STEREO VISION APPLYING OPENCV AND RASPBERRY PI

G. Pomaska
University of Applied Sciences Bielefeld, Germany, gp@imagefact.de

Commission II

KEY WORDS: Stereo Vision, OpenCV, Python, Raspberry Pi, Camera Module, Infrared Photography, Image Processing

ABSTRACT:

This article points out the single board computer Raspberry Pi and the related camera modules for image acquisition. Particular attention is directed to stereoscopic image recording and post processing software applying OpenCV. A design of a camera network is created and applied to a field application. The OpenCV computer vision library and its Python binding provides some script samples to encourage users developing their own custom tailored scripts. Stereoscopic recording is intended for extended base lines without a mechanical bar. Image series will be taken in order to wipe out moving objects from the frames. And finally the NoIR camera made infrared photography possible with low effort. Computer, accupack and lens board are assembled in a 3D printed housing operated by a mobile device.

1. INTRODUCTION

1.1 Stereo Vision

Visualizing and extracting spatial information from digital images, taken from two vantage points, referred to as stereo vision. Comparing the relative positions of objects in the frames enables extracting of 3D information as well as in the biological process. The normal case of stereo vision arranges two cameras horizontally within in a base distance, pointing in the same direction. This arrangement results in two different perspectives. A human may compare the half-pictures of the stereogram in a stereoscopic device. A computer applies algorithms to automatically match corresponding points and store the depth information in a disparity map. Initially the working chain starts from the camera calibration, followed by image refinement and stereoscopic image rectification. The stereoscopic window should be adjusted for convenient viewing. A disparity map may be converted into a point cloud, represented by 3D coordinates with a known scale or used for presentation of spatial scenes in one image by mouse movement or screen rotation on mobile devices.

1.2 OpenCV and Python

The application programmer has access to the necessary algorithms by OpenCV an API for solving computer vision problems. OpenCV incorporates methods for acquiring, processing and analyzing image data from real scenes. Interfaces to languages as C++, Java and Python are implemented on different operating systems. Primary the OpenCV library was developed since 1999 by Intel Russia. The cross-platform Software is now available under the BSD license, free for academic and commercial use. Since 2012 OpenCV is under continuous development of the non-profit organization OpenCV.org. For beginning programmers OpenCV Python binding on a Raspberry Pi is placed as a suitable tool introducing image processing and machine vision.

2. RASPBERRY PI HARDWARE

2.1 Raspberry Pi Boards

Without going to much into detail some remarks about the computer hardware are listed here. The single board computer Raspberry Pi was original invented to promote computer science in schools and education. Today one can find this little computer in industrial applications as well. By now the number of sold items is about 20 millions. Current devices are the Version 3 followed by Version 4 (since June, 2019), the Zero and the compute modules. A compute module comes as a plug-in-board with connection pins only on the board itself. A Raspberry Pi 3B with 1,6 GHz frequency, 1 GB RAM, WLAN and common interfaces has a street price of around 35 Euro. The officially supported OS is a free Raspian Linux. A micro SD-card is used as mass storage. Due to the compactness and the integrated WLAN adapter the camera stuff presented in this paper is driven by the RPi Zero W.

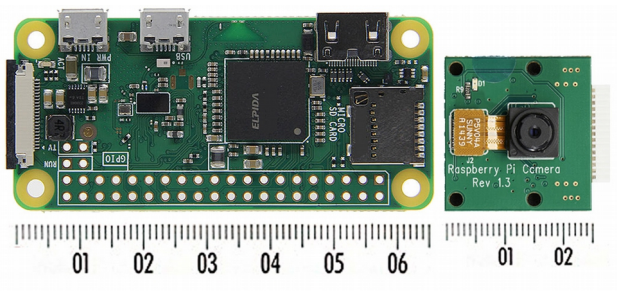

Figure 1. Raspberry Pi Zero W and camera module V1.3

2.2 RPi Camera Modules

The Raspberry Pi foundation provides related camera modules for image acquisition. Successor of the OmniVision Sensor V 1.3 with 5 MP (2592 x 1944 pix) image resolution is version 2, a Sony IMX219 sensor with an extended image resolution of 8 MP (3280 x 2464 pix). Night vision versions without infrared filters are even available. The board sizes only 25 x 20 mm, shortest focus distance is named at approx. 80 cm. M12 mounts

can be fixed for use of different lenses. Figure 2 displays the direct view on the sensor without a lens and the M12 mount for carrying changeable lenses. Beside the original dismounted lens, a tele lens and a 120 degree wide angle lens are depicted. Camera and computer are connected by a ribbon cable plugged into the CSI interface. The CSI port on the Zero is smaller therefore the cable differs from the standard shape. ZeroCam NoIR is a special camera modification for the Zero.

Shell commands raspistill, raspiyuv and raspivid, raspividyuv can drive the camera for still photography and video recording. The yuv extension don't use an encoder and writes directly to the disk. Camera format is 4:3 or 16:9 according to the selected mode. The camera produces previews only on directly connected HDMI displays. Small displays beginning from 3.5 inch with low resolutions are offered for monitoring the camera image. A command for testing the camera may use the following options:

```
raspstill -k -t 0 -vf -hf
```

K switches to keyboard mode, t defines the preview time in ms, 0 is infinitely, vf and hf flip the image vertically and horizontally. One can quit the mode by pressing the x key.

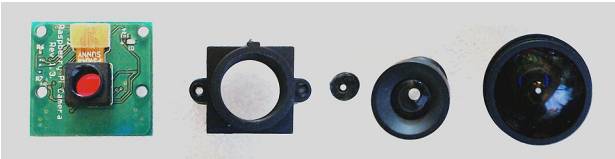

Figure 2: Lens modification accessories

3. IMAGE ACQUISITION

3.1 Camera Settings

The PiCamera class is a Python API for driving the camera via software commands. Focusing on stereo vision two cameras should fire exactly at the same time. The characteristics of a rolling shutter system do not meet a high demand on synchronization. Since we consider here a very long base line and series for image stacking the latency is a minor problem.

If a sync time within 1/1000 s is required, the crowd funding project StereoPi will comply. StereoPi is an interface board with two CSI sockets connected to a RPi compute module. The end user benefits from the application software SLP (StereoPi Live stream Playground), distributed as a Raspian Linux image file. For more details about StereoPi visit the projects web site *http://stereopi.com.*

As applied in time laps photography images should display equivalent lightning conditions in brightness, saturation and color because of intending the later stacking process. A recipe for capturing consistent images runs as follows: Fix the shutter speed, define the ISO value, set analog and digital gain to fixed values, switch exposure mode and white balance off and set fixed values for automatic white balance gains. First let the camera warm up 2 seconds and wait for automatic control. A python script that takes n images within a time delay reads as follows:

```
from picamera import PiCamera
import time
import cv2
#
BASEDIR = 'photo/'
n       =  7 # number of pictures
delay   = 10 # time between pictures
def take_picture(cam,file):
    cam.capture(file)
```

```
def viewPicture(file):
    view = cv2.resize(cv2.imread(file),\
    (320,240))
    cv2.imshow(file,view)
    cv2.waitKey(0)


def main():
    cam = PiCamera(resolution=(2592,1944),\
    framerate = 15)
    cam.iso           = 200
    time.sleep(2)#wait automatic control
    cam.shutter_speed = cam.exposure_speed
    cam.exposure_mode = 'off'
    g = cam.awb_gains
    cam.awb_mode      = 'off'
    cam.awb_gains     = g
    for i in range (n): # take n picture
        imName = BASEDIR + '%02d.jpg' %i
        take_picture  (cam,imName)
        viewPicture(imName)
        time.sleep(delay)
    cv2.destroyAllWindows()
main()
```

3.2 A Camera Network

A basic network configuration for stereoscopic imaging is given as illustrated in figure 3. A mobile router handles the fix IP addresses of two RPis and connects to a mobile computer named host here. Additional hardware in this configuration is a HDMI display and a small thermal printer. The host computer takes use of a Web interface for further image processing. Image files are named by a random code. Catching the according QR codes with a smartphone enables direct download. Random codes are stored and printed for later downloading from the internet.

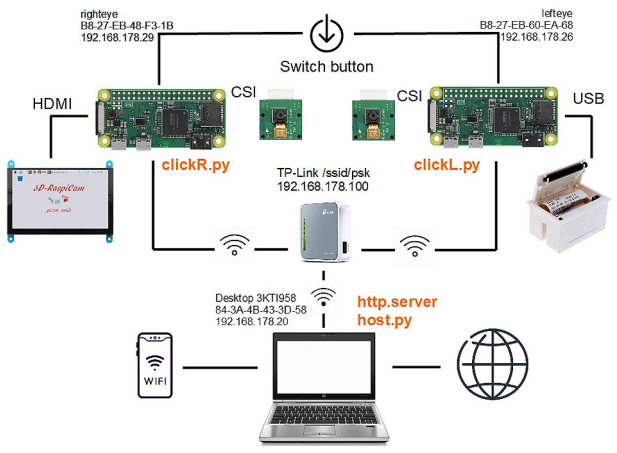

Figure 3: Camera network for taking stereograms

Virtually sync of the cameras can be implemented by GPIO request or wireless by SSH secure shell protocol or TCP socket programming. Talking about socket programming is outside the scope of this paper. Sending a command to a remote computer via SSH is in python a simple call of a sub process:

```
import subprocess
cmdLine ='ssh pi@IPremoteComputer \
raspistill -o image.jpg'
subprocess.run(cmdLine)
```

To avoid password authentification the sshpass tool should be added to the command like

```
cmdLine = 'sshpass -p passwordRemoteComputer ssh ... as above …
```

4. IMAGE PROCESSING APPLYING OPENCV

Due to the less stronger computing power of the RPi Zero, the following procedures are processed on the so called host computer. The host is a mobile device like a tablet or notebook or may be an RPi 4. Whereas the camera calibration is a pre-process, all the other functions follow as post-production.

4.1 Camera Calibration

The OpenCV camera calibration runs as a standard procedure carried out by taking pictures of a regular point pattern or a checker board. Points or intersections at the checker board are detected in the images and stored in the array of image coordinates, while the according object coordinates are given by the pattern and a constant z-Value. Comparing the measurements with the object points provides the camera matrix and lens distortion for the camera or in case of stereo-rig calibration for both cameras. Additional the relative position between the vantage points are given through the rotation matrix and the translation vector. Following the calibration image refinement can be executed by transferring image coordinates about the principle point deviation and lens distortion removal. After this step the images are presented as they were taken by a pinhole camera.

Figure 4: Calibrating a camera rig with OpenCV

4.2 Image Pre-processing

4.2.1 Erase moving objects

Both cameras are calibrated and provide upon request refined images due to the known camera matrix and distortion coefficients from a former calibration. Grabbing the images from the cameras in a predefined data structure is the first step in the workflow before the stacking process follows.

It is necessary to wipe out moving objects from multiple images taken from a fixed camera position. We do it by calculating the median of the images as noted in the script. The code gives some more information about the power of Python. The directory *server/* is transferred to the function. Applying the globe module lists all PNG-files. Looping through the list by reading the images into a numpy array and calling the methods stack and median enables saving the background extracted frame.

```
def stackServer(target='server/'):
    photos = glob.glob(target + '*.png')
    img=[]
    for filename in photos:
    img.append(np.array(Image.open\
    (filename)))
    sequence = np.stack(img,axis=3)
    result = np.median(sequence,axis=3).\
    astype(np.uint8)
    Image.fromarray(result).save\
    ('serverStack.png')
    return
```

From now on we continue processing only with two images, the left and right image of the stereogram.

4.3 Histogram Equalization

Pre-processing for balancing images is known as histogram equalization. It will make dark images less dark and bright images less bright. OpenCV provides the contrast limited adaptive histogram equalization method applied to a CLAHE object. A color image must be first transformed into the YUV or LAB format. The channels must be split and the L-channel is processed by the apply method. Subsequent the channels have to be merged back and the conversion from LAB to BGR finishes the procedure. Even in landscape shootings is a significant refinement obtainable. The following code snippet applied to an infrared image (figure 6) demonstrates the improvement.

```
def histogramAdjust(imgFile):
    img1     = cv2.imread(imgFile)
    img1_lab = cv2.cvtColor(img1,\
    cv2.COLOR_BGR2LAB)
    clahe = cv2.createCLAHE(clipLimit=2.0,
    tileGridSize=(8,8))
    img1_lab[:,:,0] = clahe.apply \
    (img1_lab[:,:,0])
    imgRes          =
    cv2.cvtColor(img1_yuv,\2
    cv2.COLOR_LAB2BGR)
    cv2.imwrite('clahe' +imgFile, imgRes)
    return
```

Figure 5: Histogram equalization applying the CLAHE method

4.4 Stereo Image Rectification

With regard to the geometric conditions an image pair should have the same image content with minimum hidden parts and a perfect alignment is requested as well. It is necessary to perform camera calibration respectively stereo-rig calibration in advance. The process used to project images onto a common

image plane is named image rectification. It differs between the calibrated and uncalibrated case. The first case takes as input values the calibration data, the latter needs corresponding points in the images. The characteristic of rectified images are that all corresponding lines are parallel to the horizontal axis and that they have the same vertical values. Finally vertical parallaxes are eliminated. Figure 6 illustrates the situation of a free hand taken image pair. Key points are detected by the ORB operator and matched with the Brute-Fore Matcher. Both algorithms are made available again by OpenCV. The corresponding points depict different values in the y-coordinates. The paralle view image pair underneath in figure 6 is a rectified version and doesn't show vertical parallaxes anymore. Colours come frome saturation extension. The uncalibrated rectification algorithm used here is taken from (Spizhevoy, Rybnikov, 2018) and assumes calibrated images recorded from good positioned cameras.

Figure 6: Stereo image rectification, the uncalibrated case

4.5 Stereo Formatting

A stereoscopic image pair has to be converted into a stereoscopic format for viewing: side-by-side, cross, or anaglyph are the alternatives. ImageMagick is a professional tool for editing or composing bitmap images. If it is installed on the machine a system command can be called like

```
cmdLine='magick montage -mode concatenate \
leftImage rightImage targetFile'
subprocess.run (cmdLine)
```

The combination of left and right image is stored in targetFile, figure 10 and 11 are samples of ImageMagicks montage command.

Setting the stereoscopic window, the plane that comes with zero parallax, and cropping the image should be outside the scope of this paper. Finally we calculate the depth map by instantiate a stereoSGBM_create object and calculate it with the compute method taking again benefit oft the OpenCV library.

5. FROM CONCEPT TO APPLICATION

Corresponding with the above delineated network configuration a hyper base stereoscopic application for taking photographs on side should be developed. Characteristics of the procedure are the extreme long base line without a physical base and subtracting moving objects from the images. That requires picture taking of sequences followed by a stacking procedure. To ensure equal lightning conditions exposure is used as described in chapter 3.1.

5.1 Camera for Field Usage

Figure 7 displays the CAD construction and the 3D printed parts of a camera housing. The lens board carries two cameras. One is equipped with a RGB sensor and M12 mount for changing lenses from tele angle to wide angle. The other is the NoIR camera equipped with a step up ring to change the infrared filters with different wavelength protection. The left camera (called Backbord) will act as the server, the right camera (called Steuerbord) as a client. The server is used for pointing and delivering pictures as requested in intervals by the client. The computer is powered via USB by the battery from inside the carrier. The camera platform has to be adjusted in the first step by a bubble level. Pointing 90 degree to the baseline is supported by software. First the both cameras point to the opposite camera each other. Afterwards rotation about 90 degrees targets the object. This principle follows the well known procedure from former photo theodolites. Accuracy of the adjustment depends on the available hardware and should comply to the uncalibrated rectification case. By the way, our camera calibration yield to a focal length of 2.596 pixel at 3280x2464 resolution that gives a horizontal field of view of about 65 degree.

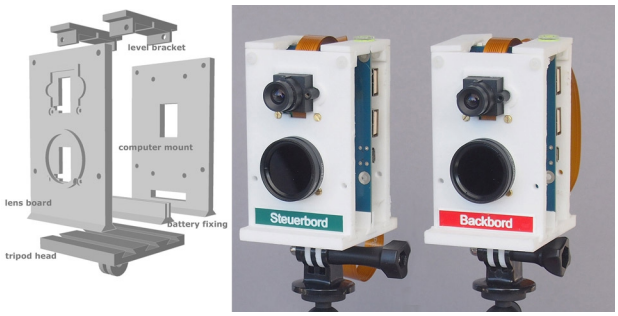

Figure 7: Camera housing design fabricated by a 3D printer

5.2 User Interface

Since there are any mechanical operating switching missing, the complete photography runs under software control. We are talking about headless systems. Booting includes running the VNC server and starting the application software. We are connected with fixed IP addresses to a network. There is no keyboard input intended. Starting the VNC viewer on a mobile device gives access to a GUI interface. If necessary, the camera settings may be predefined by a separate panel, that is used for both cams.

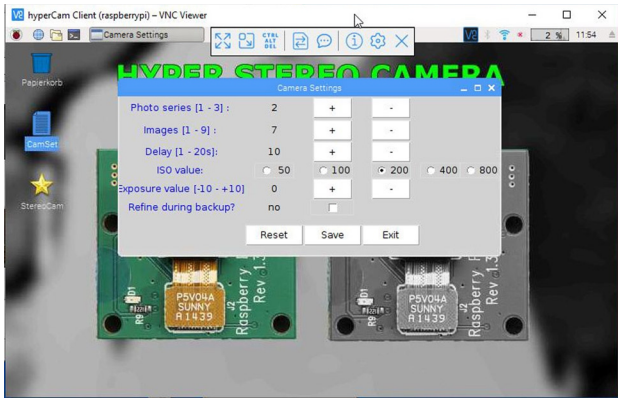

Figure 8: VNC viewer window showing the camera settings panel

The server script enables pointing along the baseline and controlling the images. After pointing the camera to the target object the server hands over the control to the client. Note that we can easy control the horizontal setting. For distance objects the parallel direction control is more difficult. Other functions are provided for taking separate series of photos or back the current photos up to a random directory. Starting photo taking is activated by the client script and performed quasi simultaneously. Consider that we take several photos for stacking and there is no need for a very short latency. Don't forget backing the photos up before starting a new job. Talking here about the File management is outside the aim of this paper. While the RPi Zero has limited power, post processing and stereoscopic calculations run on a host computer and don't effect the field work.

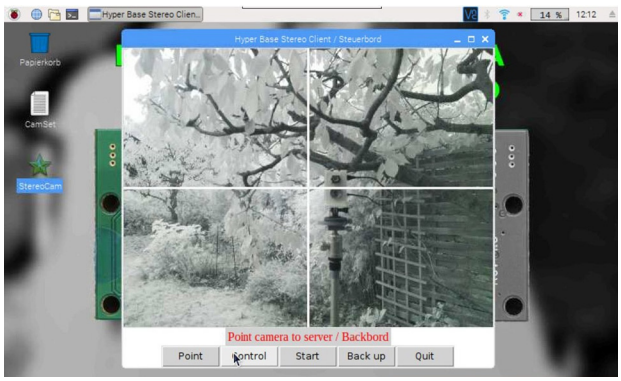

Figure 9: GUI interface for taking photographs

5.3 Infrared Photography

A digital camera sensor collects light from wavelengths approx. between 400 nm and 1000 nm. To convert the energy into colour information the light is separated by colour filtering. Well known is the Bayer-Pattern with two times green and a red and blue filter to match a pixels colour. In case of the Foveon sensor the pixel are layered instead of chessboard design. Protecting the sensor for light outside the visible range, beginning approx. at 600 nm, a IR blocking filter is assembled in front of the lens. This filter is missing on the Raspberry Pi NoIR camera module. If we take photos in daylight with the NoIR sensor it presents a purple coloured image. For now keeping out colour near infrared filters block the incoming light until a certain wavelength. For example 530 nm, 650 nm or 720 nm are customary for outdoor scenes. In landscape photography it is common to extend the sky by mixing the colour channels, for example changing blue with red. Figure 10 displays an image taken with a 720 nm filter and channel mixed between blue and red.

Figure 10: Infrared stereo image after channel mixing

The side-by-side stereo format in figure 10 looks like a greyscale image but it isn't. Colour information is still available. Finally the saturation has to be extended for getting the wanted false colour effect. The final images are deeply influenced by the existing light conditions and image manipulations as well. Observe figure 11 and enjoy the artificially output of the scene.

Figure 11: Saturation extended, cross view stereo format

REFERENCES

Image editing software.
https://imagemagick.org

Infrared and ultraviolet photography.
http://www.astrosurf.com/luxorion/photo-ir-uv.htm

OpenCV-Python Tutorials.
https://docs.opencv.org/3.4.7/d6/d00/tutorial_py_root.html

Open source stereoscopic camera based on Raspberry Pi.
http://stereopi.com/

Picamera documentation.
https://picamera.readthedocs.io/en/release-1.13/

Raspberry Pi Foundation. *https://www.raspberrypi.org/*

Spizhevoy, Alexey, Rybnikov, Aleksandr: OpenCV 3 Computer Vision with Python Cookbook. *Packt Publishing, Birmingham, 2018*