Assignment Projection Plane with Live Video

HL

1. The objectives of this assignment is to display
(1.1) digital photo on to a virtual projection plane, and
(1.2) digital video on to the virtual projection plane.

2. To accomplish the above objectives, I have posted the following programs to assist you step by step to realize the objective. These reference programs can be found at
https://github.com/hualili/opencv/tree/master/ComputerGraphics_AR/F2018

(2.1) line-display-2019-11-6.cpp, an OpenCV program to read and display images, use cmake and make to create binary executables, cmake file is listed below:

```
cmake_minimum_required(VERSION 2.8)
project(   main )
find_package( OpenCV REQUIRED )
include_directories( ${OpenCV_INCLUDE_DIRS} )
add_executable( main line-display-2019-11-6.cpp )
target_link_libraries( main ${OpenCV_LIBS} )
```

Figure 1. CMakeLists.txt file.

Use the following commands to create binary executable:

```
$cmake .
```

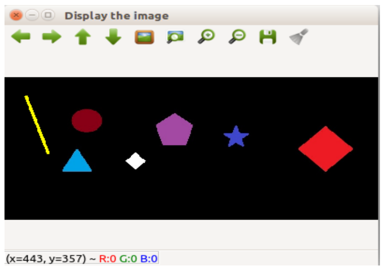
Once it is done, use

```
$make
```

to run the program:

```
./main 1image.png
```

where the binary executable is main and the input image is 1image.png. The result is shown belwo.



(x=443, y=357) ~ R:0 G:0 B:0

Figure 2. Result from line-display-2019-11-6.cpp the line is in yellow colour and it is shown on the image.

(2.2) xy-display-2019-11-6.cpp program to draw x-y coordinate system. Modify the cmakeLists.txt to replace the previous cpp program with this module, then go through 2 steps (cmake and make) to build binary executable. Then executable this program, you will have x-y coordinate system drawn on the screen on top of the image.

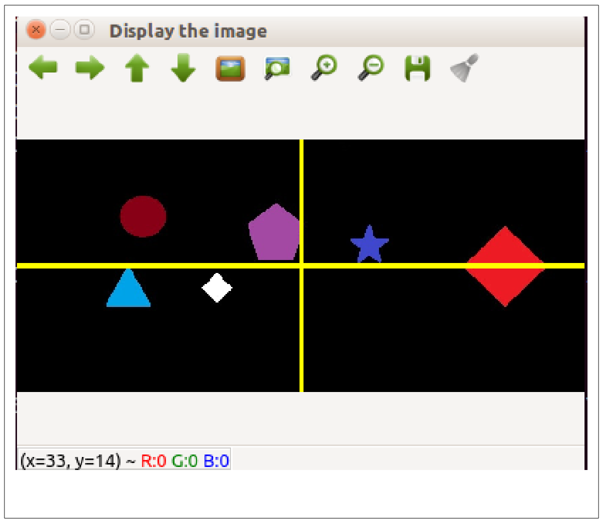


Figure 3. The result from xy-display-2019-11-6.cpp This program serves the purpose to establish virtual display coordinate system.

(3) world-viewer-openCV-2019-11-6.cpp, this program establish world-coordinate system display. You can build this program based on the above mentioned steps, the result is shown in the following figure.

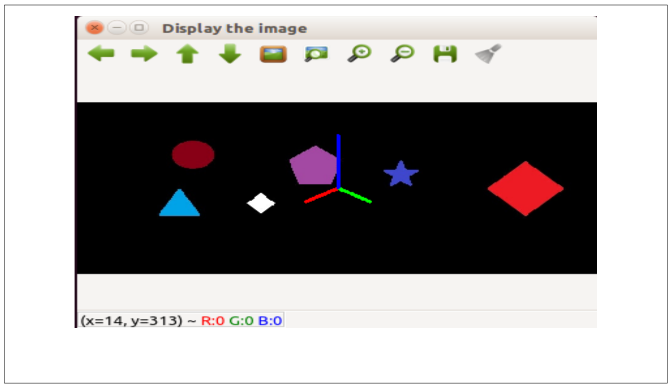


Figure 4. The result from world-viewer-openCV-2019-11-6.cpp program. Note OpenCV color is defined as b,g,r.

(4) projection-openCV-2019-11-6.cpp display the projection plane for testing purpose. Once you follow the same step to compile and build the program, it looks like the one shown below.

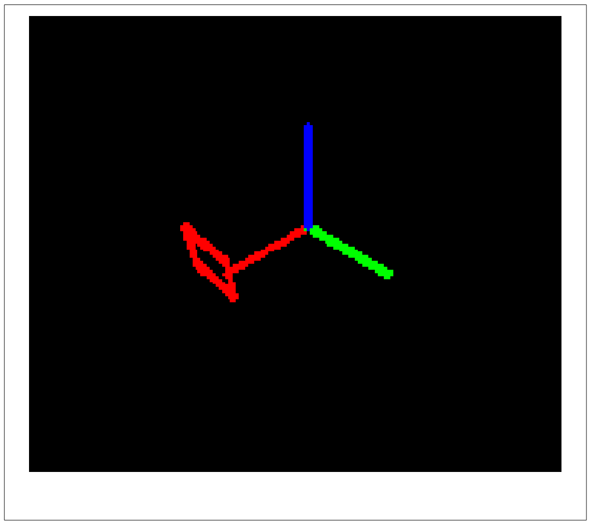


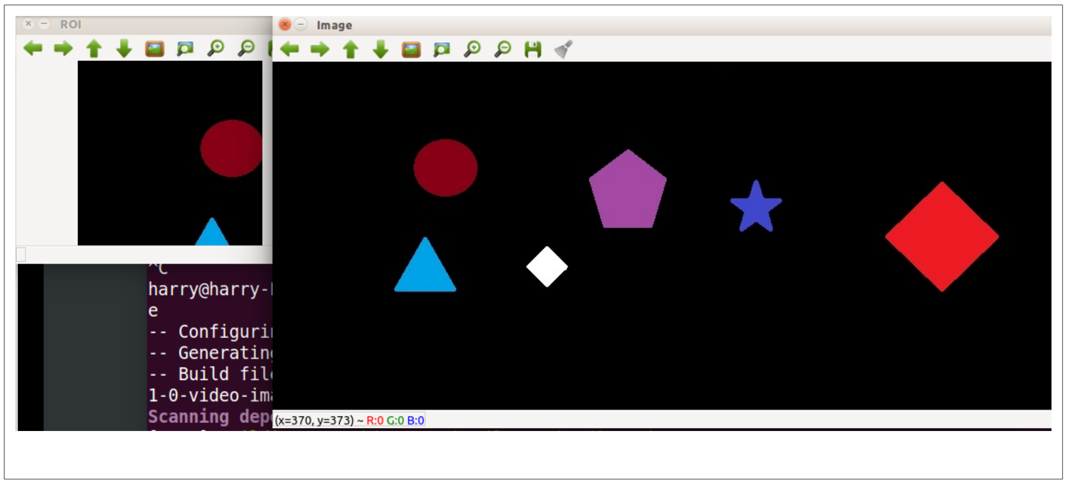
Figure 5. projection-video.cpp, this sample code display projection plane.

3. Now use opencv to access image pixels.

(3.1) roi-opencv.cpp program defines image region of interests and access to the roi. Once you build and run this program, you will use the following command:

```
$./roi 1image.png
```

Once the program is executed, then you will have the following result.



```
^C
harry@harry-
e
-- Configuri
-- Generatin
-- Build fil
1-0-video-im
Scanning dep
```

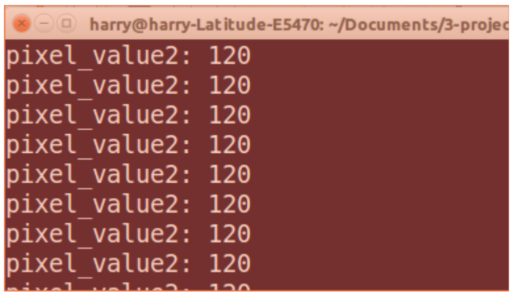
Figure 6. roi-opencv.cpp program defines and reads roi from an image. The roi is on the left.

Once you create the projection plane and now let's work on reading each individual image pixel.

(3.2) pixel-opencv.cpp program reads image pixels from the image. Once you build and run this program, you will use the following command:

```
$./pixel.cpp 1image.png
```

Once the program is executed, then you will see the pixel values printed to the console.



```
pixel_value2: 120
pixel_value2: 120
pixel_value2: 120
pixel_value2: 120
pixel_value2: 120
pixel_value2: 120
pixel_value2: 120
pixel_value2: 120
```

Figure 7. Reading pixel values from image.

4. Use the linear decoration algorithm, to read image pixels one by one, and then project the pixel onto the projection plane.

5. Submission of your work:
(5.1) one digital image of your choice;
(5.2) screen capture of the projected image on to the projection plane in the world coordinate system;
(5.3) in class demo of video projected onto the projection plane;
(5.4) source code.

(END)