TI Designs
3D Machine Vision Reference Design Based on AM572x With DLP Structured Light

TI Designs

This three dimensional (3D) machine vision design describes an embedded scanner which generates a 3D digital representation of a physical object based on structured light principles. A digital camera along with a Sitara™ AM57xx System-on-Chip (SoC) is used to capture reflected light patterns from a DLP4500-based projector. Subsequent processing of captured patterns, calculation of the 3D point cloud of the object, and its 3D visualization are all performed within the AM57xx processor.

This design provides an embedded solution with advantages in power, simplicity, cost, and size over a host PC-based implementation. With built-in DSP in the AM57xx processor, this embedded solution can achieve processing time similar to a 2-GHz dual-core i5-class processor, at a fraction of the power and cost.

Design Resources

TIDEP0076	Design Folder
TIDA-00254	Product Folder
TIDA-00361	Product Folder
AM57x Software Development Kit (SDK)	Product Folder

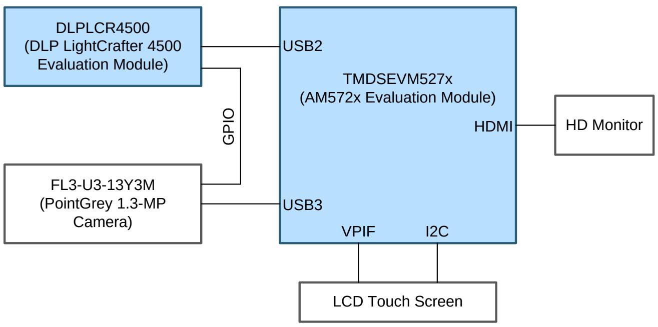
TI E2E™ Community — ASK Our E2E Experts

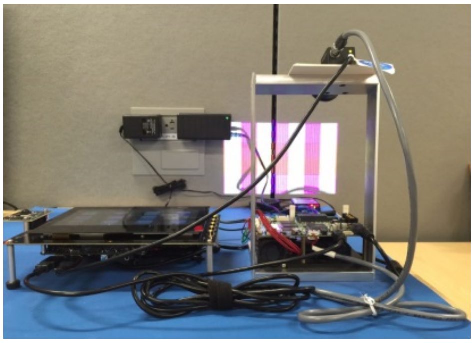
Features

- Live Demonstration of 3D Machine Inspection With DLP® Technology-Based Structured Light Techniques
- Fully Embedded 3D Machine Vision System Based on Sitara AM57xx SOC and DLP Lightcrafter™ 4500. No PC Required
- Provides 3D Object Point Cloud
- Demonstrates up to 1.3 MP Camera Capture at 60 fps
- Supports up to 4K fps of Structured Light Pattern Display
- Demonstrates Organization and Rendering of 3D Point Clouds Using Integrated Processor Cores and Graphics Processor in AM572x SoC
- Supports Display of Measured 3D Object in HD Monitor
- Demonstrates TI's Software Framework and Libraries for Ease of Integration

Applications

- Inline Inspection (AOI, SPI)
- Metrology
- Industrial Factory Automation
- Dental Scanners





An IMPORTANT NOTICE at the end of this TI reference design addresses authorized use, intellectual property matters and other important disclaimers and information.

1 Introduction

This design is based on principles and algorithms described in the TI Design TIDA-00254. Instead of a host PC, this design uses an AM5728-class embedded application processor to process captured images and display a generated 3D point cloud model.

Structured light is an optical method of 3D scanning where a set of patterns is projected upon an object, and the resulting image is captured with an imaging sensor that is offset from the projector. Structured light is an effective method to obtain accurate dimensional measurements. Structured light is widely used in applications where surface defects are detected and classified, as well as in conditions where objects or surfaces of objects do not have geometric features to extract. Typical examples of structured light-based 3D scanner systems are:

- Factory automation
- In-line inspection
- Metrology
- Surface inspection
- Metrology and parts measurement
- Agricultural produce inspection
- Dental oral scanner

Related technologies for 3D measurements are:

- Stereo vision
- Time-of-flight (TOF)

Table 1 lists a comparison of 3D scanning technologies for different areas of applications, as well as advantages and restrictions when used. Cost comparisons are relative to each other, based on typical implementations on TI devices.

Table 1. Methods for 3D Scanning and Measurement

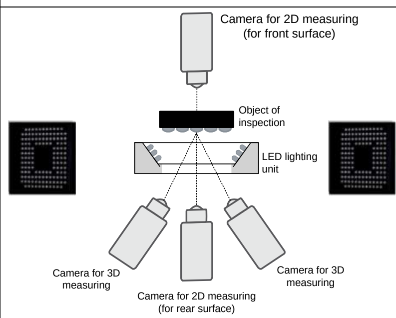
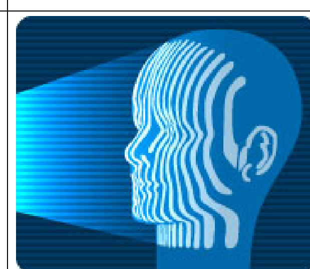
	STEREO VISION	**3D TOF**	**3D DLP**
Key Principles	Stereo disparity	TOF	Structured light
Typical Application	Broad range – ADAS, industrial, consumer, UAV	RGBZ, RGBD camera Industrial, consumer, robotics	Industrial inspection, metrology
Example Systems			
Depth Accuracy	mm to cm Difficult with smooth surface	mm to cm Variable patterns and different light sources improve accuracy	µm to cm Depends on resolution of sensor
Scanning Speed	Medium Limited by software complexity	Medium to fast Limited by camera speed	Fast Limited by image acquisition speed
Distance Range	Mid-range	Short to long range	Very short to mid-range
Low Light Performance	Weak	Good	Light source dependent
Outdoor Performance	Good	Fair Depends on illumination power	Weak to fair Depends on illumination power
Software Complexity	High	Low	Medium to high

Table 1. Methods for 3D Scanning and Measurement (continued)

	STEREO VISION	3D TOF	3D DLP
Pros	• Widely used across various applications • Wide range of software and hardware components available • Can be easily implemented on a mobile processor	• Better spatial resolution than stereo vision • Can design light sources for the specific scenarios and field of view • Can be used day or night, rain or shine • Lower power for applications that requires an Always-ON vision sensor, similar to Siri® or Cortana® in the voice side • The computing is significantly simpler than stereo vision	• Can identify anomalies in a flat surface • Can design light source to optimize reflection for the targeted objects • No interference • Allows projection of multiple patterns on the same object to extract features • Allows creation of complex patterns • Allows adaptive pattern generation
Cons	• Not usable in dark environment or adverse weather conditions • Objects required to have identifiable geometric features • May give erroneous depth if background and object mix colors	• Interference between multiple units • Needed lenses for both the sensor and light sources • Power of the light sources usually exceeds processor power significantly	• Specialized system typically targeted for inspection of defects, shapes, size
Material Cost	Low	Middle	Middle to high

A structured light 3D scanner captures images of a structure-light illuminated object then calculates depth information of the object based on the distortion of the pattern by the object. Objects in view of both the camera and projector cause different rays from the camera and projector to intersect each other. This intersection can be calculated by using the gray-coded and phase-shifted ray information from the projector along with the detected ray information from the captured images. This intersection of rays determines the real point of an object in space. The geometrical ray intersection calculations are performed by a geometry module in the software for each intersecting point.

The points generated by the geometry module are stored in a collection called a point cloud. The point cloud is all of the known points from a captured scene. Software tools and algorithms can be produced to use point clouds to create solid surfaces. The point cloud can be viewed on the display device connected to the AM57x board. The MeshLab software tool can also be used for offline viewing of point clouds.

Figure 1 shows a high-level overview of principles of depth measurement based on structured light illumination. The basic steps are:

1. A structured light pattern is predesigned or adaptively generated and projected on an object.
2. A camera captures the image, where the position and angles of the camera relative to the object and light source are calibrated.
3. A processor calculates depths of the object features based on reflected light.

Light patterns can be simple vertical or horizontal binary color lines, sinusoidal intensity lines, or with more complex color spectrum, depending on expected shapes and reflective characteristics of the object.

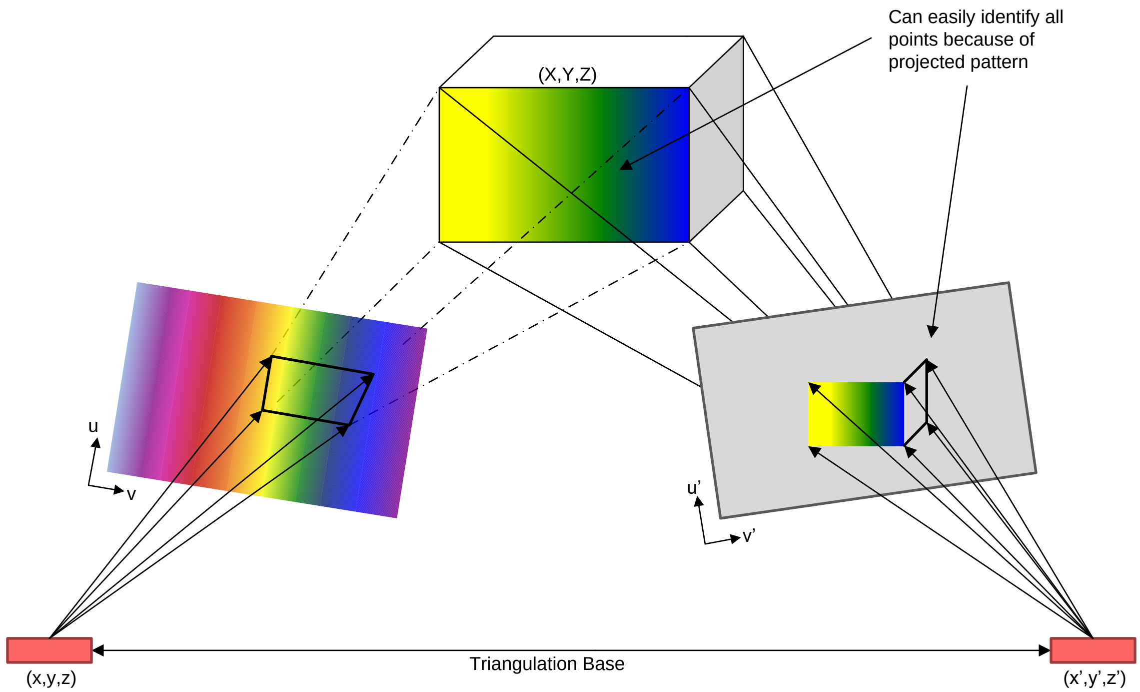


Figure 1. Principles of Depth Measurement Based on Structured Light

1.1 System Overview

1.1.1 System Description

Figure 2 shows the major components of the 3D DLP scanner design. The system is based on the following key component modules:

- Light source: projected from a DLP LCR4500 projector module
- Camera: Point Grey® 1.3 MP industrial camera, FL3-U3-13S2C or FL3-U3-13Y3M
- Processor: AM572x application processor EVM module
- Display (optional): a standard HDMI monitor driven directly from the AM5728 EVM

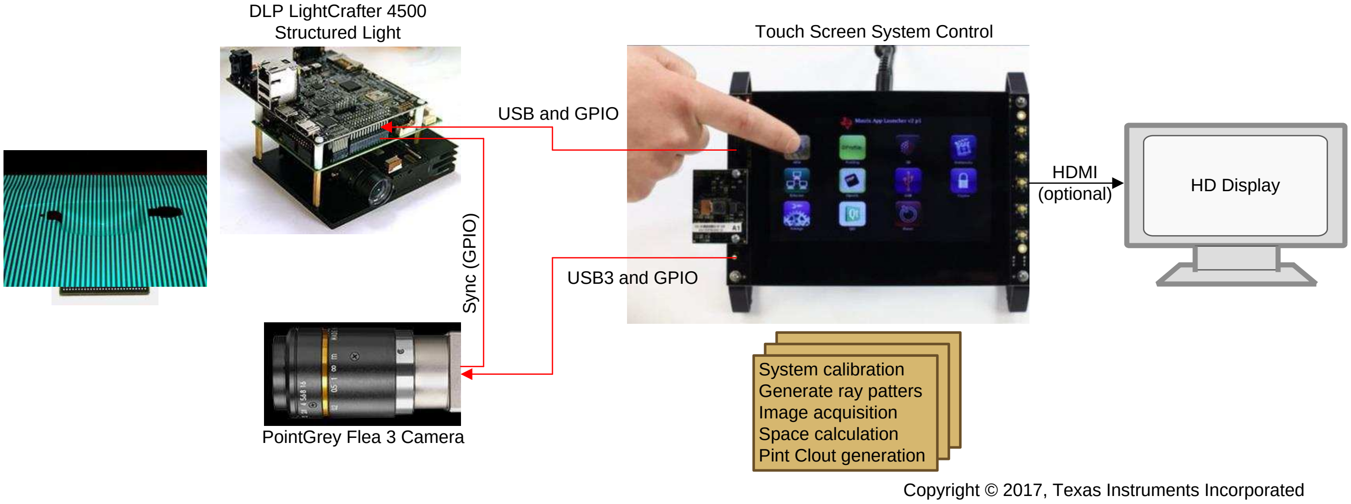


Figure 2. System Components of 3D DLP® Scanner

Figure 3 shows the internal data flow and processing functions of the design, based on a generic AM57xx application processor, with activated functions in this design highlighted. The CSI-2 interface is available only on the AM571x family of devices.

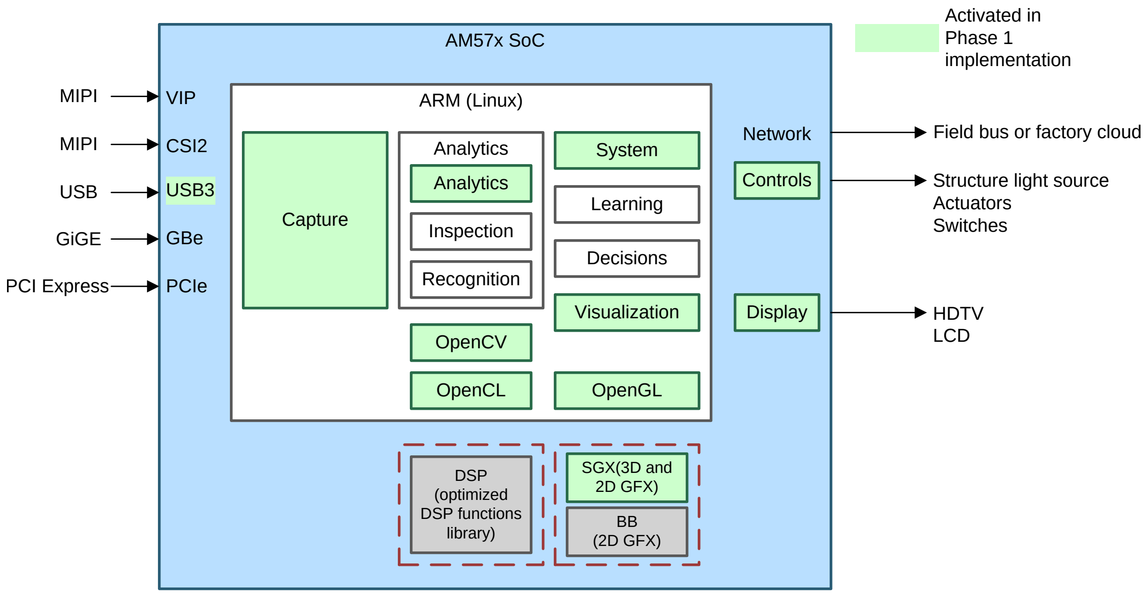


Figure 3. Internal Data Processing of 3D Scanner

1.1.2 Key System Specifications

The system enables a fully embedded design, while maximizing software reuse from equivalent PC-based systems. Key features of the system components include:

- **AM5728 Application Processor**
  - Dual-core ARM® Cortex®-A15 processors up to 1.5 GHz
  - Dual-core C66x DSP processors up to 750 MHz
  - Accelerated 3D graphics
  - Versatile image capturing capability from VIP, USB3, Gbe, or PCIe
  - Built-in HDMI transmit with PHY
  - Touch screen support
- **DLP LightCrafter 4500**
  - 18 vertical, 18 horizontal patterns, prestored
  - Projected resolution: 912 × 1140 (diamond shape)
  - Effective area after cropping: 912 × 570 (rectangular)
  - 4-KHz switching capability of light pattern generation
- **Point Grey 1.3-MP Camera [FL3-U3-13S2C or FL3-U3-13Y3M]**
  - 1280 × 1024, 60 fps mono
  - Sends out one GPIO strobe per image
- **Triggering and Synchronization**
  - GPIO-based hardware triggering from camera to the DLP projector

2 Getting Started Hardware and Software

This reference design may be recreated by acquiring the following hardware components and software modules.

2.1 Software

The 3D DLP scanner software suite performs capturing, processing, point-cloud generation, and 3D object visualization functions of the system. The development adopts a phased approach as shown in Table 2. Current revision of this design guide reflects Phase 2.

Table 2. Software Development Phases

PHASE	ARM (A15)	DSP (C66x)	DATA FLOW
Phase 1	All on A15	No DSP offloading	Sequential
Phase 2	System on A15	Partial DSP offload	Parallel camera acquisition with processing
Phase 3	System on A15	DSP offloading	Parallel

Figure 4 shows the software processing flow of Phase 1 development, where all processing is performed on the embedded A15 processor cores, except for 3D rendering which is performed on the SGX544 graphics core.

Figure 5 shows the processing flow of Phase 2 where parts of postprocessing and depth calculation are offloaded to DSP.

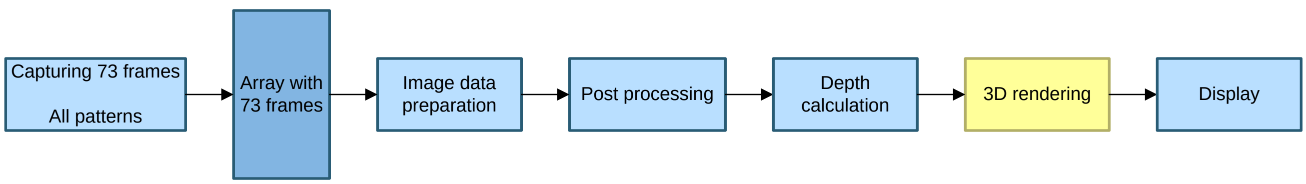


Figure 4. Phase 1 Software Processing Data Flow

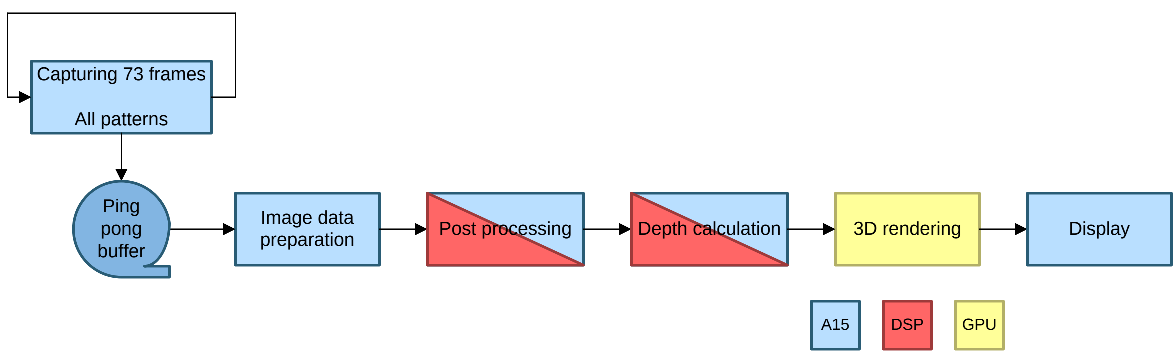


Figure 5. Phase 2 Software Processing Data Flow

2.1.1 Preparation and Control of Project Light Patterns

Pattern projection is performed by a structured light module in the DLP ALC SDK. The module generates vertical and horizontal gray-coded patterns or phase-shifted patterns which are sent to a LightCrafter 4500 projector. A firmware file with prestored patterns is prepared and statically uploaded to the projector using the LightCrafter 4500 module. For further details of how to use the DLP ALC SDK and prepare prestored light patterns see TIDA-00254.

2.1.2 Capture

Standard Point Grey driver APIs are called to capture a set of images based on a set of prestored light patterns in the DLP projector. The camera module triggers DLP light switching through a GPIO cable, and then stores all N captured frames through the USB3 to the AM57x.

2.1.3 Image Data Preparation

The Point Grey Flea3® driver API is called to convert native camera file format to standard Linux® image file format.

2.1.4 Post Processing

Captured frames are cropped and clean images are identified, matching the prestored light patterns.

2.1.5 Depth Calculation

Trigonometry and geometric functions are performed to calculate depths of the objects.

2.1.6 3D Point Cloud Generation

A 3D mesh file is generated based on the calculated depth information.

2.1.7 Display

A generated 3D mesh file is rendered by the integrated SGX544 graphics processor core, and played out on connected LCD or HDMI display monitors.

2.1.8 System Calibration

The system must be calibrated to model correct position and angles for a fixed setup. The DLP 3D Scanner SDK contains a calibration module to estimate intrinsic and extrinsic parameters of both the camera and projector. Examples of estimated parameters include focal point, lens distortion, and spatial orientation of the camera to the projector. The calibration routine must be performed any time the projector and camera change orientation with each other, or the devices are replaced.

2.2 Hardware Components

The following hardware modules are required:

- DLP LightCrafter 4500 EVM module (TI Part No. DLPLCR4500EVM), available from http://www.ti.com/tool/dlplcr4500evm
- AM572x Evaluation module (TI Part No. TMDSEVM572X), available from http://www.ti.com/tool/TMDSEVM572X
- Standard HD display monitor with HDMI port, if the optional LCD module is not available
- USB3 Industrial Camera module. In this implementation, a Flea3 camera module from Point Grey is used (https://www.ptgrey.com/flea3-usb3-vision-cameras). Specific models of FL3-U3-13S2C and FL3-U3-13Y3M are tested. A standard M0814-MP2 c-mount fixed lens from Computar® is used for the camera module. Other camera modules may be used (though they have not been tested in this design), as long as standard ARM-Linux USB3 drivers are provided.
- Mini-USB cable to connect the AM57x EVM to the DLP projector
- Micro-USB 3.0 cable to connect the AM57x EVM to the camera module
- Trigger cable, which can be assembled using instructions from the TIDA-00254 Camera Trigger Cable Assembly Guide: (Drawing No. 2514095)

2.3 Assembling Hardware Setup

Connect the hardware as follows:

1. Connect the LightCrafter 4500 to the AM57x EVM USB port (either USB 2.0 or USB 3.0 port is valid) using the USB 2.0 cable.
2. Connect the Point Grey camera to the AM57x EVM USB 3.0 port using the USB 3.0 cable.
3. Connect the HDMI cable to the AM57x EVM, and power on the EVM and monitor.
4. Connect the camera trigger cable to the GPIO port of the Point Grey camera and the input trigger connector J11 of the LightCrafter 4500.
5. Power the LightCrafter 4500.
6. Power the AM572x EVM.
7. See Figure 6 for hardware setup.

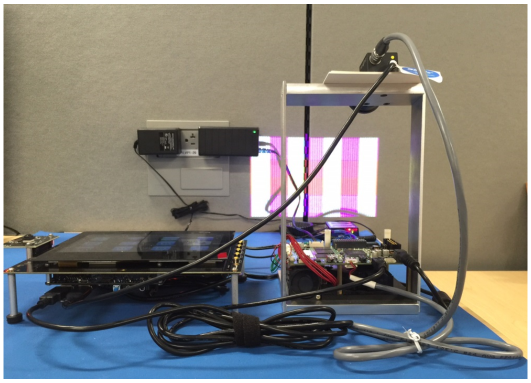


Figure 6. Hardware Setup of 3D Machine Vision Reference Design

2.4 Obtaining and Installing Software Modules

The system requires TI Processor SDK version 4.04 and later, the AM57x 3D scanner software package, as well as the camera module driver software. Follow these steps for obtaining processor SDK version 4.04 and later:

1. Download the *AM57xx Linux SDK Essentials* of PROCESSOR-SDK-LINUX-AM57X 04_00_00_04 from: http://software-dl.ti.com/processor-sdk-linux/esd/AM57X/latest/index_FDS.html.
2. Follow the *AM57xx Linux SDK SD Card Creation* link from the previous link to create an SD card file system for the EVM.
3. Download and install the USB3 camera driver from the camera vendor. If the Point Grey camera module is used, a standard Linux ARM driver software can be obtained from: https://www.ptgrey.com/support/downloads.

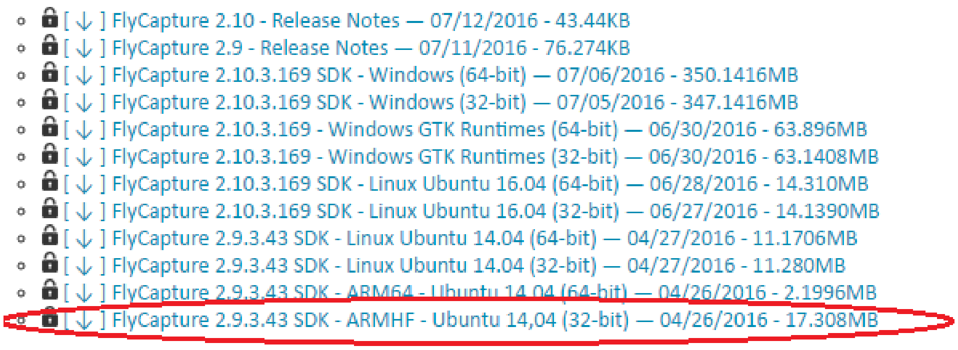


Figure 7. Available Drivers for Point Grey® Camera Module

4. Follow the README file inside the FlyCapture® package to install the package on the target AM57x file system. The FlyCapture 2.9.3.43 README file directs users to the link http://www.ptgrey.com/KB/10357 for README instructions. Go to the section *Installing the FlyCapture SDK* and follow the steps documented there. In Step 4 of the instruction set, copy the lib from the lib/C section to the system folders of the AM57x file system on target.

   Copy all libraries to system folders:

   ```
   cd flycapture-<version>_arm/lib/C
   sudo cp libflycapture* /usr/lib
   cd flycapture-<version>_arm/lib
   sudo cp libflycapture* /usr/lib/
   ```

   Also, FLIR (Point Grey) Flea 3.0 camera firmware version must be 2.14 or higher. If the firmware is older, the camera is not detected on USB3.0 port and is only detected as the USB2.0 device. Please contact FLIR (Point Grey) if the camera has older firmware.
5. Download and install the AM57x 3D Machine Vision Software Package from here: http://www.ti.com/tool/TIDEP0076.
6. Follow the README file in the package to install the application binaries. The steps are also mentioned in this guide.
7. Download and copy dlp-sdk_2.0-r11.0_armv7ahf-neon.ipk to the AM5728 file system.
8. Install (on the target) the package using opkg install dlp-sdk_2.0-r11.0_armv7ahf-neon.ipk (users may need to use options --force-depends or --force-reinstall). Files are installed to /usr/share/ti/dlp-sdk on the target file system.

[Optional step] The previous package comes with the prebuilt binary for the dlp-sdk application. The application can be recompiled by following these steps:

1. Refer to *Processor SDK - Building the SDK* at http://processors.wiki.ti.com/index.php/Processor_SDK_Building_The_SDK, to set up the build environment and get familiar with how to use Bitbake. Ensure to build arago-core-tisdk-image (MACHINE=am57xx-evm bitbake arago-core-tisdk-image);
2. Download the DLP-SDK source tarball (dlp-sdk-2.0.tar.gz) of this design guide from: http://www.ti.com/tool/TIDEP0076.
3. Place this tarball under the oe-layersetup/downloads directory, and touch oe-layersetup/downloads/dlp-sdk-2.0.tar.gz.done.
4. The recipe (dlp-sdk_2.0.bb) for building the DLP-SDK from the release tarball is also located under the design file folder at: (http://www.ti.com/tool/TIDEP0076). Create the folder oe-layersetup/sources/meta-processor-sdk/recipes-ee/dlp-sdk, and place the recipe above there.
5. The DLP-SDK has dependency on FlyCapture (Point Grey or FLIR ARMv7 Linux shared libraries). They are released under commercial license and can be obtained from Point Grey only. Download FlyCapture.2.9.3.43_armhf.tar.gz from https://www.ptgrey.com/support/downloads.
6. Place the tarball under the oe-layersetup/downloads directory, and touch oe-layersetup/downloads/flycapture.2.9.3.43_armhf.tar.gz.done
7. Bitbake the dlp-sdk recipe: MACHINE=am57xx-evm bitbake dlp-sdk
8. After the Bitbake command is successful, ./build/ build/arago-tmp-external-linaro-toolchain/work/armv7ahf-neon-linux-gnueabi/dlp-sdk/<pv>-r<pr> is created with the source code under the *example-applications* folder, and ipks under the *deploy-ipks* folder.
9. Install the rebuilt DLP-SDK package (dlp-sdk_2.0-r11.0_armv7ahf-neon.ipk) on the target file system referring to http://processors.wiki.ti.com/index.php/Processor_SDK_Building_The_SDK#Installing_Package.

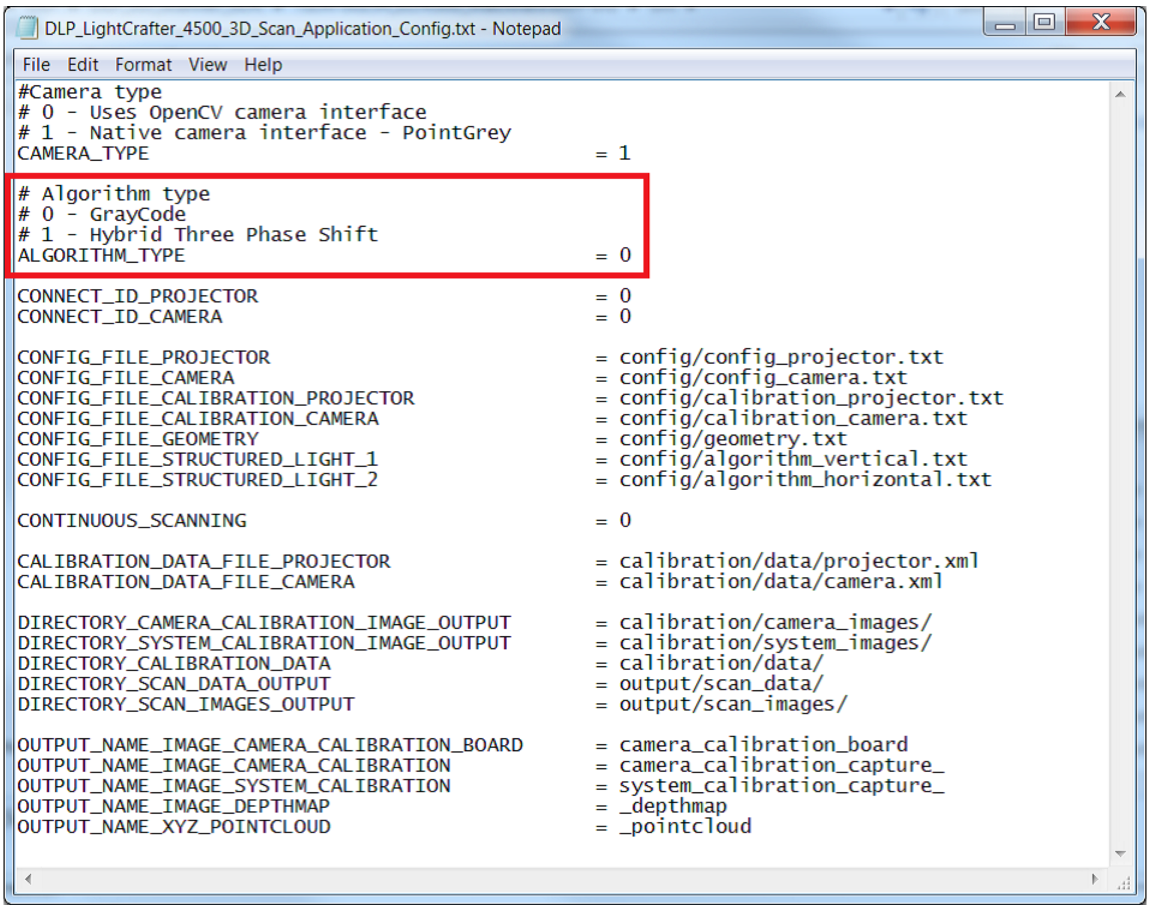
2.5 Preparation, Calibration, and Execution

Once the hardware is connected and powered up, the system is required to perform a set of calibration functions to initialize parameters based on positions, distances, and ambient light conditions of a specific setup. These steps are identical to the steps in TIDA-00254, summarized as follows.

2.5.1 Configuring Camera and Scan Type

These steps are identical to Section 3.3, *Configuration the Camera and Scan Type* in TIDA-00254. This design features two methods of scanning: binary gray code scanning and hybrid 3-phase scanning. The design also lets the user use the native Point Grey interface. To change the scan type, do the following (3-phase scanning implementation is alpha release):

1. After installing or building the design, find DLP_Lightcrafter_4500_3D_Scann_Application_Config.txt in the Build or Install folder for the reference design.
2. Open the text file.



```
DLP_LightCrafter_4500_3D_Scan_Application_Config.txt - Notepad
File Edit Format View Help
#Camera type
# 0 - Uses OpenCV camera interface
# 1 - Native camera interface - PointGrey
CAMERA_TYPE                                  = 1

# Algorithm type
# 0 - GrayCode
# 1 - Hybrid Three Phase Shift
ALGORITHM_TYPE                               = 0

CONNECT_ID_PROJECTOR                         = 0
CONNECT_ID_CAMERA                            = 0

CONFIG_FILE_PROJECTOR                        = config/config_projector.txt
CONFIG_FILE_CAMERA                           = config/config_camera.txt
CONFIG_FILE_CALIBRATION_PROJECTOR            = config/calibration_projector.txt
CONFIG_FILE_CALIBRATION_CAMERA               = config/calibration_camera.txt
CONFIG_FILE_GEOMETRY                         = config/geometry.txt
CONFIG_FILE_STRUCTURED_LIGHT_1               = config/algorithm_vertical.txt
CONFIG_FILE_STRUCTURED_LIGHT_2               = config/algorithm_horizontal.txt

CONTINUOUS_SCANNING                          = 0

CALIBRATION_DATA_FILE_PROJECTOR              = calibration/data/projector.xml
CALIBRATION_DATA_FILE_CAMERA                 = calibration/data/camera.xml

DIRECTORY_CAMERA_CALIBRATION_IMAGE_OUTPUT    = calibration/camera_images/
DIRECTORY_SYSTEM_CALIBRATION_IMAGE_OUTPUT    = calibration/system_images/
DIRECTORY_CALIBRATION_DATA                   = calibration/data/
DIRECTORY_SCAN_DATA_OUTPUT                   = output/scan_data/
DIRECTORY_SCAN_IMAGES_OUTPUT                 = output/scan_images/

OUTPUT_NAME_IMAGE_CAMERA_CALIBRATION_BOARD   = camera_calibration_board
OUTPUT_NAME_IMAGE_CAMERA_CALIBRATION         = camera_calibration_capture_
OUTPUT_NAME_IMAGE_SYSTEM_CALIBRATION         = system_calibration_capture_
OUTPUT_NAME_IMAGE_DEPTHMAP                   = _depthmap
OUTPUT_NAME_XYZ_POINTCLOUD                   = _pointcloud
```

Figure 8. Application Configuration File: ALGORITHM_TYPE

3. To perform a gray code scan, ALGORITHM_TYPE must be set to 1. To perform a hybrid 3-phase shift scan, ALGORITHM_TYPE must be 0. Figure 8 shows where the value must be changed.
4. Once a selection is made save the file, then close and reopen the reference design executable (if it was running) for the changes to take effect.

NOTE: For untriggered cameras such as a webcam, 3-phase hybrid scanning does not work due to the precise timing required. In general, any unsynchronized camera does not work with 3-phase hybrid scanning.

To change the camera interface type, do the following:

1. Follow Steps 1 and 2 for changing the scan type.
2. To change the camera type, CAMERA_TYPE, as shown in Figure 9, must be edited. Enter 0 to use the OpenCV interface and 1 to use the native camera interface. Only native camera interface is supported in this design.

DLP_LightCrafter_4500_3D_Scan_Application_Config.txt - Notepad

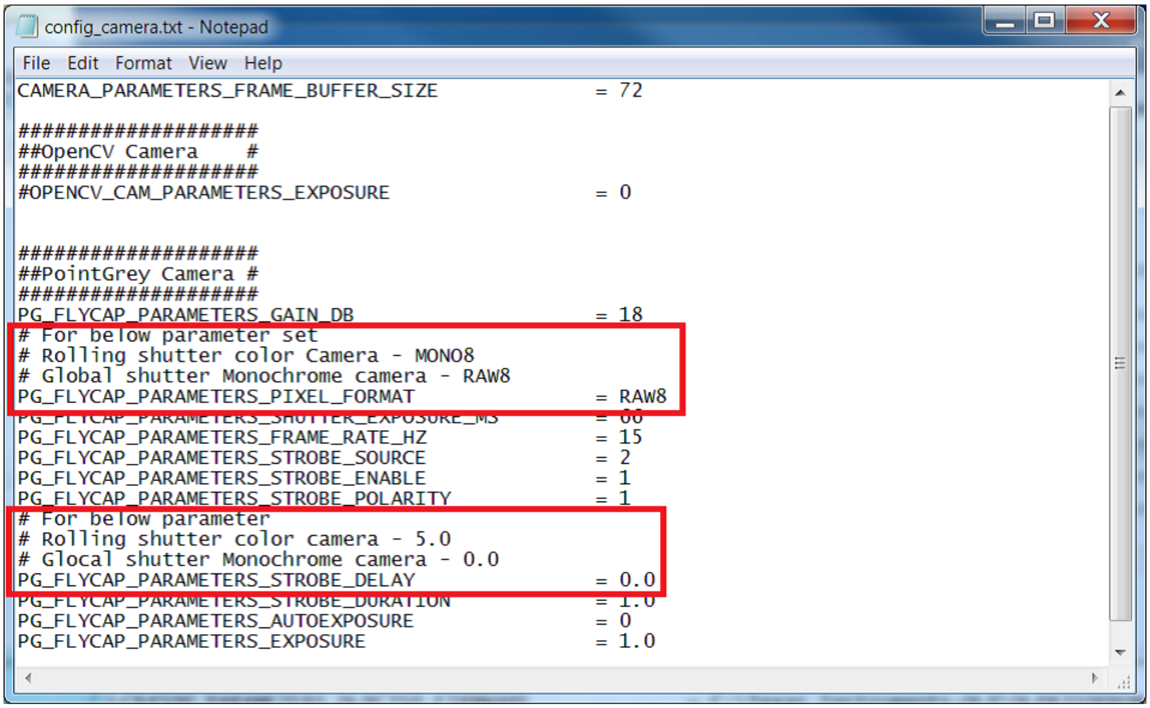
File Edit Format View Help

```
#Camera type
# 0 - Uses OpenCV camera interface
# 1 - Native camera interface - PointGrey
CAMERA_TYPE                                    = 1

# Algorithm type
# 0 - GrayCode
# 1 - Hybrid Three Phase Shift
ALGORITHM_TYPE                                 = 0

CONNECT_ID_PROJECTOR                           = 0
CONNECT_ID_CAMERA                              = 0

CONFIG_FILE_PROJECTOR                          = config/config_projector.txt
CONFIG_FILE_CAMERA                             = config/config_camera.txt
CONFIG_FILE_CALIBRATION_PROJECTOR              = config/calibration_projector.txt
CONFIG_FILE_CALIBRATION_CAMERA                 = config/calibration_camera.txt
CONFIG_FILE_GEOMETRY                           = config/geometry.txt
CONFIG_FILE_STRUCTURED_LIGHT_1                 = config/algorithm_vertical.txt
CONFIG_FILE_STRUCTURED_LIGHT_2                 = config/algorithm_horizontal.txt

CONTINUOUS_SCANNING                            = 0

CALIBRATION_DATA_FILE_PROJECTOR                = calibration/data/projector.xml
CALIBRATION_DATA_FILE_CAMERA                   = calibration/data/camera.xml

DIRECTORY_CAMERA_CALIBRATION_IMAGE_OUTPUT      = calibration/camera_images/
DIRECTORY_SYSTEM_CALIBRATION_IMAGE_OUTPUT      = calibration/system_images/
DIRECTORY_CALIBRATION_DATA                     = calibration/data/
DIRECTORY_SCAN_DATA_OUTPUT                     = output/scan_data/
DIRECTORY_SCAN_IMAGES_OUTPUT                   = output/scan_images/

OUTPUT_NAME_IMAGE_CAMERA_CALIBRATION_BOARD     = camera_calibration_board
OUTPUT_NAME_IMAGE_CAMERA_CALIBRATION           = camera_calibration_capture_
OUTPUT_NAME_IMAGE_SYSTEM_CALIBRATION           = system_calibration_capture_
OUTPUT_NAME_IMAGE_DEPTHMAP                     = _depthmap
OUTPUT_NAME_XYZ_POINTCLOUD                     = _pointcloud
```

Figure 9. Application Configuration File: CAMERA_TYPE

To change between using a global shutter monochrome camera and a rolling shutter color camera, edit the following:

1. Open config_camera.txt.
2. Figure 10 highlights the parameters that must be changed depending on the type of camera used. For a rolling shutter color camera, ensure PG_FLYCAP_PARAMETERS_PIXEL_FORMAT is set to MONO8. If the camera is a global shutter monochrome camera, set PG_FLYCAP_PARAMETERS_PIXEL_FORMAT to RAW8.
3. Similarly, set PG_FLYCAP_PARAMETERS_STROBE_DELAY to 5.0 for rolling shutter color cameras and 0.0 for global shutter mono-chrome cameras.




```
CAMERA_PARAMETERS_FRAME_BUFFER_SIZE                 = 72

####################
##OpenCV Camera    #
####################
#OPENCV_CAM_PARAMETERS_EXPOSURE                     = 0


####################
##PointGrey Camera #
####################
PG_FLYCAP_PARAMETERS_GAIN_DB                        = 18
# For below parameter set
# Rolling shutter color Camera - MONO8
# Global shutter Monochrome camera - RAW8
PG_FLYCAP_PARAMETERS_PIXEL_FORMAT                   = RAW8
PG_FLYCAP_PARAMETERS_SHUTTER_EXPOSURE_MS            = 66
PG_FLYCAP_PARAMETERS_FRAME_RATE_HZ                  = 15
PG_FLYCAP_PARAMETERS_STROBE_SOURCE                  = 2
PG_FLYCAP_PARAMETERS_STROBE_ENABLE                  = 1
PG_FLYCAP_PARAMETERS_STROBE_POLARITY                = 1
# For below parameter
# Rolling shutter color camera - 5.0
# Glocal shutter Monochrome camera - 0.0
PG_FLYCAP_PARAMETERS_STROBE_DELAY                   = 0.0
PG_FLYCAP_PARAMETERS_STROBE_DURATION                = 1.0
PG_FLYCAP_PARAMETERS_AUTOEXPOSURE                   = 0
PG_FLYCAP_PARAMETERS_EXPOSURE                       = 1.0
```

Figure 10. Camera Shutter and Color Settings

NOTE: To perform a hybrid 3-phase scan with both vertical and horizontal patterns, the exposure time of the projector and the camera must be increased to allow the frame buffer time to load. TI recommends setting the projector sequence exposure to more than 50 ms. Figure 11 and Figure 12 show some example exposure settings for the camera and projector which allow the system to perform both horizontal and vertical 3-phase scans.

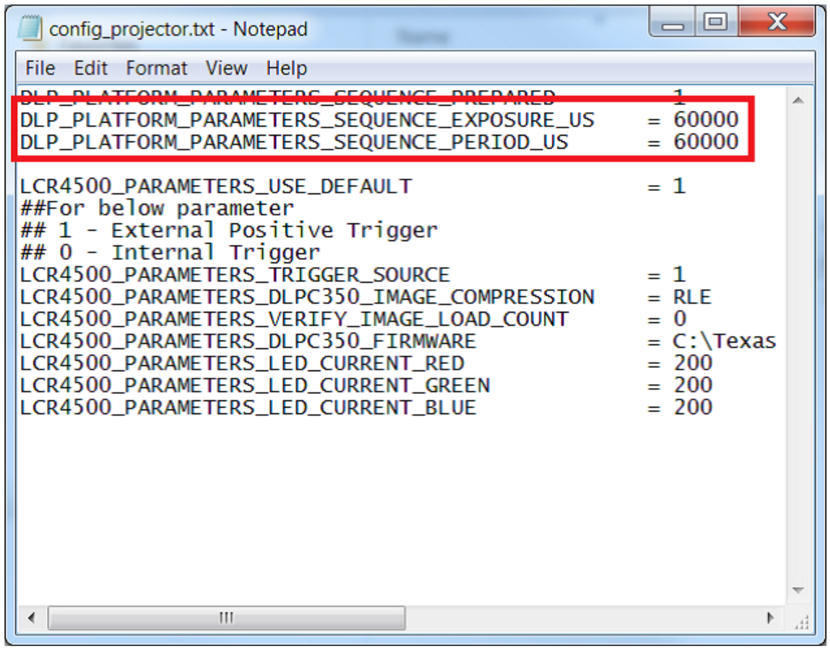


Figure 11. Projector Exposure Settings for Vertical and Horizontal 3-Phase Scanning

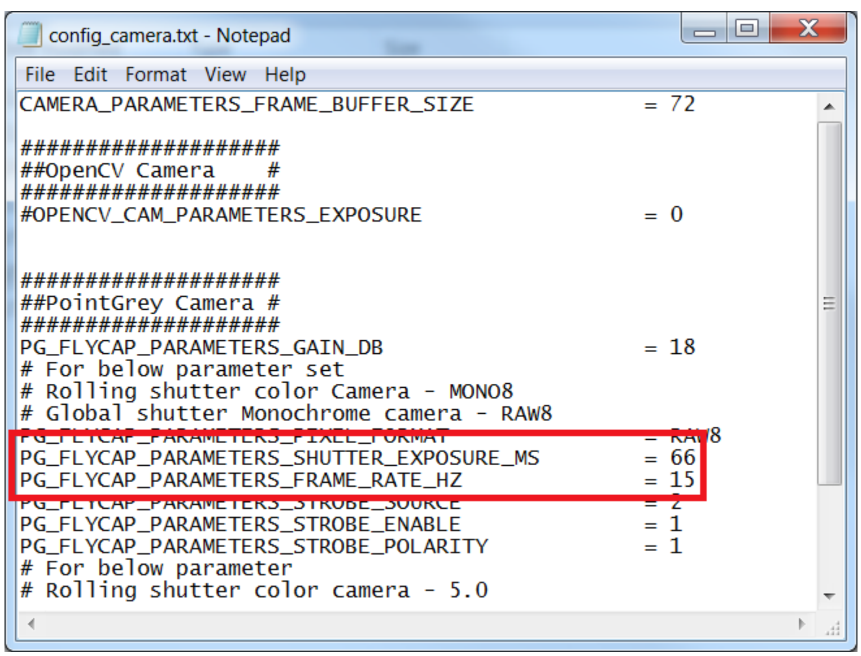


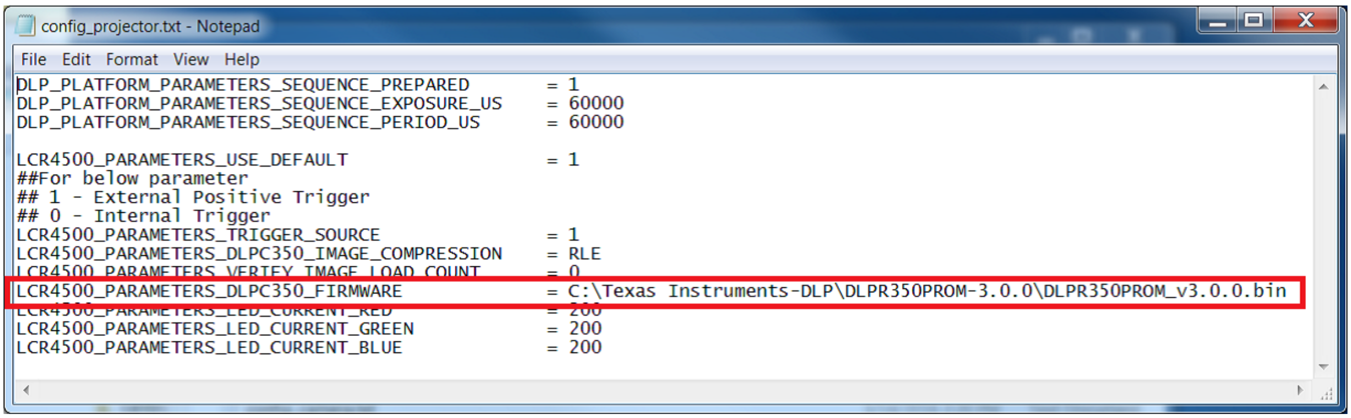
Figure 12. Camera Exposure Settings for Vertical and Horizontal 3-Phase Scanning

2.5.2 Preparing Projector

The following steps are identical to Section 3.4, *Preparing the Projector*, in TIDA-00254. The LightCrafter 4500 must be prepared with the calibration images and structured light patterns for calibration and object scanning, respectively.

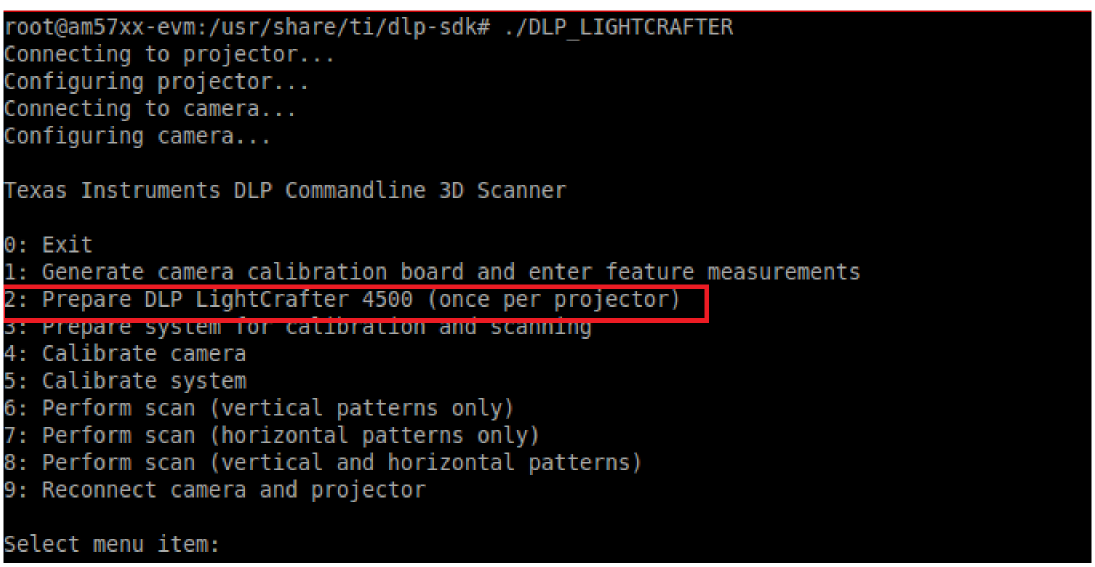
1. Before preparing the projector, download and install the firmware for the DLPC350.
2. In /config, open config_projector.txt, find the file path for the DLPC350 firmware, and then copy the whole path into the LCR4500_PARAMETERS_DLPC350_FIRMWARE parameter, as shown in Figure 13.

An example file path is: ./DLPR650PROM- 3.0.0/DLPR350PROM_v3.0.0.bin. The 3D Scanner Command Line program prepares the projector with the necessary images using menu option *2: Prepare DLP LightCrafter 4500 (once per projector)*. Enter 2 in the command line, as shown in Figure 14. A projector must be prepared only once.



```
config_projector.txt - Notepad
File Edit Format View Help
DLP_PLATFORM_PARAMETERS_SEQUENCE_PREPARED        = 1
DLP_PLATFORM_PARAMETERS_SEQUENCE_EXPOSURE_US     = 60000
DLP_PLATFORM_PARAMETERS_SEQUENCE_PERIOD_US       = 60000

LCR4500_PARAMETERS_USE_DEFAULT                   = 1
##For below parameter
## 1 - External Positive Trigger
## 0 - Internal Trigger
LCR4500_PARAMETERS_TRIGGER_SOURCE                = 1
LCR4500_PARAMETERS_DLPC350_IMAGE_COMPRESSION     = RLE
LCR4500_PARAMETERS_VERIFY_IMAGE_LOAD_COUNT       = 0
LCR4500_PARAMETERS_DLPC350_FIRMWARE              = C:\Texas Instruments-DLP\DLPR350PROM-3.0.0\DLPR350PROM_v3.0.0.bin
LCR4500_PARAMETERS_LED_CURRENT_RED               = 200
LCR4500_PARAMETERS_LED_CURRENT_GREEN             = 200
LCR4500_PARAMETERS_LED_CURRENT_BLUE              = 200
```

Figure 13. DLPC350 Firmware Parameter



```
root@am57xx-evm:/usr/share/ti/dlp-sdk# ./DLP_LIGHTCRAFTER
Connecting to projector...
Configuring projector...
Connecting to camera...
Configuring camera...

Texas Instruments DLP Commandline 3D Scanner

0: Exit
1: Generate camera calibration board and enter feature measurements
2: Prepare DLP LightCrafter 4500 (once per projector)
3: Prepare system for calibration and scanning
4: Calibrate camera
5: Calibrate system
6: Perform scan (vertical patterns only)
7: Perform scan (horizontal patterns only)
8: Perform scan (vertical and horizontal patterns)
9: Reconnect camera and projector

Select menu item:
```

Figure 14. Prepare DLP LightCrafter™ 4500 Menu Selection

Each time the 3D Scanner application is run, the system must be prepared for calibration and scanning. If option 2 has already been run for the projector, the projector can be prepared by choosing option *3: Prepare system for calibration and scanning*. Enter 3 in the menu, as shown in Figure 15.

```
root@am57xx-evm:/usr/share/ti/dlp-sdk# ./DLP_LIGHTCRAFTER
Connecting to projector...
Configuring projector...
Connecting to camera...
Configuring camera...

Texas Instruments DLP Commandline 3D Scanner

0: Exit
1: Generate camera calibration board and enter feature measurements
2: Prepare DLP LightCrafter 4500 (once per projector)
3: Prepare system for calibration and scanning
4: Calibrate camera
5: Calibrate system
6: Perform scan (vertical patterns only)
7: Perform scan (horizontal patterns only)
8: Perform scan (vertical and horizontal patterns)
9: Reconnect camera and projector

Select menu item:
```

Figure 15. Prepare for System Calibration Menu Selection

2.5.3 Creating Calibration Board

The following steps are identical to Section 3.5, *Creating the Calibration Board* in TIDA-00254. This section guides the user through the generating and measuring the camera calibration board.

1. Start the 3D Machine Vision Reference Design program, by running the executable file, as shown in Figure 14.
2. Run option *1: Generate camera calibration board and enter feature measurements* by entering 1 in the command line menu, as shown in Figure 16.

```
1


<<<<<<<<<<<<<<<<<<<<<< Generate Camera Calibration Board >>>>>>>>>>>>>>>>>>>>>>
Loading camera calibration settings...
Setting up camera calibration...
Generating camera calibration board...
Saving calibration board...

Please print the camera calibration board and attach it to a flat surface
The calibration board image can be found in: calibration/camera_images/camera_calibrp

Once the calibration image has been printed and attached
to a flat surface, measure the size of the square on the board

NOTE: Enter the measurement in the units desired for the
      point cloud (i.e. mm, in, cm, etc.)


NOTE: Both camera and system calibrations must be redone!

Enter the length of the square (do NOT include units):
```

Figure 16. Command Line Menu Prompt

3. Once the command is entered, the program generates the calibration board. Print the camera calibration board image found in the location indicated in the prompt (calibration/camera_images/camera_calibration_board.bmp). The camera calibration board is approximately half the size of the total projection area.
4. Attach the printed calibration board to a flat, white surface that is larger than the projection area, as shown in Figure 17. The number of squares on the grid can be changed in the configuration files for the program. The default grid is 7 × 10.

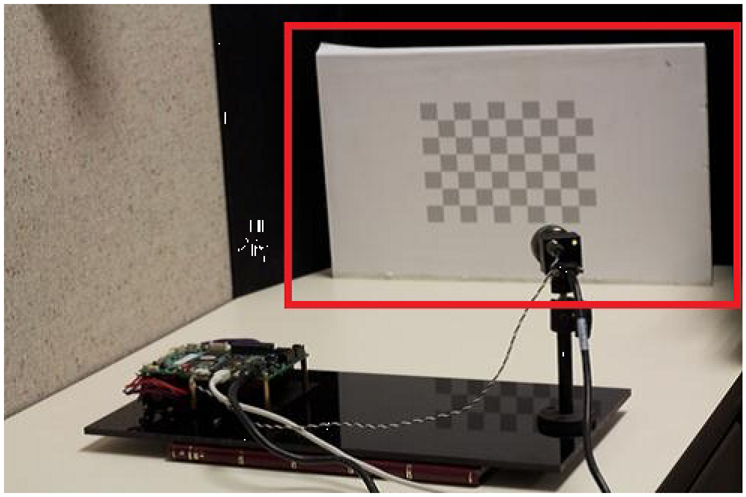


Figure 17. Calibration Board Attached to Flat Calibration Surface

5. After attaching the camera calibration board to the calibration surface, measure the length of one side of one of the squares on the grid, and type the number into the command prompt as shown in Figure 16. Do not enter any units in the command line and press the Enter button to continue.

NOTE: The generated point clouds show unit-less distances. The actual units depend on how the user measured their calibration board. For example, if each square is 2 cm wide, enter 2 into the prompt. The generated point clouds show distances which appear unit-less but are actually in centimeters.

2.5.4 Calibrating Camera

The following steps are identical to Section 3.6, *Calibrating the Camera*, in TIDA-00254. This section guides the user through the process of creating the physical connections between the LightCrafter 4500, the host PC, and the Point Grey Flea3 camera, and then calibrating the camera.

NOTE: Section 2.5.3 must be completed before the camera can be calibrated.

1. Connect the GPIO output trigger from the camera to the input trigger of the projector using the cable detailed in the TIDA-00254-CAMERA_TRIGGER_CABLE_ASSEMBLY.pdf file, as shown in Figure 18.

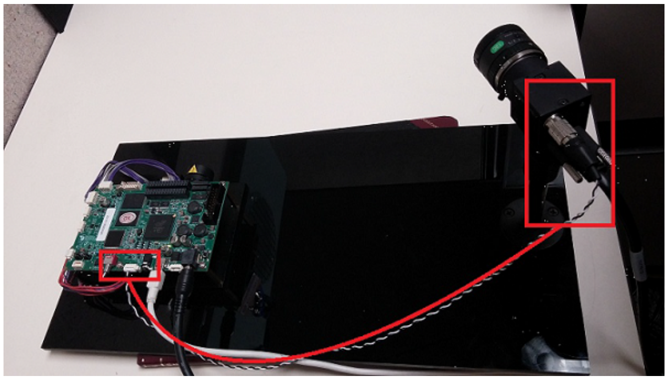


Figure 18. Connecting Camera to Host EVM

2. Connect the Point Grey Flea3 camera to the host EVM USB port.
3. Connect the LightCrafter 4500 to the host EVM USB 2.0 port.
4. Ensure there is sufficient distance between the camera and the projector. The camera and projector should be separated by a 20° to 45° angle as formed by the object being scanned, as shown in Figure 19.

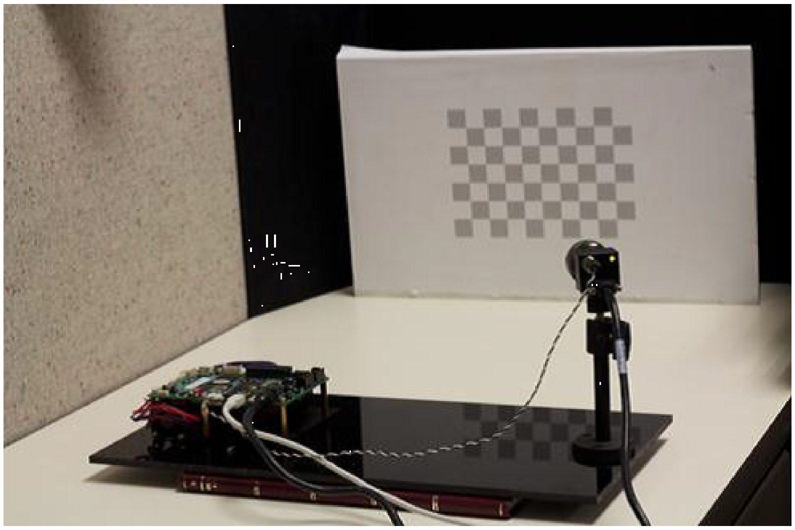


Figure 19. Projector, Camera, and Object Spatial Orientation

5. Enter menu option 4 to start the camera calibration. Follow the prompts and directions on the screen during the entire process.

6. A live camera view window appears on the host PC. Position the camera calibration board entirely in the frame, as shown in Figure 20.

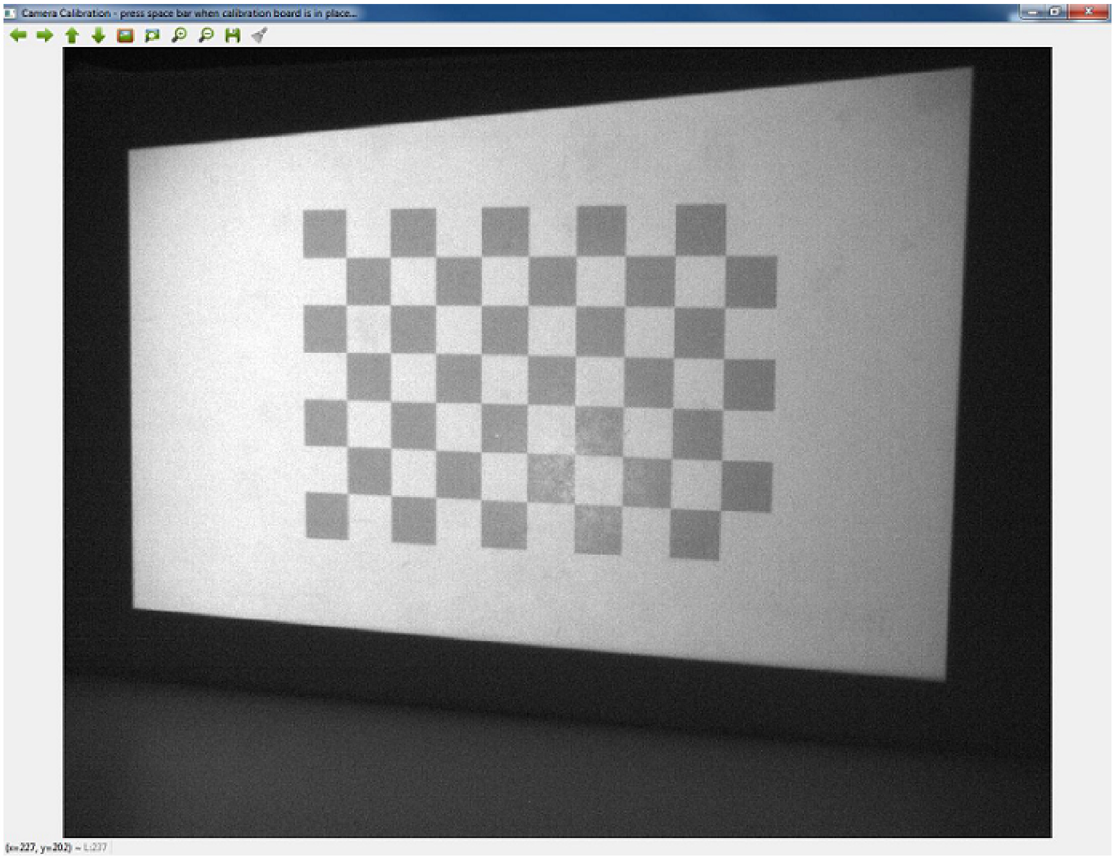


Figure 20. Camera Calibration Board Live View

7. Stop down the aperture as low as possible, while still being able to discern the gray and white squares on the calibration board and minimize all sources of glare. Ensure the projection area is in focus, then lock the aperture and focus. Figure 21 shows an example of an overexposed image and Figure 22 shows example of an underexposed image.

NOTE: If the aperture size or focus of the camera is changed after Step 7, the resulting point cloud data is impacted. Perform camera calibration routine again if the results are undesirable.

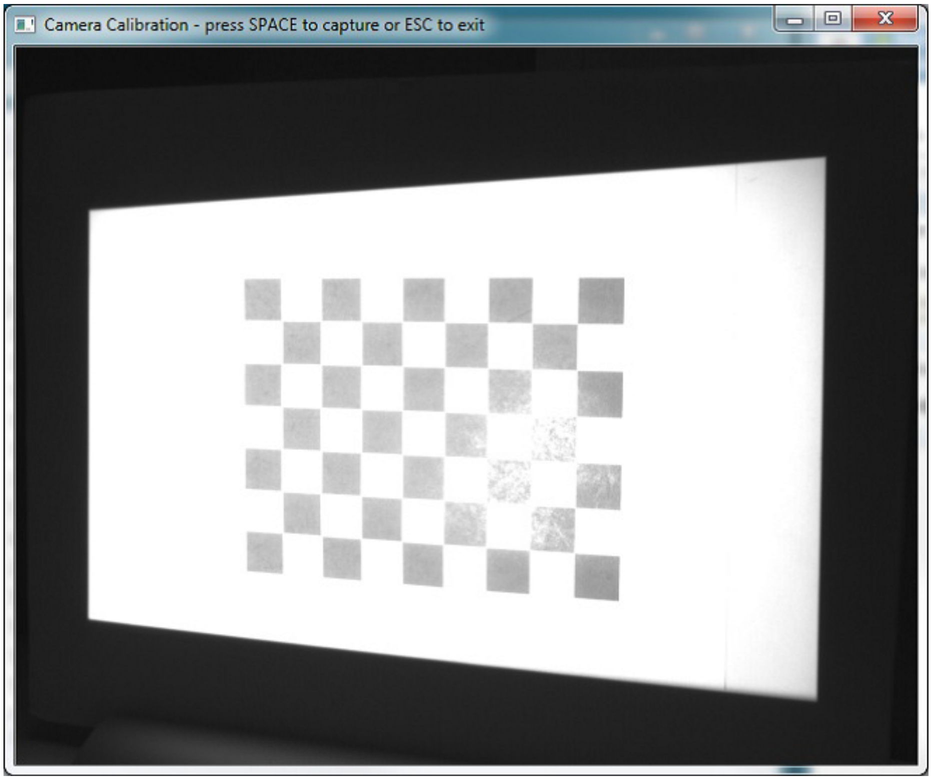


Figure 21. Overexposed Camera Capture

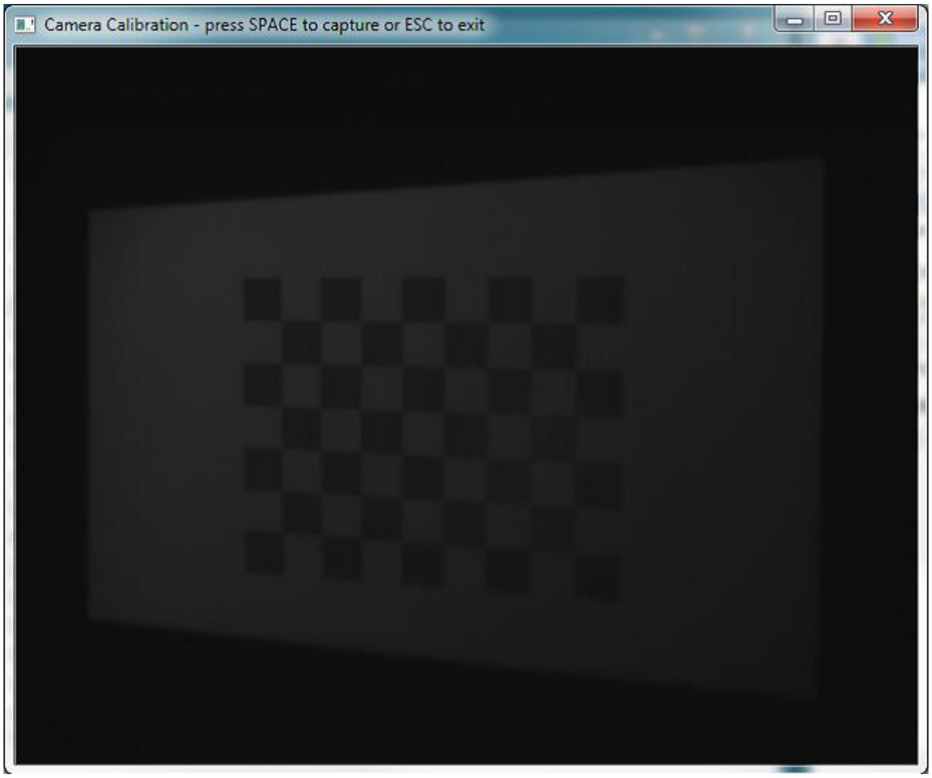
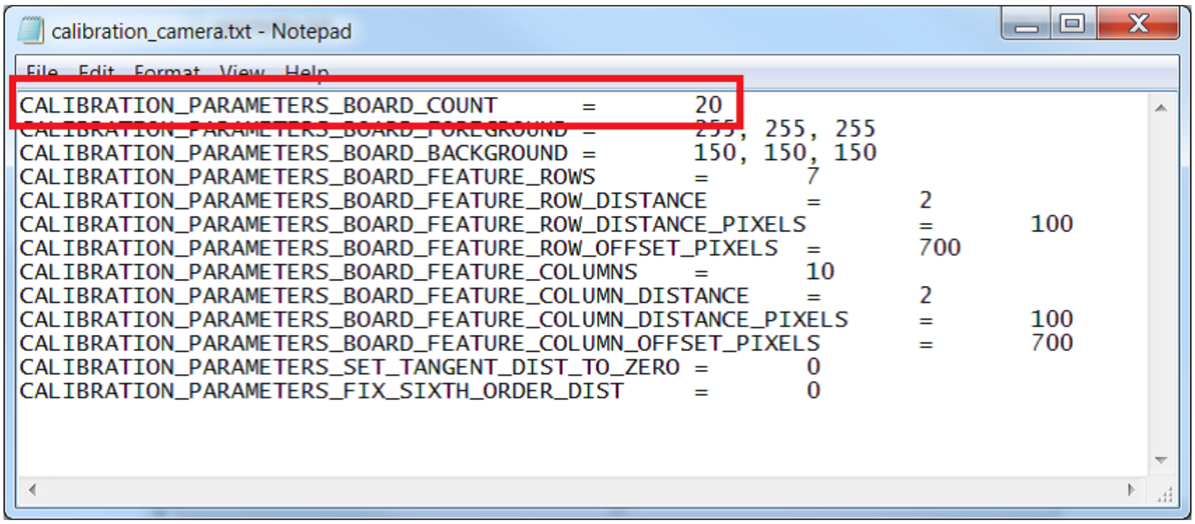


Figure 22. Underexposed Camera Capture

8. Click the live camera view window on the host PC, and verify the calibration board is in focus.
9. From the live camera view window, position the camera at varying angles and distances from the projection surface. Place the grid in different areas of the camera view and press the space bar to capture images.

Default settings require 20 calibration images although this parameter can be adjusted. In /config, find calibration_camera.txt. Figure 23 shows the parameter which specifies the number of calibration images. Figure 24 shows some recommended calibration images. It is acceptable to move the camera at this point in the calibration procedure.



calibration_camera.txt - Notepad

File Edit Format View Help

```
CALIBRATION_PARAMETERS_BOARD_COUNT          =        20
CALIBRATION_PARAMETERS_BOARD_FOREGROUND =        255, 255, 255
CALIBRATION_PARAMETERS_BOARD_BACKGROUND =        150, 150, 150
CALIBRATION_PARAMETERS_BOARD_FEATURE_ROWS        =        7
CALIBRATION_PARAMETERS_BOARD_FEATURE_ROW_DISTANCE        =        2
CALIBRATION_PARAMETERS_BOARD_FEATURE_ROW_DISTANCE_PIXELS        =        100
CALIBRATION_PARAMETERS_BOARD_FEATURE_ROW_OFFSET_PIXELS   =        700
CALIBRATION_PARAMETERS_BOARD_FEATURE_COLUMNS     =        10
CALIBRATION_PARAMETERS_BOARD_FEATURE_COLUMN_DISTANCE     =        2
CALIBRATION_PARAMETERS_BOARD_FEATURE_COLUMN_DISTANCE_PIXELS      =        100
CALIBRATION_PARAMETERS_BOARD_FEATURE_COLUMN_OFFSET_PIXELS        =        700
CALIBRATION_PARAMETERS_SET_TANGENT_DIST_TO_ZERO =        0
CALIBRATION_PARAMETERS_FIX_SIXTH_ORDER_DIST      =        0
```

Figure 23. Camera Calibration Configuration File

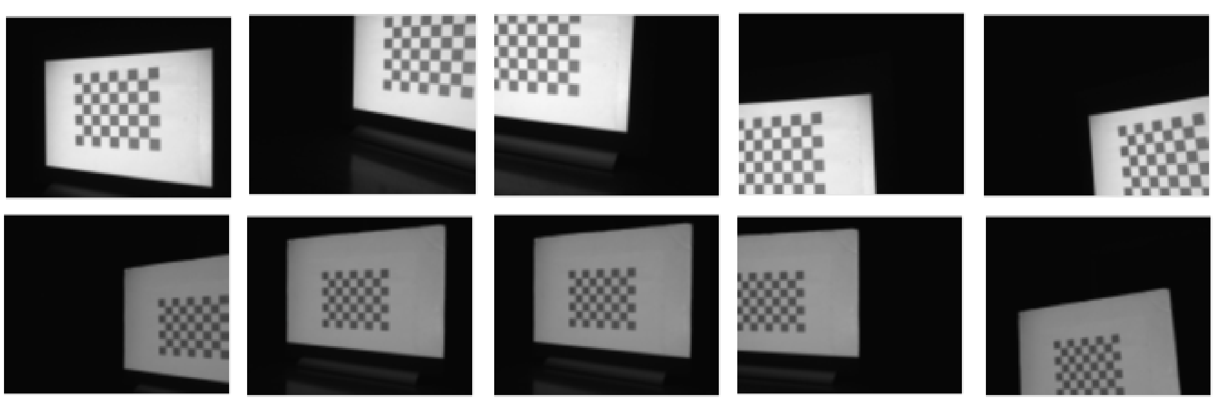


Figure 24. Calibration Board Image Capture Positions

The calibration process estimates the lens focal length, focal point, lens distortion, and the translation and rotation of the camera relative to the calibration board. The calibration procedure generates a reprojection error. Zero reprojection error is ideal, however an error below two is adequate for typical usage.

If the reprojection error is not satisfactory or if initial scans do not provide good results, run the camera calibration routine again (see Figure 25).

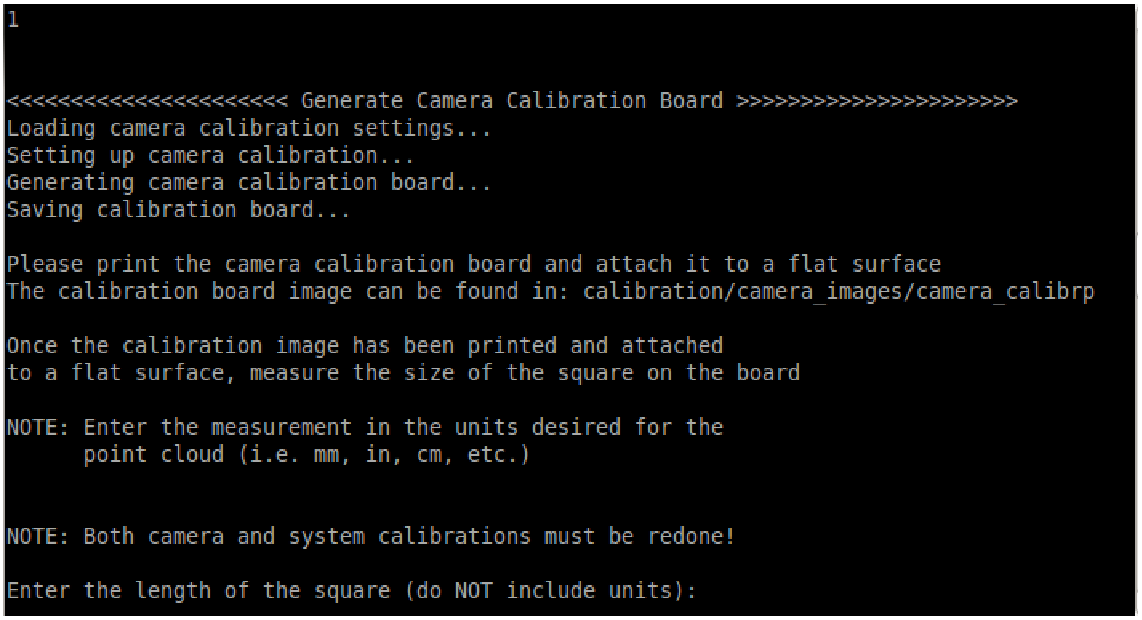


Figure 25. Screenshot for Generating Camera Calibration Board

2.5.5 Calibrating Projector

The following steps are identical to Section 3.7, *Calibrating the Projector*, in TIDA-00254. This procedure calibrates the projector and projector-camera system. Perform this procedure only with valid camera calibration already completed. When the camera calibration is complete and the projector is prepared, the system calibration can be performed.

1. Start the system calibration process by entering 5 in the command line prompt. Read the directions in the prompt in detail. Default calibration requires five images. To change this default, open calibration_projector.txt in \config. Figure 26 shows the parameter to change.

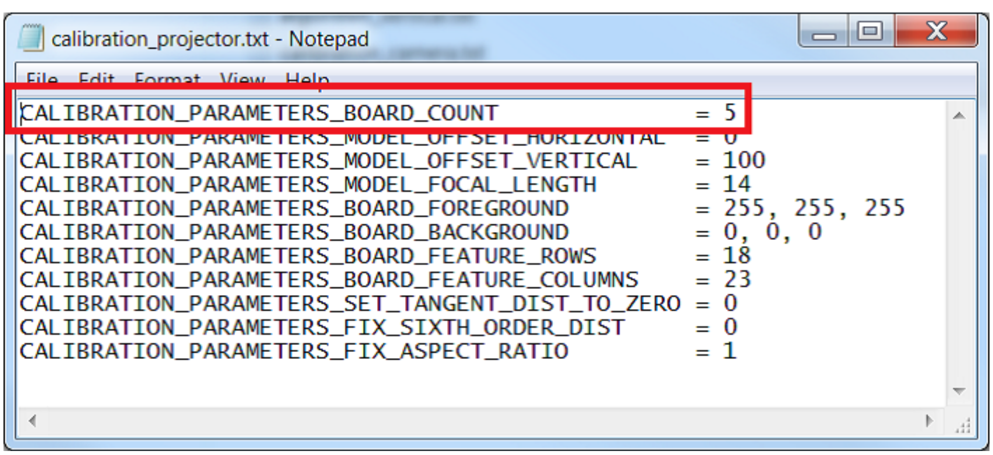


Figure 26. Number of Projector Calibration Shots

2. The projector displays a calibration board. The projected calibration board must be larger than the camera calibration board, but still fall entirely on the calibration surface. Adjust the position of the camera to center the projected calibration board in the live view (see Figure 27).
3. Select the live camera view window, and press the space bar to capture the centered calibration board. Avoid glare from the projected board or the captured image is discarded by the software. Rotate the angle of the backstop on all three axes in the captured images. Figure 28 shows some recommended projector calibration capture orientations. The camera captures three patterns after the space bar is clicked: solid white, black and white chessboard, and solid black.

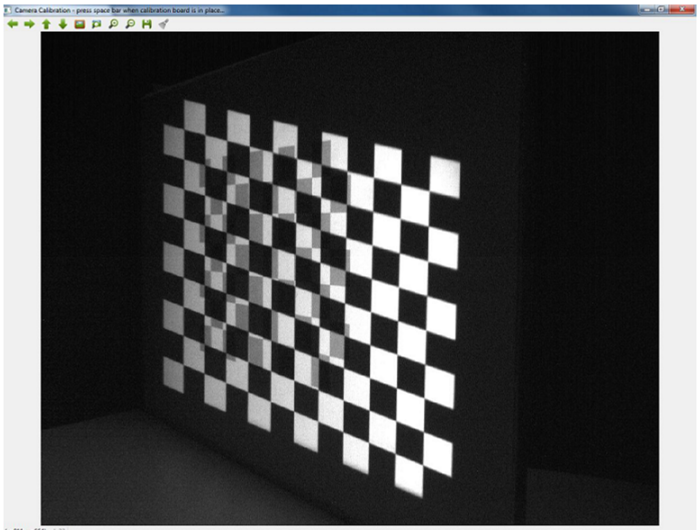


Figure 27. Projector Calibration Chessboard Capture

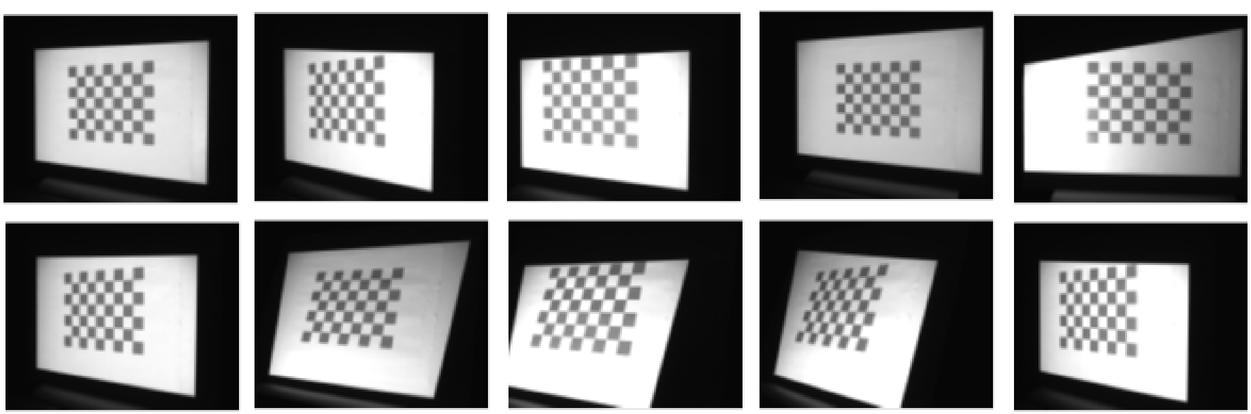


Figure 28. Projector Calibration Board Capture Positions 1 to 10

4. Repeat Step 3 five times, capturing various angles and positions of the calibration board by rotating and moving the calibration surface. Ensure the projected calibration board falls entirely on the calibration surface in each capture.

5. The system calibration process estimates extrinsic and intrinsic parameters, as well as lens distortion parameters, for the projector. The system calibration also estimates the camera-projector orientation. The calibration procedure generates a reprojection error similar to the camera calibration. Zero reprojection error is ideal, however an error below two should be adequate for typical usage. If the reprojection error is not satisfactory, verify the calibration as detailed in Section 2.5.6 before performing the calibration again.

2.5.6 Calibration Verification

The following steps are identical to Section 3.8, Calibration Verification in TIDA-00254, and were copied from there. Ignore mention of the MeshLab in the following steps for calibration on the AM57x device. The generated point cloud can be seen on the display device connected to the AM57x. Once system calibration is complete, it must be verified.

1. Scan a flat, white surface, like the backdrop for the printed calibration image, by entering the perform scan command 8 in the command line menu. The output depth map should look similar to Figure 29.

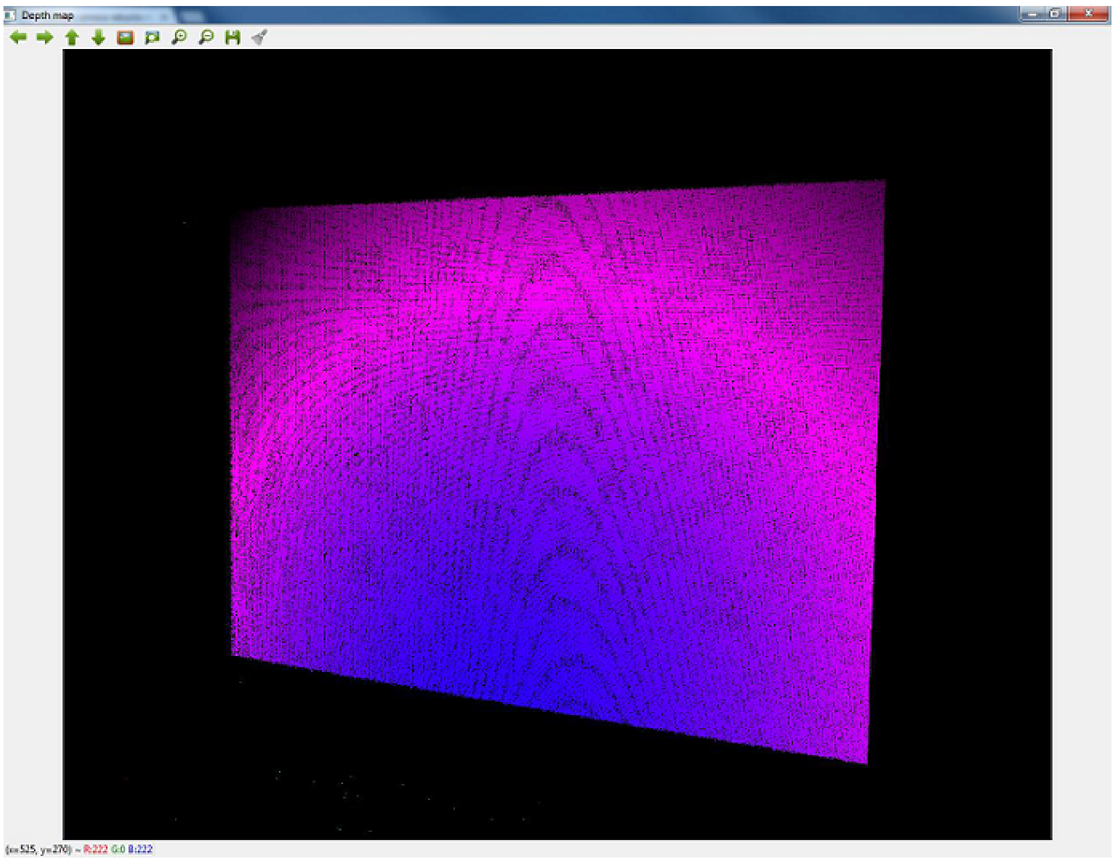


Figure 29. Typical Depth Map of Flat Surface

2. If the depth map is missing a significant amount of points, as shown in Figure 30, check the camera and projector synchronization by looking at the captured images and verifying the gray coding is displayed correctly. It is also possible that the scene was not static. Ensure that the objects being scanned are not moving.

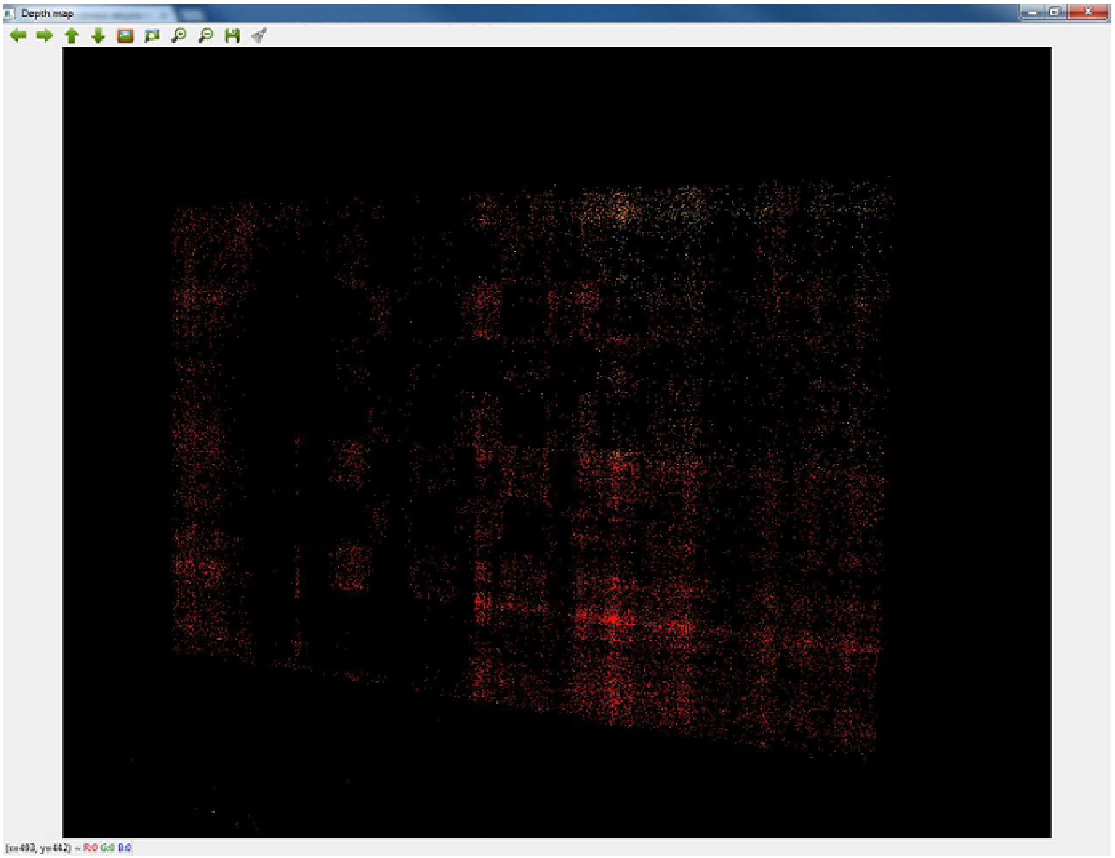


Figure 30. Deficient Depth Map of Flat Surface

Figure 31 shows a depth map with acceptable density. Optionally, users can save the point cloud (.XYZ) file and open it using MeshLab on a PC. Inspect the point cloud (on the AM57xx EVM or PC) for accurate reproduction of the scanned board.

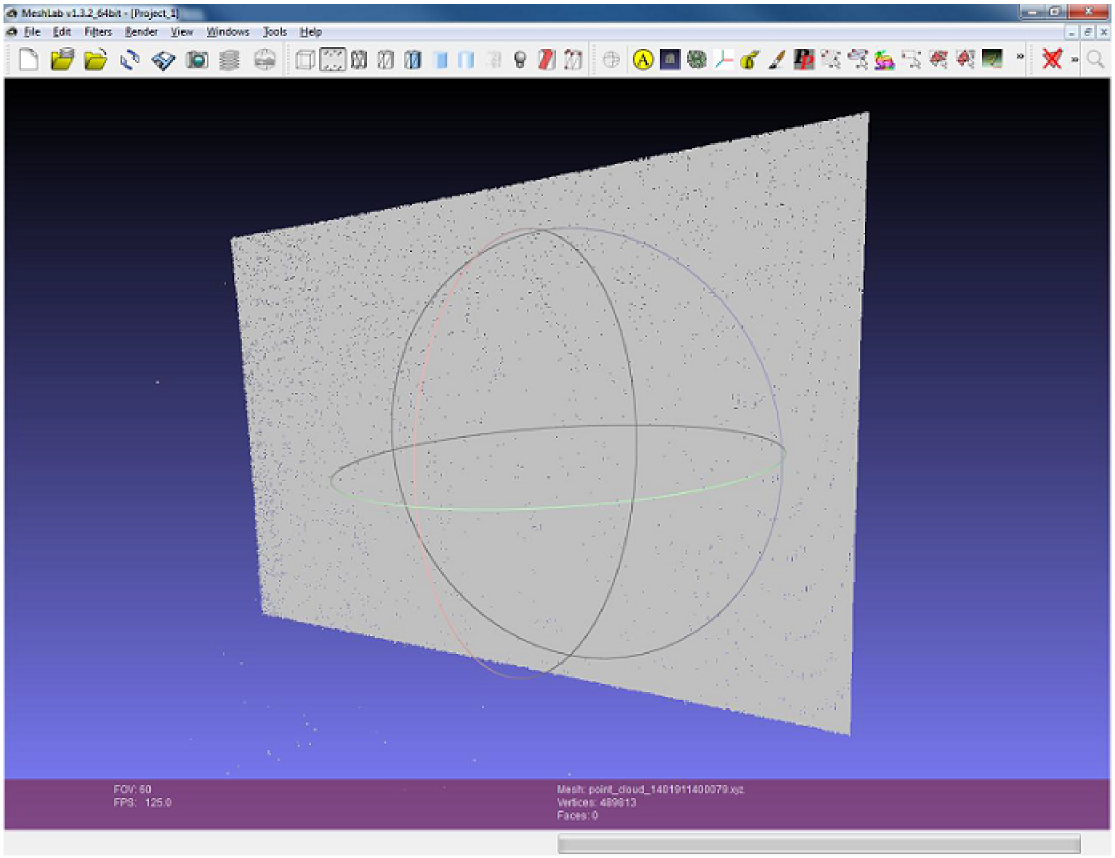


Figure 31. Point Cloud of Flat Surface

If the depth map is twisted or distorted around the edges, the calibration must be performed again, paying special attention to placing the printed calibration board close to the edges of the camera frame.

Figure 32 shows an example of an unacceptable point cloud.

NOTE: The 3D Machine Vision Reference Design is capable of very accurate measurements. If the flat surface being scanned contains a perceivable twist, verify that the surface is not twisted before performing the calibration routine again.

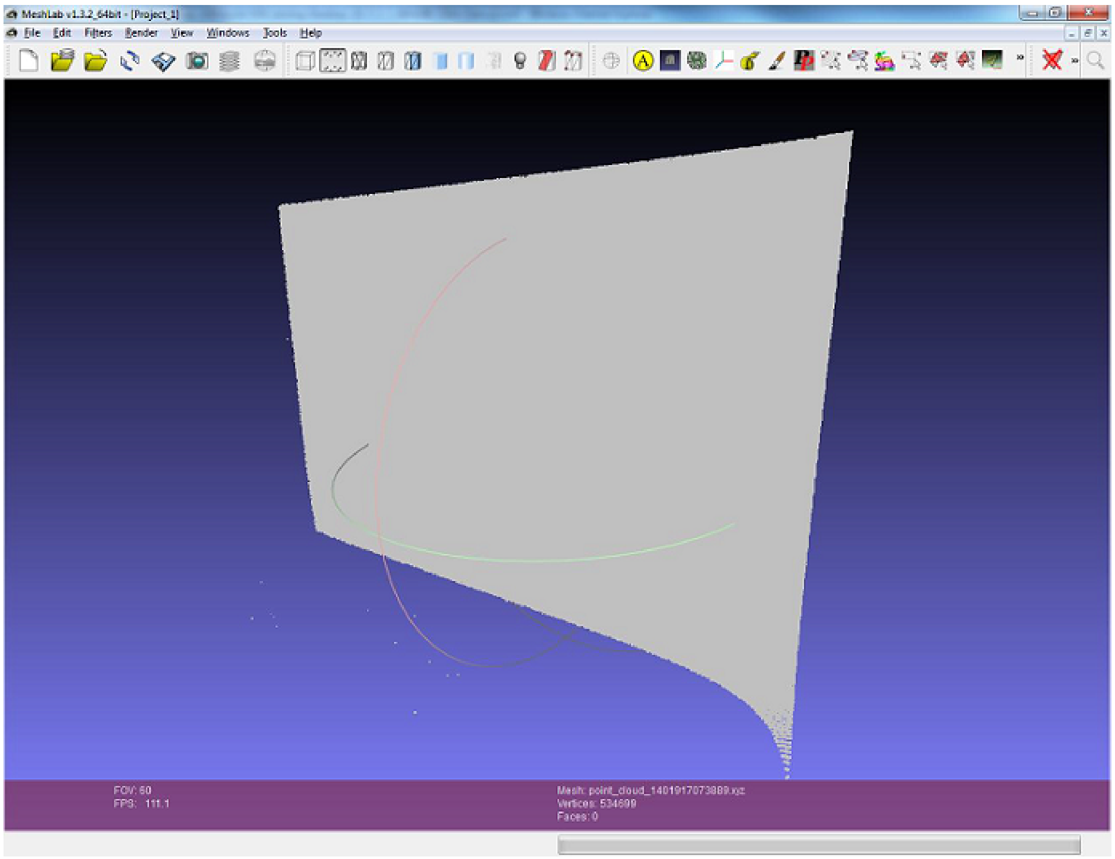


Figure 32. Point Cloud of Flat Surface Generated With Poor Calibration Data

2.5.7 Scan and Display 3D Object

With the system calibration complete and verified, scanning an object is done by placing the object of interest in the field of view of the camera and projector. Follow these steps:

1. Run the 3D Scanner executable file, and enter command 8: Perform scan (vertical and horizontal patterns), (see Figure 33).

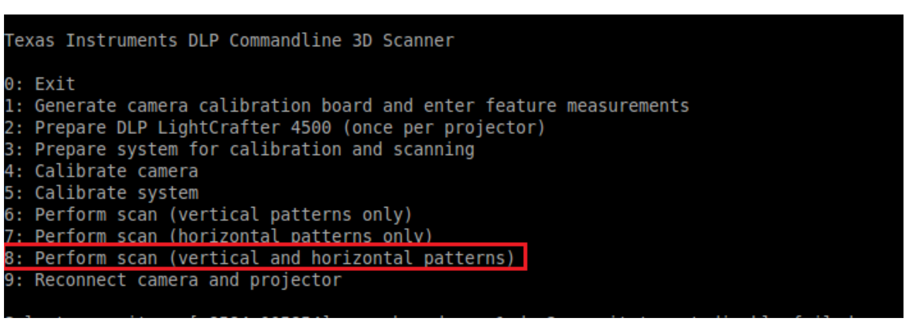


Figure 33. Menu Selection to Perform Scan

2. After entering command 8, the object to be scanned gets displayed on the monitor (see Figure 34 and Figure 35).

```
8
Loading camera and projector calibration data...
Constructing the camera and projector geometry...

Place the scanning target within the view of the camera and projector.
Press SPACE or ESC from window when ready to scan...
No such plugin for spec  "Auto"
init done
Using Wayland-EGL
wlpvr: PVR Services Initialised
```

Figure 34. Messages When Scanned Object Appears on Display

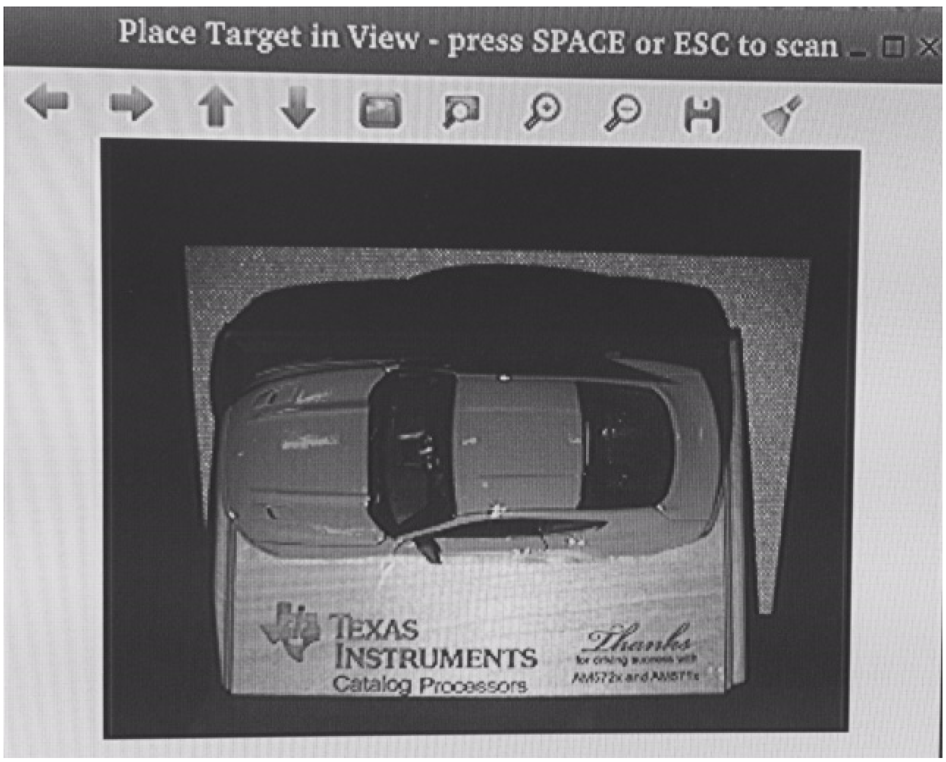


Figure 35. HD Display of Scanned Image

3. Adjust the camera settings so that the object to be scanned is neither too dark nor over exposed. Once the camera is set properly, place the mouse (connected to the AM57x board) cursor on the image and press the Esc or Enter button on the keyboard connected to the AM57x EVM.

4. After reading the instructions on the terminal, press ENTER (see Figure 36). The object scanning begins and the depth map image of the scanned object is displayed on the LCD/HDMI monitor connected to the AM57x EVM. The depth map image can be zoomed in and out by pressing the i or I (in) or o or O (out) key on the keyboard connected to the AM57x EVM. The output point cloud can be saved as an XYZ file by pressing s or S on the keyboard connected to the AM57x EVM. The XYZ file is saved in the ../output/scan_images/ directory. The XYZ images can be viewed offline using the MeshLab tool.

```
wlpvr: PVR Services DeInitialised

Point Cloud Viewer Operation:
i/I = Zoom in
o/O = Zoom out
s/S = Save point cloud xyz file
a/A = Auto-rotate the point cloud
c/C = Turn point cloud color on/off

NOTE: Press ESC key to quit the scan routine

Press ENTER after reading the above instructions...
```

Figure 36. Menu Selections for 3D Machine Vision Functions

2.5.8 Phase 2 Optimizations

There are two modifications added in this release:

- Overlap camera stream acquisition with processing, so acquisition and processing are not sequential anymore
- Headless and batch operation: command line options are added to allow operation without typing top-level commands
- Parts of decode pattern and generate point cloud operations are offloaded from A15 to DSPs (C66x) through OpenCL (this also relaxes A15 loading):
  – LineLine intersect
  – DecodeInverted

2.5.9 Headless Operation

In this version of software, it is possible to run in batch mode (for example: without keyboard attached, but only using ssh (terminal) session).

```
root@am57xx-evm:/usr/share/ti/dlp-sdk# ./DLP_LIGHTCRAFTER -h
Usage: ./DLP_LIGHTCRAFTER , command line options:
        -a         ... accumulate points in point cloud
        -s         ... skip saving acquired images (performance optimization)
        -b xyzd    ... sequence of commands, as sequence of digits, e.g. 380
        -d [1|0]   ... display ON (1) or OFF (0)
        -r scan_count     ... repeat count for continuous scanning
Example: ./DLP_LIGHTCRAFTER -d 0 -b 380 -r 15
```

NOTE: For repeated scanning operation, users must modify DLP_LightCrafter_3D_Scan_AM57xx_Config.txt and set the CONTINUOUS_SCANNING option to 1.

2.5.10 Troubleshooting

Typical problems may be related to either the AM57x EVM or the 3D scanner demo setup. For problems related to the 3D scanner setup, see Chapter 4 of TIDA-00254 for debug steps.

For problems related to the AM57x EVM, see the documentation in the Processor SDK at http://processors.wiki.ti.com/index.php/Processor_SDK_Linux_Software_Developer’s_Guide.

3 Test Data and Performance Benchmarks

Table 3 lists the processing times of Phase 1 software.

Table 3. Phase 1 System Processing Time

STEP	PROCESSING	CPU TIME (ms)	NOTES
1	Capture	920 ms	73 frames at 50 fps
2	File conversion	2200 ms	Majority of time spent in PG API fc2ConvertImageTo()
3	Pattern sorting	5 (550 to 600) ms	Dominated by writing acquired images to file system (can be skipped in production system)
4	Decoding patterns	380 to 450 ms	Time required for decoding a batch of 36 patterns (either horizontal or vertical) – depends on image content
5	3D point cloud generation	750 to 950 ms	Time depends on count of points in 3D cloud
–	Total processing time	4.2 to 4.5 (4.8 to 5.1) seconds	Longer time required if writing acquired images to the file system (step 3)

Table 3 lists the processing times of Phase 1 software.

Table 4. Phase 2 System Processing Time

STEP	PROCESSING	CPU TIME (ms)	NOTES
1	Capture	230 to 250 ms	73 frames at 50 fps. Extra time required as capture is overlapped with processing.
2	Image retrieval (from camera buffers)	300 to 320 ms	–
3	Pattern sorting	5 ms	Dominated by writing acquired images to file system (skipped here)
4	Decoding patterns	(360 to 390 ms) × 2	This is time required for decoding a batch of 36 patterns (either horizontal or vertical) – depends on image content. For both vertical and horizontal patterns, the total time is 700 to 800 ms.
5	3D point cloud generation	400 to 600 ms	Time depends on count of points in 3D cloud
–	Total processing time	1.7 to 2.0 s	Add 0.5 to 0.6 seconds if writing acquired images to the file system

3.1 *Future Improvement*

The Phase 1 development described in this version of the design guide is limited to ARM-based processing. Phase 2 and Phase 3, as in explained in Section 2 and Table 2, may be further developed to leverage available DSP processing capability and enabling tighter control of the synchronization. These developments may significantly reduce system processing time.

4 Design Files

4.1 Schematics

To download the schematics, see the design files at http://www.ti.com/tool/TIDEP0076

4.2 Bill of Materials

To download the bill of materials (BOM), see the design files at http://www.ti.com/tool/TIDEP0076

4.3 Assembly Drawings

To download the assembly drawings, see the design files at http://www.ti.com/tool/TIDEP0076

5 Software Files

To download the software files, see the design files at TIDEP0076.

Application binary and source code are available as a single download package at http://www.ti.com/tool/TIDEP0076.

Reference Section 2.2 and Section 2.4 for required hardware EVM board files, as well as the SDK file system required to run the demo.

6 Related Documentation

1. Texas Instruments, *Accurate Point Cloud Generation for 3D Machine Vision Applications Using DLP® Technology and Industrial Camera*, TIDA-00254 User's Guide (DLPU019)
2. Texas Instruments, *AM57x Software Development Kit: Processor SDK for AM57x Sitara™* http://www.ti.com/tool/PROCESSOR-SDK-AM57X
3. Texas Instruments, *Portable Point Cloud Generation for 3D Scanning using DLP® Technology Reference Design*, TIDA-00361 User's Guide (TIDU985)
4. MeshLab Blog, http://meshlab.sourceforge.net

6.1

6.1.1 Trademarks

Sitara, Lightcrafter are trademarks of Texas Instruments.
DLP is a registered trademark of Texas Instruments.

7 About the Authors

Dr. JIAN WANG is a Chip Architect with the Catalog Processor Group. He joined TI in 2006 as a Video Systems Engineer, working on the Davinci family of digital media processors. His recent roles focus on SoC system architecture for machine vision and next-generation industrial applications.

DJORDJE SENICIC is a Senior Software Engineer with the Catalog Processor Group. He joined TI in 2000 as an embedded software engineer, working on High Density Media Gateway C55x- and C66x-based solutions. Later he worked on multicore Video Codecs on C66x and DM81xx platforms, and also TI HPC K2H platforms. His recent roles focus on Processor Linux software package for Sitara an K2 devices, including machine vision applications.

MANISHA AGRAWAL is an Applications Engineer with the Catalog Processor Group. She joined TI in 2006 as a Software Engineer working on Video Codecs for the Davinci family of digital media processors. Her recent roles focus on growing and supporting Sitara line processors for multimedia, machine vision, and industrial applications.

MARK NADESKI is a Business Development Manager in TI's Processors Business Unit. He has worked at TI since 1992. His current focus is on machine vision.

Revision History

NOTE: Page numbers for previous revisions may differ from page numbers in the current version.

Changes from September 8, 2016 to August 24, 2017 **Page**

- Changed information in *Software Development Phases* 7
- Changed *Figure 4. Phase 1 Software Processing Data Flow* 7
- Added *Figure 5. Phase 2 Software Processing Data Flow* 7
- Changed to SDK version 4.04 9
- Changed to 04_00_00_04 9
- Added information to the instructions in *Obtaining and Installing Software Modules* 10
- Added *Phase 2 Optimzations* section 29
- Added *Headless Operation* section 29
- Changed *Phase 1 System Processing Time* table 30
- Added *Phase 2 System Processing Time* table 30