TDA4x（LDC）畸变矫正

Fredy Zhang; Joe Shen; Cary Zhang *EP FAE*

摘要

Jacinto™ 7 TDA4xx 是目前 TI 最新一代的汽车处理器，面向 ADAS 和自动驾驶车辆 (AV) 应用， 并且建立在 TI 在 ADAS 处理器市场十余年先进地位中积累的广泛市场知识基础之上。TDA4VM 以业界先进的功耗/性能比为传统和深度学习算法提供高性能计算，并具有很高的系统集成度，从而使支持集中式 ECU 或多种传感器的高级汽车平台实现可扩展性和更低的成本。

TI 最新一代的汽车处理器 TDA4x 集成了视觉预处理加速器 Vision Pre-processing Accelerator（VPAC）, 提供了常用的图像处理功能，比如颜色处理和增强、噪声过滤、宽动态（WDR）、镜头畸变矫正（LDC）、图像金字塔、图像缩放等。基于 TI 超过 20 年在数百万种产品中部署的多个 Soc 系统，TI 发布了最新一代的成像系统，即德州仪器第七代 ISP，并被集成在了 Jacinto7 TDA4VM 和 DRA829 处理器。

在汽车的应用中，鱼眼摄像头被广泛使用。因此，使用 VPAC 加速器内部 LDC 模块可以高效对镜头做畸变矫正。本文将详细介绍镜头畸变的过程。

修改记录

Version	Date	Author	Notes
1.0	May 2022	Fredy Zhang	First release

目录

图

1. VPAC 简介

Jacinto7 集成了视觉预处理加速器 Vision Pre-processing Accelerator（VPAC）, 提供了常用的图像处理功能，比如颜色处理和增强、噪声过滤、宽动态（WDR）、镜头畸变矫正、图像金字塔、图像缩放等。基于 TI 超过 20 年在数百万种产品中部署的多个 Soc 系统，TI 发布了最新一代的成像系统，即德州仪器第七代 ISP，并被集成在了 Jacinto7 TDA4VM 和 DRA829 处理器。

如图 1 VPAC 系统框图所示，由以下主要模块组成：

- Video Imaging Subsystem (VISS)
- Lens Distortion Correction (LDC)
- Bilinear Noise Filter (BNF)
- Multi-scaler (MSC)

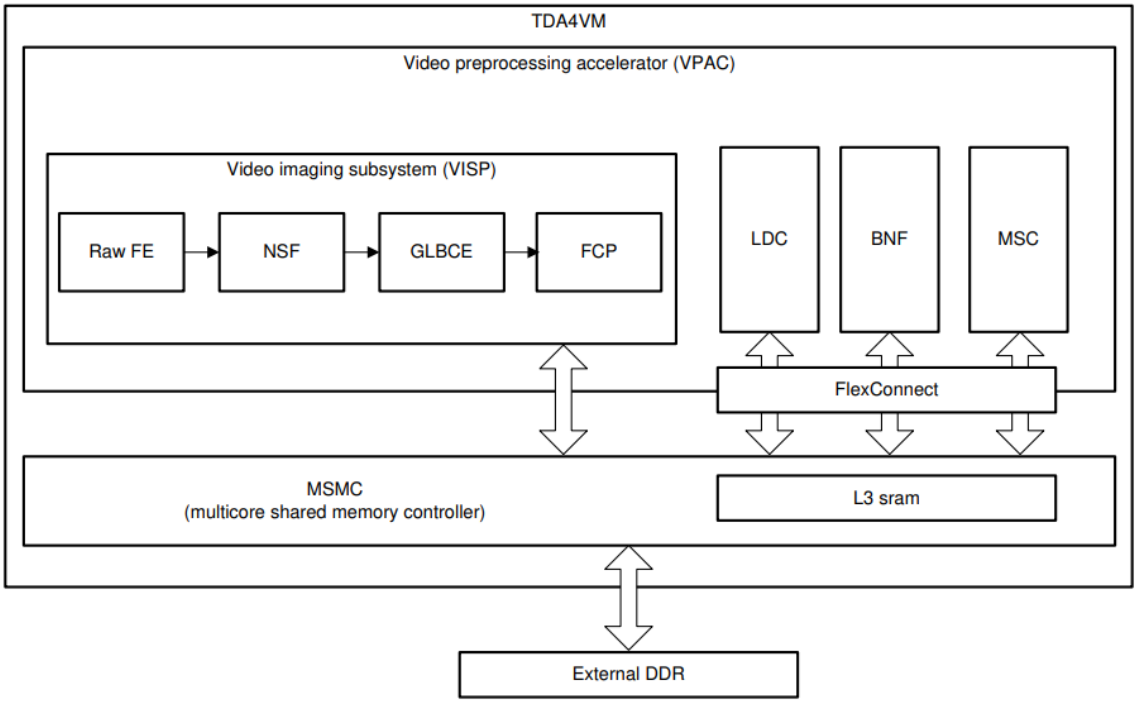

图 **1**. TDA4VM VPAC 系统框图

各个模块的主要功能如下：

- VISS（Video Imaging Subsystem）模块主要处理 RAW 数据并输出 YUV 图像，YUV 图像支持 YUV420 或者 YUV422 格式。其包含 RAW FE（Raw Front End）、NFS（Noise Filter）、GLBCE（Global and Local Brightness Contrast Enhancement）、FCP(Flex Color Processing)等模块。
- **LDC（Lens Distortion Correction）主要对图像进行畸变矫正。**
- MSC（Multi-Scalar）主要功能是对图像进行缩放。
- BNF（Bilateral Noise Filtering）双边滤波去除噪声。

后文将针对 LDC 模块的使用进行详细介绍。

2. LDC（畸变矫正）

2.1. LDC 简介

鱼眼镜头的拍摄角度大，成像角度宽，但是在进行图像采集和成像的过程中存在大量畸变，直接使用采集而来的图像很难满足实际需求，因此需要对图像进行畸变矫正。在 TDA4x 处理器，内部集成了 LDC 模块，我们可以方便使用 LDC 硬件模块对图像进行畸变矫正。

如图 2 所示，对于输入的图像，经过 LDC 后，鱼眼相机的图像得到了矫正。

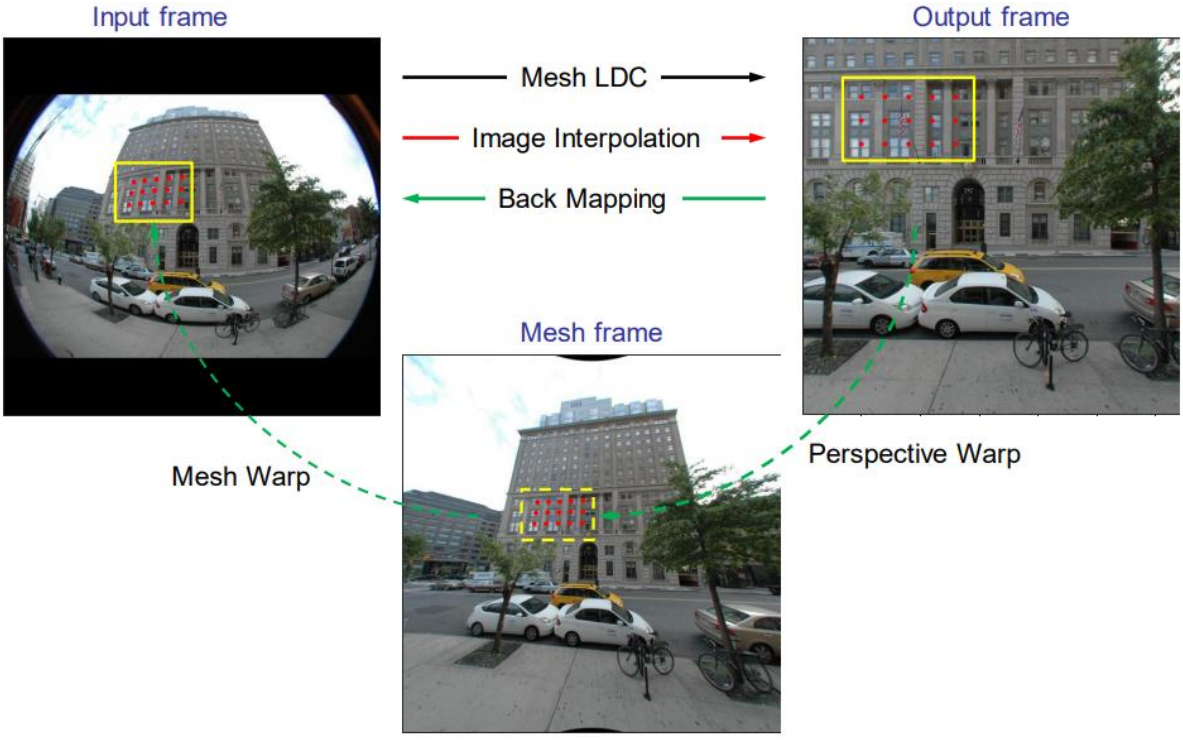

图 **2**. LDC Operations

格式支持：

- YUV420 12-bit packet input and YUV420 12-bit packed output
- YUV420 12-bit unpacked input and YUV420 12-bit unpacked output
- YUV420 8-bit packet input and YUV420 8-bit packed output
- YUV422 8-bit packed input and YUV420 8-bit packed output
- YUV422 8-bit packed input and YUV422 8-bit packed output

LDC 支持的功能如下：

- Autonomous memory-to-memory operation
  - Tile based processing
- Generic mesh based distortion model to correct multiple distortion types
- Perspective warp transformation for perspective correction; affine transform is a subset of perspective warp and supports scaling and rotation
- Support multiple input and output YUV data formats:

  - Among the supported data formats are UYVY (YCbCr 422), NV12 (YCbCr 420) and NV21 (YCbCr 420)
- Support up to 8192 x 8192 image dimension
- Pixel Interpolation
  - Bi-cubic interpolation for Y and bilinear interpolation for Cb/Cr
  - Bilinear interpolation mode to offer double throughput
- Performance (with small overhead)
  - 1 cyc/pixel (bilinear)
  - 2 cyc/pixel (Bi-cubic)
- Support to process either Luma only or Chroma only modes in YUV420 input data format to save bandwidth
- Support for independent block size in multiple regions (up to 9 regions)
- Dual output channels for programmable output pixel size
- ECC support on Mesh Data internal storage memories and width conversion LUTs on dual output channel
- Interface to HTS module for block level synchronization with other Ips
- Circular addressing support for Output Write (for SL2 interface) and Input Read
- KeyStone3 compliant infrastructure
  - 48-bit addressing on master interface
- Event for Block/Frame completion and Error scenarios

2.2. TDA4x LDC 步骤及环境

本节内容将介绍 TDA4x LDC 畸变矫正的环境和步骤。 下一小节将举例介绍 LDC 的实现过程。 畸变矫正所需要的环境如下：

1. DCC Tuning tool： DCC 工具里面的 Mesh LDC 插件是用来支持 LDC 畸变矫正的工具，因此需要在 PC 上安装该工具，该工具目前仅支持 Windows 版本； DCC Tools 的界面如下（工具的获取请联系 TI 技术支持）：

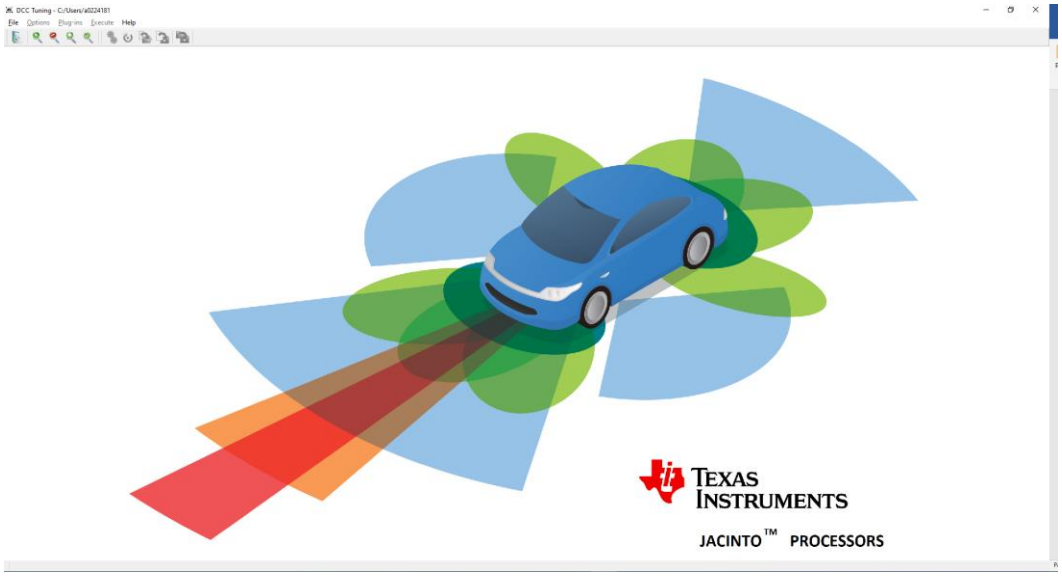

图 **3**. DCC Tools

2. 镜头的参数表格（联系镜头模块的供应商获取）；
3. Matlab （Windows）或 Octave（Linux/Ubuntu）环境用以执行 Mesh LUT 文件的转换。（Matlab 是 Windows 运行 Matlab 程序的环境，Octave 是 Linux 运行 Matlab 程序的环境，两个环境有一个就可以）；
4. 镜头的 YUV 图像。

TDA4x LDC 畸变矫正的流程如下：

1. 执行如下 Matlab 程序生成 Mesh LUT 文件（mesh.txt）（注意： 这里需要输入镜头的参数文件-spec_file, 该参考文件需要从镜头厂商获取，表格如链接）

```
function [] = gen_lut(spec_file, pitch_in_mm,f_in_mm, W, H, hc, vc,s ,m)
f = f_in_mm/pitch_in_mm ;
[h_p , v_p] = meshgrid( 0:W, 0:H);
[h_d,v_d] = xyz2distorted(h_p,v_p, f/s, hc, vc,spec_file, pitch_in_mm);
h_delta = round((h_d-h_p) * 8);
v_delta = round((v_d-v_p) * 8);
mh = h_delta(1:2^m:end, 1:2^m:end)';
mv = v_delta(1:2^m:end, 1:2^m:end)';
dlmwrite('mesh.txt', [mh(:), mv(:)],  'delimiter', ' ');

function [h_d, v_d] = xyz2distorted(x, y, z, hc, vc, spec_file, pitch_in_
mm)
[phi, r] = cart2pol(x-hc, y-vc);
theta = atan2(r, z);
lut = read_spec(spec_file, pitch_in_mm);
r = interp1(lut(:,1), lut(:,2), theta);
[h_d, v_d] = pol2cart(phi, r);
h_d = h_d + hc;
v_d = v_d + vc;


function lut = read_spec(spec_file, pitch_in_mm)
lut0 = dlmread(spec_file);
theta = lut0(:,1)/180*pi;
lut = [theta, lut0(:,2)/pitch_in_mm];
```

2. 使用 DCC 工具利用镜头的 YUV 图片和 mesh.txt 生成 xml 文件： 这里解释一下 XML 文件，如图 3 和 4 所示，DCC 工具输入文件采用 XML 格式。

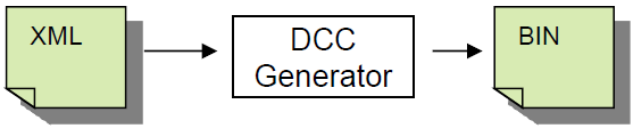

图 **4**. DCC Toolchain

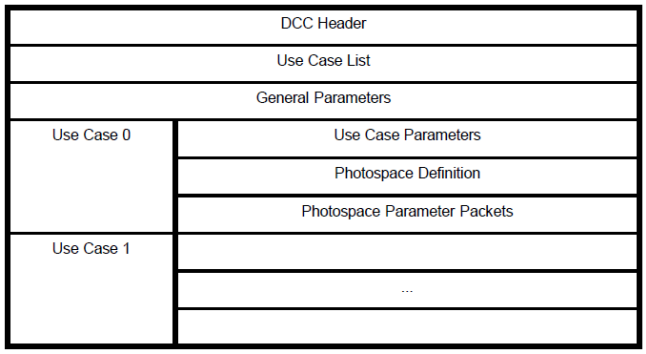

图 5. DCC File Structure

3. 替换 imaging 下面的 xml 文件
4. 在 imaging 目录下利用 generate_dcc.sh 生成 bin 文件

需要注意的是 xml 的镜头 ID 跟代码里面是一致的。DCC tools 里面包括了 Mesh LDC 模块。 这个模块的目的是生成 XML 文件供 SDK 使用。

XML: The exported .xml file name is sensor_name_mesh_ldc_dcc.xml

2.3. TDA4x LDC 实现

2.3.1. Mesh LUT 生成

在 Matlab 里面执行如下程序，生成 TDA4x LDC Mesh LUT:

```
    function [] = gen_lut(spec_file, pitch_in_mm,f_in_mm, W, H, hc, vc,s ,m)
    f = f_in_mm/pitch_in_mm ;
    [h_p , v_p] = meshgrid( 0:W, 0:H);
    [h_d,v_d] = xyz2distorted(h_p,v_p, f/s, hc, vc,spec_file, pitch_in_mm);
    h_delta = round((h_d-h_p) * 8);
    v_delta = round((v_d-v_p) * 8);
    mh = h_delta(1:2^m:end, 1:2^m:end)';
    mv = v_delta(1:2^m:end, 1:2^m:end)';
    dlmwrite('ldc_mesh.txt', [mh(:), mv(:)],  'delimiter', ' ');

    function [h_d, v_d] = xyz2distorted(x, y, z, hc, vc, spec_file, pitch_in_
mm)
    [phi, r] = cart2pol(x-hc, y-vc);
    theta = atan2(r, z);
    lut = read_spec(spec_file, pitch_in_mm);
    r = interp1(lut(:,1), lut(:,2), theta);
    [h_d, v_d] = pol2cart(phi, r);
    h_d = h_d + hc;
    v_d = v_d + vc;


    function lut = read_spec(spec_file, pitch_in_mm)
```

```
    lut0 = dlmread(spec_file);
    theta = lut0(:,1)/180*pi;
    lut = [theta, lut0(:,2)/pitch_in_mm];
```

注意这里的参数配置，输入文件是镜头参数 txt 文件：

```
        s = 2;
        m = 4;
        pitch_in_mm = 0.0028;
        f_in_mm = 0.85;
        W = 1280;
        H = 944;
        hc = W/2;
        vc = H/2;
        Wmesh = ceil(W / 2^m) * 2^m;
        Hmesh = ceil(H / 2^m) * 2^m;
        gen_lut("spec_file.txt", pitch_in_mm, f_in_mm, Wmesh, Hmesh, hc, vc,
s, m);
```

镜头的参数文件 spec_file.txt 格式如下：

```
    0    0
    0.1  0.001545358
    0.2  0.003090688
    0.3  0.004636121
    0.4  0.006181564
    0.5  0.007726863
    0.6  0.009272339
    0.7  0.010817678
    0.8  0.012363201
    0.9  0.013908753
    …
```

2.3.2. DCC Tools

将上面的“mesh.txt”连同来自相机的 LDC 输入 YUV 图像加载到 DCC 调整工具中，以生成 LDC 预览和 LDC 的 DCC xml 文件。

调整工具可以处理块大小和 填充细节（请按照帮助菜单中的 LDC 插件指南进行操作）。

1. 打开 DCC 工具，创建 Project，Mesh LDC 插件如图 1 所示，打开 Mesh LDC 插件。

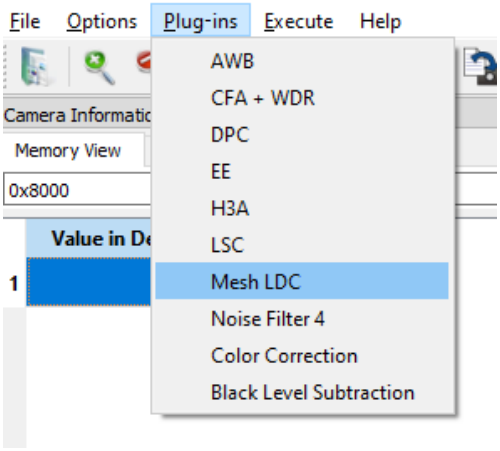

图 **6**. DCC LDC Plugin

2. 调整 Advanced params。

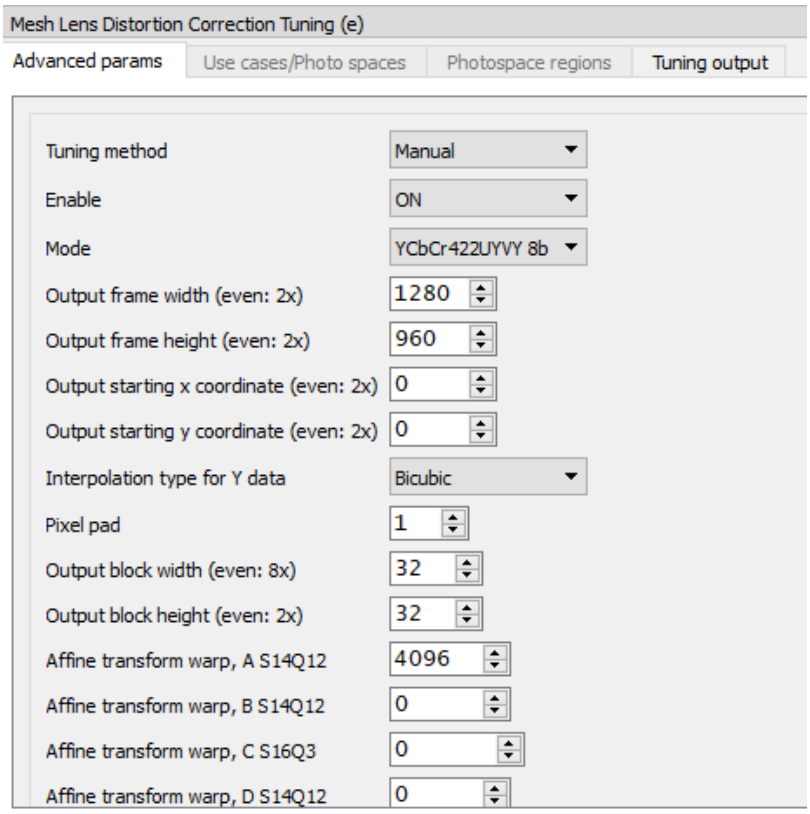

图 **7**. DCC LDC Plugin

3. 在“Input image” 输入栏输入 YUV 图片并点击 load。 在 LDC LUT file 输入栏输入生成的 mesh.txt 文件路径。

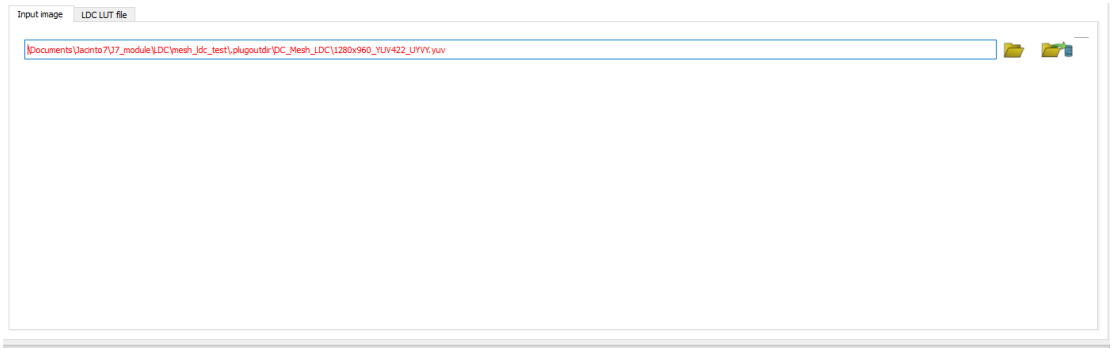

图 **8**. Input Image and LDC LUT file

4. 然后点击“Process Plugin”生成 output_image.yuv 可以预览窗口 Tuning output 进行预览。

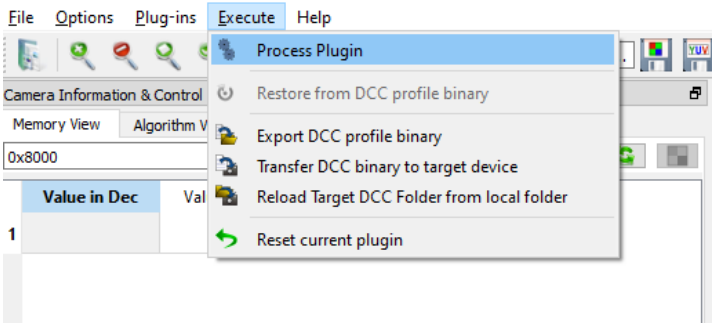

图 **9**. DCC Process Plugin

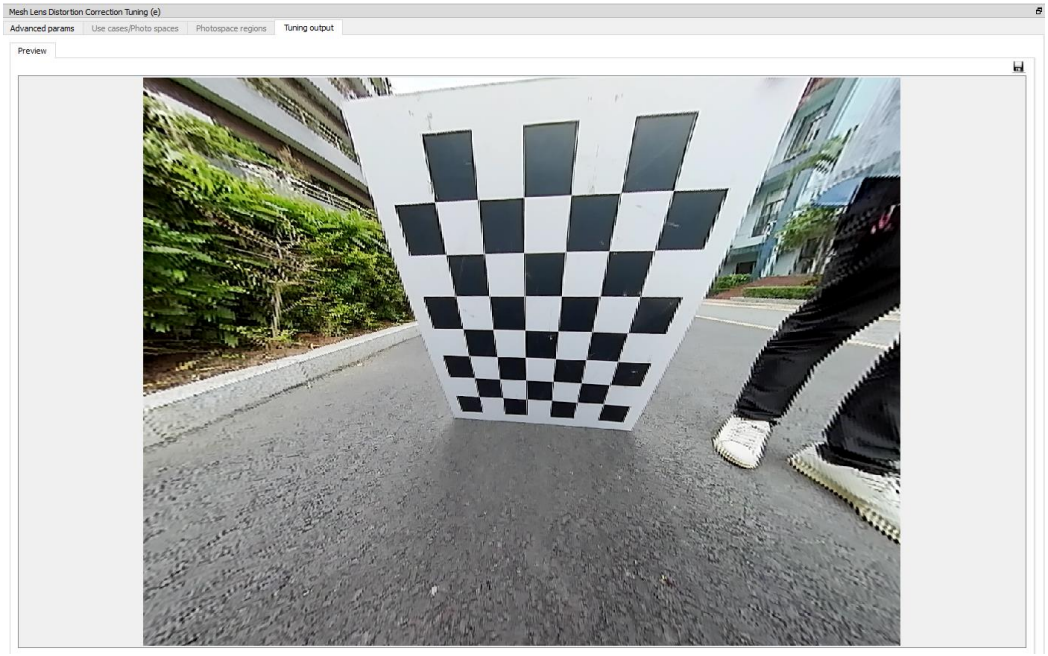

图 **10**. DCC Tuning Output

5. 最后， 点击 Export DCC profile binary, 将 Export 出的 xml（asamplesensor_mesh_ldc_dcc.xml）和 bin 文件（cid42_asamplesensor_mesh_ldc_dcc.bin）复制到 PSDKRA 中“imaging/sensor_drv/src/sensor_name/dcc_xmls/”文件夹中。

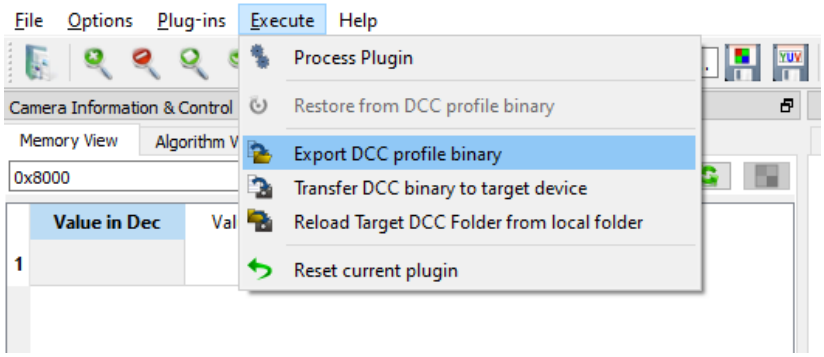

图 **11**. Export DCC profile binary

2.3.3. SDK 更新 LDC

在 Imaging 目录下, 需要运行该文件夹下的“generate_dcc.sh”并重新 clean build imaging 模块然后编译 vision app。PSDKRA 示例应用程序（例如“vx_app_single_cam”）将为 LDC 选择新的 DCC 设置。

3. 参考

1. https://software-dl.ti.com/jacinto7/esd/processor-sdk-rtos-jacinto7/08_02_00_05/exports/docs/vision_apps/docs/user_guide/group_apps_basic_demos_app_single_cam.html
2. https://www.ti.com/lit/zip/spruil1
3. https://www.ti.com/lit/pdf/spracx9

重要声明和免责声明

TI“按原样”提供技术和可靠性数据（包括数据表）、设计资源（包括参考设计）、应用或其他设计建议、网络工具、安全信息和其他资源，不保证没有瑕疵且不做出任何明示或暗示的担保，包括但不限于对适销性、某特定用途方面的适用性或不侵犯任何第三方知识产权的暗示担保。

这些资源可供使用 TI 产品进行设计的熟练开发人员使用。您将自行承担以下全部责任：(1) 针对您的应用选择合适的 TI 产品，(2) 设计、验证并测试您的应用，(3) 确保您的应用满足相应标准以及任何其他功能安全、信息安全、监管或其他要求。

这些资源如有变更，恕不另行通知。TI 授权您仅可将这些资源用于研发本资源所述的 TI 产品的应用。严禁对这些资源进行其他复制或展示。您无权使用任何其他 TI 知识产权或任何第三方知识产权。您应全额赔偿因在这些资源的使用中对 TI 及其代表造成的任何索赔、损害、成本、损失和债务，TI 对此概不负责。

TI 提供的产品受 TI 的销售条款或 ti.com 上其他适用条款/TI 产品随附的其他适用条款的约束。TI 提供这些资源并不会扩展或以其他方式更改 TI 针对 TI 产品发布的适用的担保或担保免责声明。

TI 反对并拒绝您可能提出的任何其他或不同的条款。

邮寄地址：Texas Instruments, Post Office Box 655303, Dallas, Texas 75265