

Subscribe to DeepL Pro to edit this document.  
Visit [www.DeepL.com/pro](https://www.deepl.com/pro?cta=edit-document)for more information.

* Update History

Created on 06/28/2021

06/30/2021 Addition of function

LCEPT Manual

* operating environment
  + OS: Raspberry Pi OS (Debian Linux)
  + Memory: 8GB or more recommended
* Required Software and Libraries
  + Pythonv3.7.3
  + gphoto2 v2.5.27
  + dcrawv9.28
  + ptpcam
  + OpenCV
  + Qt
* How to distribute the code
  + Share as a developer on GitHub (original can be changed with an updated history)
  + BOX or Dropbox sharing (copy of code managed by GitHub)
* Distribution Code Summary
  + common\_tools
    - setup.txt: Explanation of the installation procedure for the above required software, etc.
    - view\_img.py: View image and check pixel position.
    - get\_point\_data.py: Obtains the average value of RGB values in the range from the HDR synthesized RGB csv file, based on the information described in point.csv.
    - make\_luminance\_image.py: Create a pseudo-color image from a csv file of luminance distribution and display luminance values at arbitrary locations.
    - make\_luminane\_image\_m.py: Masks a binarized image to a csv file of luminance distribution, the resulting pseudo color image, and the luminance values at any location.
    - calc\_e.py: Conversion from csv file of luminance distribution to illuminance.
  + Measurement codes for each camera model
    - capture.sh: Set camera exposure and acquire images according to the information in evList.csv.
    - conv.sh: Convert the acquired RAW images to TIFF images and create XYZ csv files.
    - conv\_loop.sh: Execute conv.sh for all directories included in dirList.txt, which describes a list of directories where images are saved.
    - conv\_tiff.sh: Converts the acquired RAW image to a TIFF image.
    - conv\_xyz.py: Main code to convert from TIFF image to XYZ csv file.
    - conv\_hdr.py: Main code to convert from TIFF image to RGB csv file.
* System directory configuration (default configuration method)
  + Create a directory "LCEPT" directly under the home directory (/home/pi), and create the following directories under the directory to manage the code and the shooting data.
  + The black directories and files in the figure below are created when the code for the photo shoot is copied to the PC.
  + The directories and files in red in the figure below are created when a series of scenes (e.g., experiment types) are taken. The "scenario01" and "scenario02" directories are called "image parent directories.
  + The blue directories and files in the figure below are created by the system execution. The directory "yymmdd\_hhmmss" synchronized with the shooting date and time, which is created for each shooting, is called the "image child directory.
  + The codes for each model are stored under /LCEPT/tools/. However, even for the same model, if the lens attenuation characteristics differ due to different lenses, a separate directory is created in this hierarchy for management. (Because lens attenuation compensation is described in the code of conv\_xyz.py)



* How to use each code
  + In the following explanations, when a directory or file is specified as a command line argument, it is written as [directory name or file name]. The directory or file name should be entered as a full path (/home/pi/LCEPT/data/xxxxxxxx). In actual input, [ ] is not used.
  + capture.sh
    - Create an image parent directory to store the captured images, and place evList.csv, which contains the exposure information to be used for shooting, under the parent directory.
    - Move the current directory to the directory where the capture.sh file used for shooting is stored, and execute the following
    - $ sudo . /capture.sh [image parent directory].
    - Image child directories are created under the image parent directory, and the captured images are stored in these directories. The names of the image child directories are sequentially added to dirList.txt.
  + conv.sh
    - Move the current directory to the directory where conv.sh used for conversion is stored and execute the following.
    - $ sudo . /conv.sh [image child directory].
    - Based on picInfo.csv and sysInfo.csv in the image child directory, the conversion from RAW image to TIFF image and from RGB values to XYZ values is performed and the XYZ values are written out as a csv file.
  + conv\_loop.sh
    - Move the current directory to the directory where conv\_loop.sh is stored for conversion, and execute the following
    - $ sudo . /conv\_loop.sh [image parent directory].
    - conv.sh is executed for all image child directories listed in dirList.txt in the image parent directory.
  + conv\_tiff.sh
    - Move the current directory to the directory where conv\_tiff.sh is stored for conversion, and execute the following
    - $ sudo . /conv\_tiff.sh [image child directory].
    - RAW images in the image child directory are converted to TIFF images.
  + conv\_xyz.py
    - It is usually invoked and used within conv.sh and conv\_loop.sh.
    - The code creates an HDR image by combining TIFF images and converting them to XYZ values.
    - Various variables for conversion to XYZ values (signal range of RGB values used for HDR composition, conversion coefficients to XYZ, and lens correction) are described in sysInfo.csv, and the descriptions are changed as necessary.
    - Move the current directory to the directory where conv\_xyz.py is stored for conversion and execute the following
    - $ python3 conv\_xyz.py --input [image child directory].
    - Read picInfo.csv and sysInfo.csv in the image child directory and write out a csv file of XYZ values.
  + conv\_hdr.py
    - Move the current directory to the directory where conv\_hdr.py used for conversion is stored, and execute the following
    - $ python3 conv\_hdr.py --input [image child directory].
    - Read picInfo.csv in the image child directory and write out a csv file of RGB values after HDR composition.
  + view\_img.py
    - Move the current directory to the directory where view\_img.py is stored and execute the following.
    - $ python3 view\_img.py --input [image child directory/specific TIFF file].
  + get\_point\_data.py
    - Save a point.csv file in the image child directory, describing the pixel positions and ranges for which RGB values are to be obtained.
    - Move the current directory to the directory where get\_point\_data.py is stored and execute the following
    - $ python3 get\_point\_data.py --input [image child directory].
    - From the csv file of RGB values in the image child directory, average the RGB values for the locations listed in point.csv and write them as a file rgb\_vals.csv.
  + make\_luminance\_image.py
    - Move the current directory to the directory where make\_luminance\_image.py is stored and execute the following
    - $ python3 make\_luminance\_image.py --input [image child directory/dataY.csv].
    - A pseudo color image and a luminance histogram are displayed.
  + make\_luminance\_image\_m.py
    - Create a binarized image of the area to be masked for the luminance image and save it in the directory containing the luminance CSV file to be masked with the file name "mask.jpeg".
    - The mask image shall be an array of the same number of pixels in the TIFF image, with black (0, 0, 0) for the areas not designated as the calculation range of average luminance and white (255, 255, 255) for the areas designated as the calculation range.
    - Move the current directory to the directory where make\_luminance\_image\_m.py is stored and execute the following
    - $ python3 make\_luminance\_image\_m.py --input [image child directory].
    - A pseudo color image and a luminance histogram are displayed.
  + calc\_e.py
    - Move the current directory to the directory where calc\_e.py is stored and execute the following
    - $ python3 calc\_e.py --input [image child directory/dataY.csv].
    - Illuminance values are output.
* Calibration of the measurement system (how to write sysInfo.csv)
  + Derivation of RGB>XYZ transformation coefficients
    - Create an image parent directory in which to store images for calibration.
    - Save evList.csv to the image parent directory for use during calibration.
    - Prepare a color chart (Macbeth color checker, etc.) with known XYZ values (including actual measurement with CS100-A, etc.), and place the color chart in a lighting environment where the lighting intensity can be changed and is as uniform as possible.
    - Multiple levels of illumination intensity are set for the color chart, and the system captures images (capture.sh) so that the color chart is in the center of the image under each illumination intensity, and the XYZ values of the color chart under each illumination intensity are acquired (measured).
    - Develop (conv\_tiff.sh) the captured image.
    - Display the developed image (view\_img.py) and check the x-coordinate of the color chart center, the y-coordinate of the color chart center, and the width d from the center for each color chart in the captured image. Save the point.csv file containing these data in the image child directory.
    - Create a csv file (conv\_hdr.py) of the RGB values of the HDR image.
    - Obtain the average RGB values of the color chart locations (get\_point\_data.py) from the csv file of RGB values using the information in the point.csv file.
    - Multiple regression analysis is performed from the XYZ and RGB values to derive conversion coefficients.
    - The obtained conversion coefficients are reflected in the constant definitions in sysInfo.csv.
  + Confirmation of image rendering range
    - From the TIFF image taken and developed, check the width and height of the image and reflect them in the constant definitions in sysInfo.csv.
    - When using a fisheye lens, check the radius from the center, which corresponds to an angle of view of 180 degrees, and reflect it in the constant definition in sysInfo.csv.
    - To correct for lens attenuation from the center of the image to the edges of the image, edit directly where matLens is assigned a value in conv\_xyz.py code.
  + Setting the range of RGB values to be used for conversion
    - For images taken at each exposure amount, set the upper and lower limits of RGB values to be used for conversion to HDR images, and reflect them in the constant definitions in sysInfo.csv.
    - Since TIFF images are developed with gamma 1, in principle they are valid for conversion to HDR images as long as the RGB values are not 0 or 255, but lower and upper limits are set to account for noise effects.
* Measurement Procedure
  + Create a directory (image parent directory) under /home/pi/LCEPT/data (for example, scenario01) to store the captured images, etc. In the created directory, create a configuration file for the measurement system (sysInfo.csv) and an exposure level configuration file ( Save the evList.csv file.
  + Move the current directory to the directory where the bash files (e.g., capture.sh) of the system to be used for measurement are stored, and execute capture.sh.
  + An image child directory (the directory name at the time when capture.sh is executed) is automatically created and the captured images are saved. A copy of picInfo.csv and sysInfo.csv, which describe the image file name and exposure information, are also saved in the image child directory.
  + The development and conversion of the captured raw image to XYZ values is done in conv.sh or conv\_loop.sh.
  + The luminance and illuminance values are checked by using make\_luminance\_image.py and calc\_e.py from the CSV file of the output luminance distribution.