more dynamic scenes, use a spatial filter.

4 Spatial Filters

Spatial filters attenuate noise by reducing the variance among the neighborhood of a pixel. The end result is a more smoothed surface, often at the cost of reduced contrast of the image's structural elements, such as blurred lines and points.

A ***Gaussian filter*** is probably the most common spatial filter for smoothing an image. It uses a Gaussian weighting function, G(p – p') to average the pixels, p', within a square neighborhood, w, centered about the pixel, p. The Gaussian function is a radially symmetrical function defined as:

$$G(x) = \frac{1}{2\pi\sigma^2} e^{-\left(\frac{x}{\sigma}\right)^2} \tag{4}$$

The computation works like this: for each pixel, p, the filtered output is weighted by its neighbors and itself. The neighbors are iterated by p'.

$$O(p) = \frac{\Sigma_{p'\in W} G(p-p') I_{p'}}{\Sigma_{p'\in W} G(p-p')} \tag{5}$$

A Gaussian filter can be applied to the phase map before conversion to a depth map or it can be applied directly to the depth map; whichever the choice, simply replace *I* in Equation 5 with the chosen map.

The output is then normalized by dividing the sum of the weighting factors. The end result is a blurred image, as the example in Figure 4 shows. In practice, the neighborhood is a 3×3 or a 5×5 kernel. It is also computationally efficient to pre-compute the kernel, G(p – p').

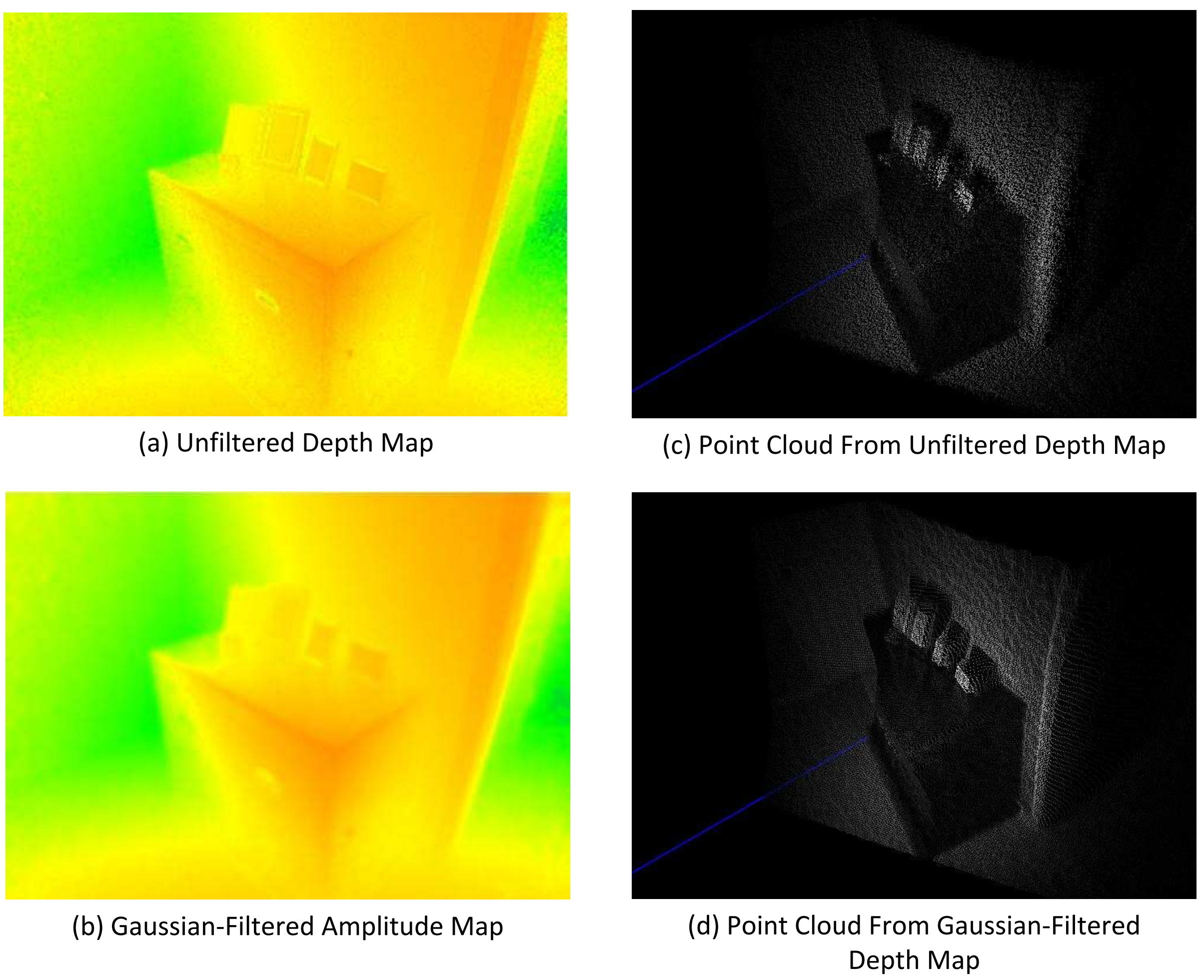


Figure 4. Smoothing Effect of Gaussian Filter

A ***Bilateral filter*** is, in essence, an edge-preserving version of the Gaussian filter, except it enhances the weighting function by adding another Gaussian term that compares the center pixel's intensity with its neighbors'.

$$O(p) = \frac{\Sigma_{p' \in W}\, G(p-p')\, G(I_p - I_{p'}) I_{p'}}{\Sigma_{p' \in W}\, G(p-p')\; G(I_p - I_{p'})} \tag{6}$$

The rationale for adding this term is this: edges represent intensity discontinuity while surfaces maintain intensity continuity. By adding an intensity term to the weighting function, neighboring pixels with greater intensity difference will contribute less to the pixel's filtered value, because they are likely at different depth, while neighboring pixels with similar intensity contribute more, because they are likely at comparable depth.

For a time-of-flight sensor, the bilateral filter defined in Equation 6 needs modification. Time-of-flight sensors produce both amplitude maps and phase maps. The amplitude map is usually a crisp stable grayscale image that can be used as I in the term, $G(I_p – I_{p'})$. But, since we are filtering the phase map, the last term in the numerator, $I_{p'}$, must be replaced with the phase map, $\varphi_{p'}$. More specifically, bilateral filter, when applied to time-of-flight sensors, should be formulated as:

$$O(p) = \frac{\Sigma_{p' \in W}\, G(p-p')\, G(I_p - I_{p'}) \varphi_{p'}}{\Sigma_{p' \in W}\, G(p-p')\; G(I_p - I_{p'})} \tag{7}$$

The previously described filter requires both amplitude map and phase map as input. The phase map can be replaced with the depth map.

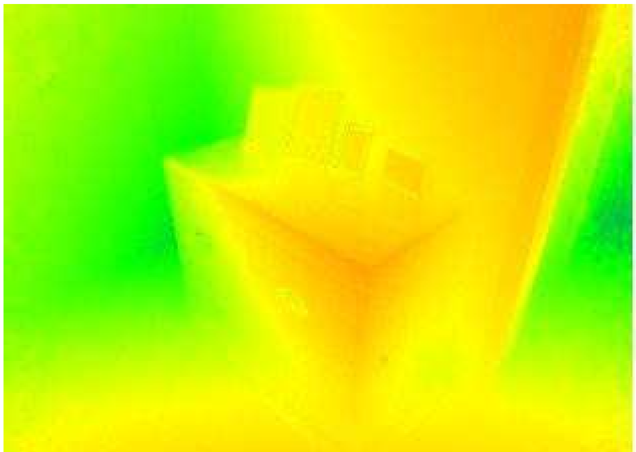


Figure 5. Bilateral Filter (Note the Sharper Boundaries of Picture Frames, as Compared to Figure 5b).

Spatial Median Filter is different from the median-average filter—whereas the median-average filter finds the median of a temporal sequence, the *spatial median* filter finds the median of a neighborhood centered about a pixel, as Figure 6 illustrates:

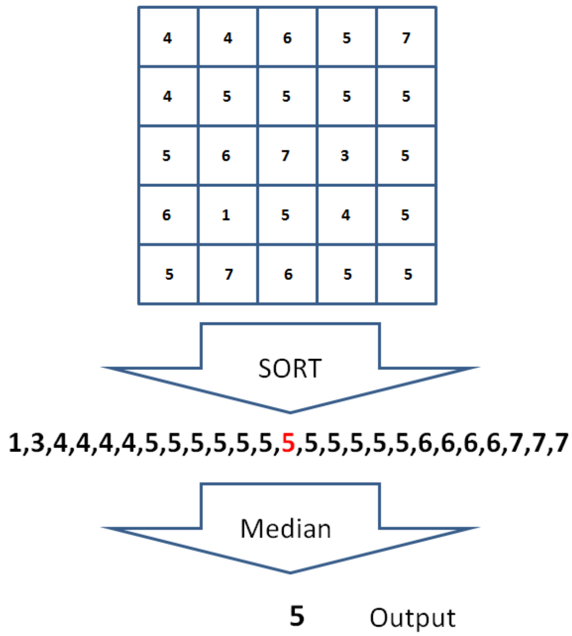
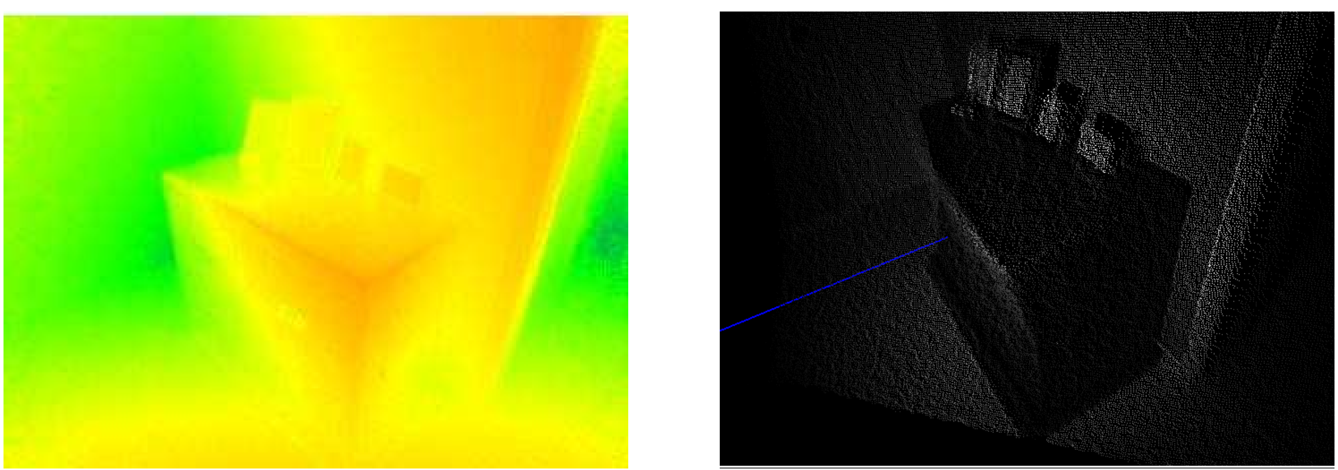


Figure 6. Spatial Median Filter Operation

Like the Gaussian filter and the bilateral filter, the neighborhood is usually a 3×3 or a 5×5 kernel, but the spatial median filter does not require an amplitude map as input, but operates directly on the depth map. Also, the median of a neighborhood is a real pixel value, unlike the interpolated values produced by Gaussian filters and bilateral filters.



(a) Depth Map From Median Average Filter (b) Point Clouds From Median Average Filter

Figure 7. Effect of Median Average Filter

5 Dead-Banding

After the software filter is applied, any residual noise appears as small jitters in the 3D point cloud. For real-time rendering, the jitter can be distracting to the viewers. Dead-banding could be applied to stabilize the point clouds.

A simple strategy to auto-adjust the dead-band is to widen it as long as too many pixels are changing. More specifically, if the number of pixels changing between two consecutive depth maps is greater than an allowable number of pixels (defined as a percentage of the total number of pixels), then widen the dead-band; otherwise, narrow it. The following heuristics implement the desired behavior:

$$\varepsilon_{new} = \begin{cases} \varepsilon_{old} - \Delta, & \mu < r \times W \\ \varepsilon_{old} + \Delta, & \mu > r \times W \end{cases}$$

where

- ξ is the dead-band
- μ is the number of pixels changing between two sequential depth maps
- r is a percentage of the total pixels
- W is the total number of pixels (8)

While dead-banding improves stability, it does not necessarily increase accuracy. Also, in previous formula, ε may increase and decrease without bound, as along as the associated condition is satisfied; therefore, it is desirable to enforce upper and lower bounds. Excessive upper bound causes the image to become insensitive to actual changes in the scene.

6 Summary

In this article, we reviewed adjustments of the system settings, such as illumination, exposure, frame rate, and confidence limit to improve the quality of depth measurements. We also reviewed temporal filters, such as IIR filter and median-average filter, and spatial filters, such as Gaussian filter, bilateral filter and spatial median filter, and discussed dead-banding to improve point cloud viewing. Appropriate use of these techniques significantly improves the system performance.

7 References

1. *Time-of-Flight Cameras—An Introduction* (SLOA190)
2. *Introduction to the Time-of-Flight (TOF) System Design* (SBAU219)

IMPORTANT NOTICE

Texas Instruments Incorporated and its subsidiaries (TI) reserve the right to make corrections, enhancements, improvements and other changes to its semiconductor products and services per JESD46, latest issue, and to discontinue any product or service per JESD48, latest issue. Buyers should obtain the latest relevant information before placing orders and should verify that such information is current and complete. All semiconductor products (also referred to herein as "components") are sold subject to TI's terms and conditions of sale supplied at the time of order acknowledgment.

TI warrants performance of its components to the specifications applicable at the time of sale, in accordance with the warranty in TI's terms and conditions of sale of semiconductor products. Testing and other quality control techniques are used to the extent TI deems necessary to support this warranty. Except where mandated by applicable law, testing of all parameters of each component is not necessarily performed.

TI assumes no liability for applications assistance or the design of Buyers' products. Buyers are responsible for their products and applications using TI components. To minimize the risks associated with Buyers' products and applications, Buyers should provide adequate design and operating safeguards.

TI does not warrant or represent that any license, either express or implied, is granted under any patent right, copyright, mask work right, or other intellectual property right relating to any combination, machine, or process in which TI components or services are used. Information published by TI regarding third-party products or services does not constitute a license to use such products or services or a warranty or endorsement thereof. Use of such information may require a license from a third party under the patents or other intellectual property of the third party, or a license from TI under the patents or other intellectual property of TI.

Reproduction of significant portions of TI information in TI data books or data sheets is permissible only if reproduction is without alteration and is accompanied by all associated warranties, conditions, limitations, and notices. TI is not responsible or liable for such altered documentation. Information of third parties may be subject to additional restrictions.

Resale of TI components or services with statements different from or beyond the parameters stated by TI for that component or service voids all express and any implied warranties for the associated TI component or service and is an unfair and deceptive business practice. TI is not responsible or liable for any such statements.

Buyer acknowledges and agrees that it is solely responsible for compliance with all legal, regulatory and safety-related requirements concerning its products, and any use of TI components in its applications, notwithstanding any applications-related information or support that may be provided by TI. Buyer represents and agrees that it has all the necessary expertise to create and implement safeguards which anticipate dangerous consequences of failures, monitor failures and their consequences, lessen the likelihood of failures that might cause harm and take appropriate remedial actions. Buyer will fully indemnify TI and its representatives against any damages arising out of the use of any TI components in safety-critical applications.

In some cases, TI components may be promoted specifically to facilitate safety-related applications. With such components, TI's goal is to help enable customers to design and create their own end-product solutions that meet applicable functional safety standards and requirements. Nonetheless, such components are subject to these terms.

No TI components are authorized for use in FDA Class III (or similar life-critical medical equipment) unless authorized officers of the parties have executed a special agreement specifically governing such use.

Only those TI components which TI has specifically designated as military grade or "enhanced plastic" are designed and intended for use in military/aerospace applications or environments. Buyer acknowledges and agrees that any military or aerospace use of TI components which have ***not*** been so designated is solely at the Buyer's risk, and that Buyer is solely responsible for compliance with all legal and regulatory requirements in connection with such use.

TI has specifically designated certain components as meeting ISO/TS16949 requirements, mainly for automotive use. In any case of use of non-designated products, TI will not be responsible for any failure to meet ISO/TS16949.

Products		**Applications**	
Audio	www.ti.com/audio	Automotive and Transportation	www.ti.com/automotive
Amplifiers	amplifier.ti.com	Communications and Telecom	www.ti.com/communications
Data Converters	dataconverter.ti.com	Computers and Peripherals	www.ti.com/computers
DLP® Products	www.dlp.com	Consumer Electronics	www.ti.com/consumer-apps
DSP	dsp.ti.com	Energy and Lighting	www.ti.com/energy
Clocks and Timers	www.ti.com/clocks	Industrial	www.ti.com/industrial
Interface	interface.ti.com	Medical	www.ti.com/medical
Logic	logic.ti.com	Security	www.ti.com/security
Power Mgmt	power.ti.com	Space, Avionics and Defense	www.ti.com/space-avionics-defense
Microcontrollers	microcontroller.ti.com	Video and Imaging	www.ti.com/video
RFID	www.ti-rfid.com		
OMAP Applications Processors	www.ti.com/omap	**TI E2E Community**	e2e.ti.com
Wireless Connectivity	www.ti.com/wirelessconnectivity		