

Methods of information systems synthesis

UDC 004.932

doi: 10.20998/2522-9052.2019.1.06

K. Dergachov, L. Krasnov, O. Cheliadin, O. Plakhotnyi

National Aerospace University – Kharkiv Aviation Institute, Kharkiv, Ukraine

WEB-CAMERAS STEREO PAIRS COLOR CORRECTION METHOD AND ITS PRACTICAL IMPLEMENTATION

Subject of study. The article proposes new method and tool for color correction of web-cameras in stereo vision systems in order to improve the quality of their work. **The goal** is a comparative analysis of the quality indicators of well-known color correction methods and the development of a new method and working algorithms for the joint procedure for color correction and rectification of video frames of the left and right cameras. **Objectives:** the task was to carry out a theoretical analysis of the quality indicators of well-known color correction algorithms and methods, to develop new working algorithms, write the program codes of these algorithms in Python using the necessary OpenCV functions. Conduct experimental studies of these algorithms. Evaluate the performance of the stereo system in the laboratory, and test the reliability of the results obtained using statistical analysis methods. **Methods used:** comparative analysis of known methods and algorithms by statistical modeling, synthesis of new algorithms and evaluation of the effectiveness of their work by conducting laboratory field tests. **The results obtained:** a comparative analysis of the performance of the known methods of color correction of stereo cameras was carried out, new efficient algorithm was proposed for solving this problem. **Findings.** Scientific novelty of the results: new algorithm for correcting the color balance of webcams used in stereoscopic vision systems have been created, featuring high color correction accuracy and working in real time using the functions of the OpenCV library in the Python software environment.

Keywords: stereoscopic vision systems; color correction of video streaming images of web-cameras of a stereo system; joint rectification and color balance of the left and right chambers.

Introduction

Currently, the demand for low-cost but high-quality vision systems has increased significantly. The presence of cheap uncalibrated web-cameras with the ability to connect them to a computer via USB ports allows you to create fairly accurate stereo systems with easily adjustable parameters.

However even minor stereoscopic measurer parameters deviation leads to its work quality deterioration up to complete stereo effect losing. Factors analysis [1, 2] shows that spacial-time specifications and color rendering parameters discrepancy leads to measuring space deformation and measurement quality reduction of stereoscopic computer vision systems. Thus and so before exploitation start it is required to pass next steps for high quality stereo pair images formation:

- produce programmatic synchronisation of cameras;
- produce cameras internal calibration to determine its specifications;
- produce cameras pair external calibration using shared coordinate system;
- produce cameras resulting images rectification to set up mutual confirmity of cameras apipolar lines;
- produce stereo channels color correction.

There is a wide-spread perception that brightness difference and white color balance of stereo-channels are not so significant for stereo-matching. However, by authors opinion, different intensity of color components distorts visual perception and significantly influence on determination quality of space positioning parameters.

Steps of work listed above are relatively easy to implement using traditional methods. Thus in example

web-cameras internal calibration can be performed using Matlab's Camera Calibration Toolbox [3] or with help of OpenCV library [4]. But where are some unsolved issues. So the complexity of the precision adjustment of the stereo system is connected with the lack of a unified approach to the choice of methods for color correction of cameras and their rectification. It adds both perspective and relevance to the challenge and requires new constructive solutions.

Objectives

To increase stereo system work quality next objectives were defined: produce a comparative analysis of known color correction methods quality indicators, propose a new method and working algorithms for joint procedure of color correction and rectification of left and right cameras video frames. Codebase for these algorithms had should be written with Python language and OpenCV library help. It was necessary to present detailed results of stereo pair video color correction algorithms experimental studies. Performance evaluation of the stereo system should be carried out under laboratory conditions, and the reliability of the data obtained should be verified by statistical analysis of the data obtained.

Analysis of stereo pair cameras color correction methods

Cameras stereo pairs color correction as well as their spatial calibration, according to the authors, it is advisable to carry out in two stages. First, it is necessary to accurately set the white balance and fix the differences in the intensity of video sequences images color components of each camera separately, and then carry

out a joint adjustment of these parameters taking into account the results of mutual rectification of the cameras. Further, we consider this approach in more detail.

The main difficulties in performing color correction are due to local and global color discrepancies between cameras in the stereo pair [5]. A significant factor is also the presence of glare on objects when observing them from different angles. Such discrepancies are due to the differences in the photosensitive arrays and light filters of the stereo pair cameras, and their different position relative to the sources of the scene illumination.

Global color correction methods can be divided into three main classes: histogram matching, global color transfer and clustering method.

When classifying local images color correction methods, it is appropriate to distinguish two main methods – block-based method and local color correction for underwater stereo images.

Any of the above color correction methods requires the addition of such processing with detection and compensation algorithms for glare.

When using histogram matching techniques work color correction algorithms generally reduces delamination images in **RGB**-space into individual components **R**, **G** and **B**, followed by constructing histograms of the distribution of luminance pixels for the left and right views of stereoscopic. Then a conversion function is formed for one of the angles to obtain the minimum difference between these histograms for each color component.

Global color transfer methods are based on the synthesis of a linear transformation function of the **RGB** color space so that the target image has the same color tones as the original image. In this case, the average values of the pixels are calculated for each of the **R**, **G**, **B** channels for the source and target images, and then using the shift, rotation, and scaling matrices, the necessary joint transformation is performed on the three color channels.

Correction clustering method application is based on the selection of key frames and carrying out probabilistic clustering of pixels with the subsequent calculation of the color correction parameters and their interpolation in non-key frames. Similar principles are applied when building local color correction methods and glare removal algorithms [5–17].

Unfortunately, described images color correction methods, which have a number of undoubted advantages, are distinguished by a large amount of computation and are unsuitable for working in real time. Therefore, it is advisable to explore the possibilities of optimizing the structure of such algorithms for creating software that allows you to quickly and efficiently carry out color correction of stereo camera pairs

The essence of the proposed method

To solve the problem of joint color correction of stereo pair cameras, the authors proposed to use a method based on determining the vertical and horizontal parallax of the target mark image when combining images of the left and right cameras Δx and Δy . The resulting image offsets Δy are also used for the stereopair images rectification – bringing reference points to a single epipolar line.

Then on the left and right images of the scene two fragments of the same dimension are selected. Let's call them «regions of interest» **ROI**. Based on the previously obtained offsets Δx and Δy , digital convergence (combination of pixel accuracy) of these fragments (**ROI_left** and **ROI_right**) is performed. Next, the color characteristics of the right channel are corrected relative to the left one, which is selected as the reference.

Below is a more detailed description of this method, as well as an algorithm synthesized on its basis, a generalized block diagram of which is shown in Fig. 1.

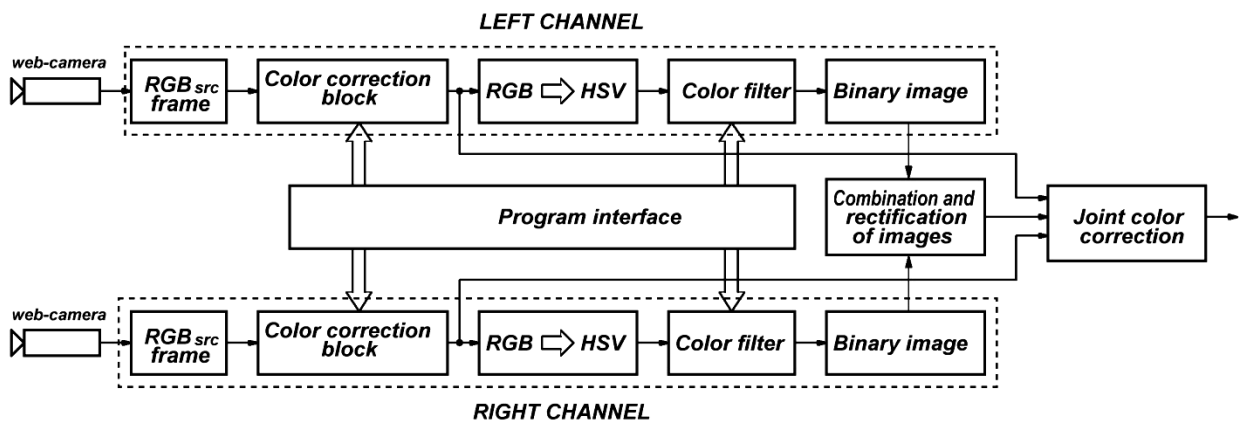


Fig. 1. The block diagram of algorithm for cameras color correction

When solving various problems of stereo comparison (for example, measuring the distance and building maps of the depth of the scene), it is common to use reference points that have certain distinctive features (brightness, color, etc.). The authors believe that the most stable and little sensitive to changes in the illumination of the scene is the target label of a saturated color (in our case red). It is fixed on a special screen. There-

fore, for reliable detection of the scene target in the frame used the method of its selection on color basis.

We will assume that the shooting of a scene with a target mark placed on it, is made by a stereo pair, previously calibrated by the standard method.

Received signals from webcams are a sequence of images presented in the **RGB** color space. Due to the high variability of the scene illumination, preliminary

color correction and white color balancing are usually required independently for each camera. Color correction algorithm can be selected online in the program interface menu.

RGB color space use is unproductive, since the desired color areas search implies the analysis of all three components **R**, **G** and **B**. Therefore, it is necessary to switch to another color space.

To search objects in images by color and brightness, as a rule, the **HSV** color space is used, where **Hue**, **Saturation** and **Value** are the parameters that determine hue, saturation and brightness. They are normalized as follows:

- **Hue** – $0 - 360^\circ$;
- **Saturation** – $(0 - 100)\%$;
- **Value** – $(0 - 100)\%$.

Therefore, the most important procedure after the input and color correction of stereo channels video data was to convert the **RGB** color space to the **HSV** space using the corresponding OpenCV function:

Therefore, the most important procedure after the input and color correction of stereo channels video data was to convert the **RGB** color space to the **HSV** space using the corresponding OpenCV function:

```
hsv=cv2.cvtColor(frame,cv2.COLOR_BGR2HSV)
```

The quality of the detection is significantly affected by the accuracy of components choice corresponding to the color of the observed object under different lighting conditions of the scene. Any point on the scale **H** defines a two-dimensional region with different values of **S** and **V**. But for the selected value of **H**, the target region in this two-dimensional space is more reliably determined by the condition: $V > V_{min}$ and $S > S_{min}$, having V_{min} , S_{min} – some constants. Therefore, it is customary to choose the target range on the **H** scale, indicating two values - **Hmin** and **Hmax**.

Note that the range of values for **HSV** parameters for highlighting red color usually lies within

```
lower_range = np.array([0,50,100])
up_range = np.array([10,255,255]).
```

The procedure for selecting the operating range of the **HSV** space parameters for the observed object is best carried out interactively, depending on the current lighting conditions and the color of the object itself. This approach is implemented in the following algorithm with the help of setting elements of the interface (sliders).

For selection of the desired color (for example, blue) components range was performed filtering (setting the color mask) using the threshold function

```
mask_color = cv2.inRange(hsv,
low_range,up_range).
```

After color filtering, the image of the frame is displayed in shades of gray (the range of brightness values of pixels from 0 to 255). This procedure allows you to reliably detect an object by its color, but it does not completely suppress background noise containing similar color components. This difficulty can be easily overcome by binarization with clipping on the brightness threshold of the image filtered by color. To perform it next function was used

```
mask_color = cv2.threshold(mask_color,p,
255,cv2.THRESH_BINARY)[1]
```

Note the selection of the cutoff threshold for brightness p in this function, it depends on the interference level. It is desirable to install it interactively.

The next important step in building the algorithm is to determine the coordinates of the target in the current frames of the left and right cameras. When working with binary images, the most productive way of this solution is to calculate the moment invariants of such an image, which allow to determine the coordinates with a high degree of accuracy. Image moments are calculated using the next function:

```
moments = cv2.moments(thresh,1).
```

The moments function returns an array of moments up to the third order. However, to calculate the coordinates of the object center, only moments of the first order $m01$ and $m10$ as well as a moment of zero order $m00$ are required. They are defined as follows:

```
dM01 = moments['m01'],
dM10 = moments['m10'],
dArea = moments['m00'].
```

Moment $m00$ is the number of all single pixels belonging to the selected object, and moments $m01$ and $m10$ are the sum of the X and Y coordinates of these pixels. To determine the coordinates of the center of the observed object, it is necessary to perform the normalization of these moments to the moment of zero order.

When carrying out this procedure, it is advisable to perform an additional threshold operation that allows you to filter out false objects, what possibility during the operation of the color filtering is not excluded. In this case, if there is a priori information about the size of the observed object, it is possible to eliminate false objects by the condition: if $dArea > N$, having N – the number of single pixels of the moment $m00$.

In the following example, the program will only respond to moments containing more than 50 pixels.

```
if dArea > 50:
x = int(dM10 / dArea),
y = int(dM01 / dArea).
```

This example eliminates random red lights that have a relatively small area in the frame.

After binarizing the images and determining the coordinates of the target mark in the left and right channels, these images are combined using the direct overlay method. The results of combining these images are calculated values of horizontal and vertical offsets Δx and Δy , which are used both for the rectification of stereo pair images and for the digital convergence of allocated **ROI** regions.

Color correction algorithms development for individual cameras

As noted earlier, it is advisable to first adjust the white color balance for each camera of the stereo pair separately, and then adjust the results obtained for a pair of cameras.

There are three main methods of white color balancing for digital cameras:

- automatic balance leveling;
- installations for a standard lighting conditions set;
- arbitrary installations on the reference object.

Let us consider in more detail the implementation of the basic algorithms for each of the listed methods.

«Gray world» algorithm. The most popular method of automatic color correction of images is the method called **«Gray World»**, since it is assumed that the sum of all the colors in the image of a natural scene gives gray color. When using this method in space, the original image of I_{src} is used with a dimension of $M \times N$ in to three components $(R_{src}, G_{src}, B_{src})$. Here I_{src} – source image and I_{dst} – destination image. Then the average brightness values of the pixels for each of these components are calculated.:

$$R' = \frac{1}{MN} \sum_{i=1}^M \sum_{j=1}^N R_{src}(x_i, y_j); G' = \frac{1}{MN} \times \sum_{i=1}^M \sum_{j=1}^N G_{src}(x_i, y_j); B' = \frac{1}{MN} \sum_{i=1}^M \sum_{j=1}^N B_{src}(x_i, y_j).$$

Having x_i and y_j are the current row and column numbers of the source image I_{src} . Then, the generalized coefficient of average brightness MB (Medium Brightness) of pixels is determined as follows:

$$MB = (R' + G' + B')/3$$

and scaling the pixels of the I_{src} scene with the corresponding coefficients to obtain $R_{dst}, G_{dst}, B_{dst}$ – components of the output image I_{dst} :

$$R_{dst} = R_{src} \cdot MB/R, G_{dst} = G_{src} \cdot MB/G, B_{dst} = B_{src} \cdot MB/B.$$

After adjustment, these components are combined into a color image I_{dst} that can be easily compared with the source image I_{src} visually. For additional clarity, images of the difference between the output and initial brightness of all pixels are used for the corresponding color components obtained as a result of conversion:

$$\Delta R = R_{dst} - R_{src}, \Delta G = G_{dst} - G_{src}, \Delta B = B_{dst} - B_{src}.$$

«Perfect reflector» algorithm. It is based on the assumption that the brightest areas of the image relate to glare on surfaces, the model of reflection of which is such that the color of the glare is equal to the color of the illumination (dichromatic model). For such a model, according to the three color components $R_{src}, G_{src}, B_{src}$ of the source image I_{src} , it is necessary to determine the maximum values of the pixels brightness $R_{srcmax}, G_{srcmax}, B_{srcmax}$. After that, the brightness of the image I_{src} pixels is scaled according to the rule:

$$R_{dst} = R_{src} \cdot 255/R_{srcmax}, G_{dst} = G_{src} \cdot 255/G_{srcmax}, B_{dst} = B_{src} \cdot 255/B_{srcmax}$$

The contrast algorithm «autolevels» is based on the idea of stretching the intensity of the color components $R_{src}, G_{src}, B_{src}$ of the original image over the entire range. For this, the maxima and minima of the color components $R_{srcmax}, G_{srcmax}, B_{srcmax}, R_{srcmin}, G_{srcmin}, B_{srcmin}$ are determined, and then the following transformations are applied

$$R_{dst} = (R_{src} - R_{srcmin}) \cdot (255 - 0) / (R_{srcmax} - R_{srcmin}), G_{dst} = (G_{src} - G_{srcmin}) \cdot (255 - 0) / (G_{srcmax} - G_{srcmin}), B_{dst} = (B_{src} - B_{srcmin}) \cdot (255 - 0) / (B_{srcmax} - B_{srcmin}).$$

Algorithm for improving the color images contrast «Equalization» with the transition from **RGB** to **YUV** color space. This is a color model in which the color is represented as 3 components - brightness (**Y**) and two color difference (**U** and **V**). In the **YUV** image, only the brightness component (**Y**) is subjected to equalization using OpenCV library function:

```
img_yuv[:, :, 0] =
cv2.equalizeHist(img_yuv[:, :, 0]).
```

Then the image is inversely converted from **YUV** to **RGB**. The color balance remains unchanged, since the color difference components **U** and **V** have not been transformed.

For clarity, we show an example of the the contrast enhancement algorithm program code for a color image synthesized in Python using the OpenCV functions:

```
import cv2
import numpy as np
img = cv2.imread('input.jpg')
img_yuv =
cv2.cvtColor(img, cv2.COLOR_BGR2YUV)
# equalize the histogram of the Y channel
img_yuv[:, :, 0] =
cv2.equalizeHist(img_yuv[:, :, 0])
# convert the YUV image back to RGB
img_output =
cv2.cvtColor(img_yuv, cv2.COLOR_YUV2BGR)
cv2.imshow('Color input image', img)
cv2.imshow('Histogram equalized', img_output)
cv2.waitKey(0)
```

There are other options for building such algorithms. However, it should be emphasized that the choice of a specific camera color correction algorithm (or their combination) is due to the peculiarities of the scene lighting. Therefore, it is advisable to provide the possibility of an interactive algorithm change when creating the software for this task.

Algorithm of joint color correction of stereo pair cameras

Next, we consider the operation of the algorithm for joint color correction of scene images in the left and right stereo channels (Fig. 2, 3).

After the images are rectified by shifting the image of the right channel relative to the left channel by the amount of vertical offset Δy , areas of interest are selected (**ROI_left** and **ROI_right**). Note that the convergence (exact alignment) of these images is based on the horizontal offset Δx between the left and right channel of the stereo pair.

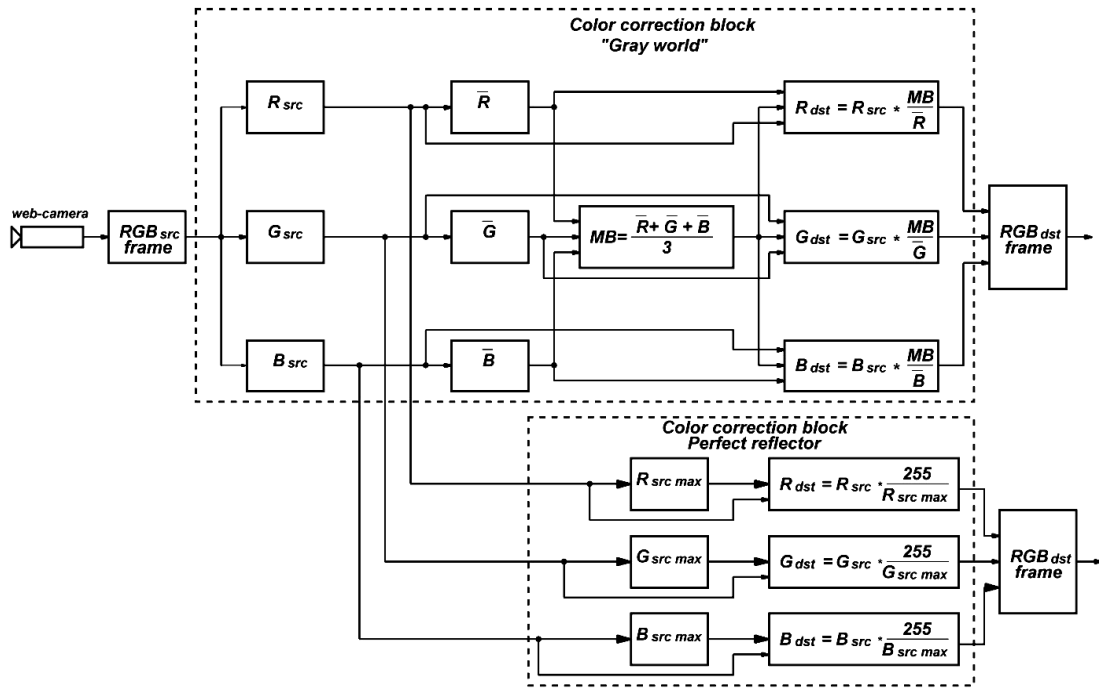


Fig. 2. Block diagram of possible color correction algorithms for a separate stereo pair channel

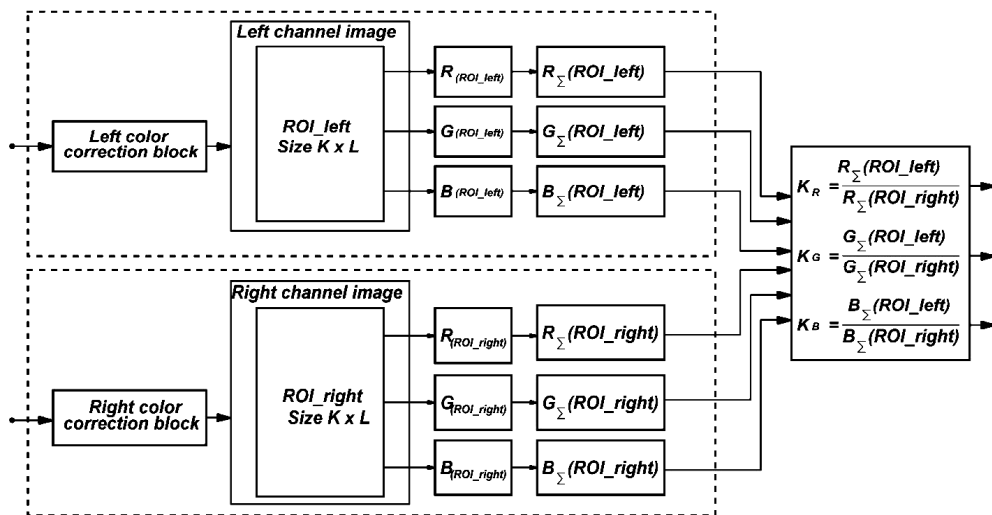


Fig. 3. White balance adjustment chart for web cameras of a stereo pair

It should be remembered that the size of the *ROI* channels should adequately reflect the nature of the scene light. But they cannot be chosen too large, since this will lead to an unjustified increase in the computational complexity of the algorithm. In our opinion, the optimal size of *ROI* is the central fragment of an image of 100x100 pixels with a frame format is 480x640. There is no doubt that the *ROI* fragments should be selected interactively. The joint color correction of the stereo pair cameras is performed under the assumption that one of the images is taken as basis (referenced image). In our case, *ROI_left* is taken as a standard for white color balance and image brightness. The second image (*ROI_right*) will be considered as dependent, and the color components of the image will be corrected relatively to the base one.

When mapping *ROI*, you can build histograms of the distribution of the color components *R*, *G* and *B* for visual comparison. However, it is clear that such an

approach is ineffective. It is much more productive to evaluate and compare the numerical integral characteristics of the individual color components *ROI_left* and *ROI_right*. Therefore, in this algorithm, the total brightness indices of the individual color components for the left and right channels are calculated:

$$R_{\Sigma}(ROI_{left}) = \sum_{k=1}^K \sum_{\ell=1}^L R_{srcleft}(x_k, y_{\ell});$$

$$G_{\Sigma}(ROI_{left}) = \sum_{k=1}^K \sum_{\ell=1}^L G_{srcleft}(x_k, y_{\ell});$$

$$B_{\Sigma}(ROI_{left}) = \sum_{k=1}^K \sum_{\ell=1}^L B_{srcleft}(x_k, y_{\ell}),$$

having x_k and y_{ℓ} – are rows and columns current numbers of the source image *ROI_left*.

Indicators $R_{\Sigma}(ROI_{left})$, $G_{\Sigma}(ROI_{left})$ and $B_{\Sigma}(ROI_{left})$ are similarly calculated for the right channel. Based on the information on the total brightness of the color components of images ROI_{left} and ROI_{right} , correction factors are calculated:

$$\begin{aligned} k_R &= R_{\Sigma}(ROI_{left}) / R_{\Sigma}(ROI_{right}), \\ k_G &= G_{\Sigma}(ROI_{left}) / G_{\Sigma}(ROI_{right}), \\ k_B &= B_{\Sigma}(ROI_{left}) / B_{\Sigma}(ROI_{right}). \end{aligned}$$

The obtained coefficients k_R, k_G and k_B are used to adjust the white balance of the dependent image (in our case, the right stereo channel) relative to the base (left stereo channel). The color components R, G and B are multiplied by these coefficients. Note that with sufficiently accurate white color balancing, it is usually not necessary to adjust the brightness in stereo channels.

Nevertheless, we consider the method of compensating for differences in the integral brightness of the left and right channels. For comfortable images perception when playing stereoscopic recording, it is necessary to adjust the total brightness of the base image in the region of interest $I_{\Sigma}(ROI_{left})$. It can be calculated using weights p according to formula:

$$\begin{aligned} I_{\Sigma}(ROI_{left}) &= p_1 \cdot R_{\Sigma}(ROI_{left}) + \\ &+ p_2 \cdot G_{\Sigma}(ROI_{left}) + p_3 \cdot B_{\Sigma}(ROI_{left}). \end{aligned}$$

The coefficients p are determined by the physiological properties of the human eye [18] and are taken equal to:

$$p_1 = 0.2126; p_2 = 0.7152; p_3 = 0.0722.$$

Here is a detailed record of the formula for calculating the integral brightness of the base (left) channel:

$$\begin{aligned} I_{\Sigma}(C) &= p_1 \cdot \sum_{k=1}^K \sum_{\ell=1}^L R_{src\ left}(x_k, y_{\ell}) + p_2 \times \\ &\times \sum_{k=1}^K \sum_{\ell=1}^L G_{src\ left}(x_k, y_{\ell}) + p_3 \cdot \sum_{k=1}^K \sum_{\ell=1}^L B_{src\ left}(x_k, y_{\ell}). \end{aligned}$$

Similarly, the indicator of the total brightness $I_{\Sigma}(ROI_{right})$ is calculated for the right (dependent) channel. But to achieve in stereo channels the full balance of the total brightness, it is necessary to adjust the indicators obtained for this channel using the correction factors k_R, k_G and k_B obtained earlier.

$$\begin{aligned} I_{\Sigma}(ROI_{left})^* &= \left(p_1 \cdot \sum_{k=1}^K \sum_{\ell=1}^L R_{src\ right}(x_k, y_{\ell}) \right) \cdot k_R + \\ &+ \left(p_2 \cdot \sum_{k=1}^K \sum_{\ell=1}^L G_{src\ right}(x_k, y_{\ell}) \right) \cdot k_G + \\ &+ \left(p_3 \cdot \sum_{k=1}^K \sum_{\ell=1}^L B_{src\ right}(x_k, y_{\ell}) \right) \cdot k_B. \end{aligned}$$

Thus, it is possible to correct the relative deviation of the channels total brightness after adjusting the white color balance.

Quantitative assessment of the used algorithms effectiveness

You can use different criteria to describe the degree of compliance of the source I_{src} and the resulting I_{dst} images transformations.

We believe that in our case it is appropriate to calculate the dispersion indicators for each color component:

$$\begin{aligned} MSE_R &= \frac{1}{MN} \sum_{i=1}^M \sum_{j=1}^N \left(R_{src}(x_i, y_j) - R_{dst}(x_i, y_j) \right)^2; \\ MSE_G &= \frac{1}{MN} \sum_{i=1}^M \sum_{j=1}^N \left(G_{src}(x_i, y_j) - G_{dst}(x_i, y_j) \right)^2; \\ MSE_B &= \frac{1}{MN} \sum_{i=1}^M \sum_{j=1}^N \left(B_{src}(x_i, y_j) - B_{dst}(x_i, y_j) \right)^2. \end{aligned}$$

Dispersion of information parameters is not the only criterion for assessing the degree of difference between two images or their color components. For this purpose, it is possible to successfully use the linear norm of difference (the normalized sum of the modules of two images corresponding pixels brightness values differences). This data is used to evaluate the effectiveness of the algorithms.

But most clearly, the color components change degree is characterized by pair wise represented histograms of pixel brightness distribution. Such visual data successfully complements digital quality indicators.

Laboratory stand and program for stereo vision problems research

The composition and design of the stand. For the joint color correction of a stereo pair consisting of two web-cameras, a laboratory stand was used, the general view of which is shown in Fig. 4, a. It includes a computer with software, a stereo pair based on two identical web-cameras with the possibility of connecting them to a computer via USB ports, as well as a rectangular screen on a tripod for mounting a special target mark. Design features of the stereo pair are showed in more detail in Fig. 4, b.

When rigidly mounting cameras relative to each other, it is possible to change the position of the stereo pair in height and turn it to 360° around the vertical axis. In addition, each camera can be rotated at a small angle in the horizontal and vertical planes. But the main feature is the adjustment of the basic distance between the cameras using a special control ruler. This ensures that the base distance is set to within a fraction of a millimeter. With the help of such a stand, it is possible to successfully solve a wide range of stereo vision problems – to calibrate cameras, to perform the rectification of stereo images, to perform color correction, etc.

Stereo camera color correction software. While formulating the problem, the concept of the proposed method, the structure of the algorithms for its implementation were discussed in some detail, and the choice of means and methods of programming was justified. The generalized structure of the software is shown in Fig. 1.



Fig. 4. General view of the laboratory stand – a; stereo pair design – b

Python was chosen as programming language with help OpenCV library resources usage. In this section we will discuss the possibilities of the created software, the most important element of which is the user interface.

It is displayed on Fig. 5. In essence, the program interface is a specialized desktop program. Such interface construction allows flexible interactive system configuration.

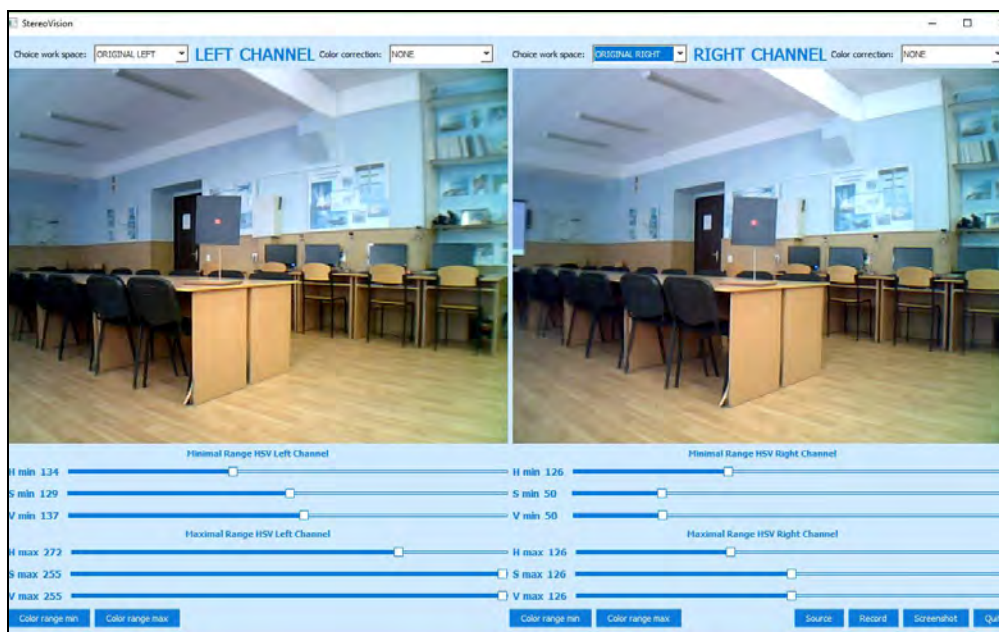


Fig. 5. General view of the program interface

Two windows of 640x480 pixels each are symmetrically arranged on the program's desktop, which corresponds to the avi * format of the left and right channels of the stereo system. In these windows, you can display in the various combinations the input frames of the received stereo stream, the results of video frame conversions and the final results of processing. In addition, there is a row of buttons and controls that allow you to interactively configure the system to solve the desired task in the shortest possible time on the desktop.

Let us consider their properties in more detail. In the lower left corner of the interface there are four buttons: “Source”, “Record”, “Screenshot” and “Quit”. By pressing the “Source” button, the data entry method is selected – from the stereo pair webcams using “online” mode or from a previously recorded file. The second input method provides the possibility of multiple repetition of video data processing using various algorithms, which allows you to objectively evaluate and compare the results of work in different ways.

When registering video data from webcams in the application, it is possible to record a video stream of the

desired length by pressing the “Record” button. If further replay is necessary in the program, this record should be saved in the Python system folder. The recording is saved as two files for left and right cameras in avi* format. The file name displays the system time of synchronous registration (for example, left_cam_19-01-31-12-59, right_cam_19-01-31-12-59, which shows the year, month, date and time a registration start with an accuracy of one second).

Similarly, for analyzing the operation of channel color correction algorithms by clicking the “Screenshot” button, two images of current frames in the png format are saved in the system folder (left_cam_19-01-31-13-10.png and right_cam_19-01-31-13-10.png). The benefits of such option are evident from the example of creating a series of chess field snapshots for performing stereo pair calibration procedure. This requires a series of 10-15 chessboard images in various angles for the left and right channels. The technology of such shooting is shown in Fig. 7.

To select the mode of work with video stream frames, two drop-down menus are provided for each chan-

nel in the upper part of the interface - “Choice work space” and “Color correction”. Work with one of them is shown in Fig. 6. The first menu allows you to display the view of the ORIGINAL LEFT / RIGHT input sequences, BINARY LEFT / RIGHT binary images obtained after color filtering, the BINARY TOGETHER cumulative effect and frame images after RESULT LEFT / RIGHT data processing. The “Color correction” menu contains the algorithms names for color correction of the corresponding stereo channel – GRAY WORLD, AUTOLEVELS, PERFECT REFLECTOR and EQUALISATION. Note that it is possible to use different color correction algorithms in the left and right channels independently of each other.

But the most interesting is setting the parameters of color filters to highlight objects of interest to us and assess their position in the frame. To do this, use the settings of the color filter, the results of which are displayed in the selected window BINARY LEFT or BINARY RIGHT.

Sliders enable interactive selection of H, S and V parameters desired range in the *HSV* color space in range *low_range* – *up_range*. However, the procedure for selecting the *HSV* parameters to highlight the object of interest to you is quite a laborious process, sometimes requiring a long time. In Fig. 8 clearly shows what the result of binarization looks like with clipping on the frames brightness threshold after their color filtering and inaccurately setting the desired range of *HSV* parameters. To overcome this drawback, the program introduced a pre-setting of *HSV* color filter parameters range. In the bottom row of interface control elements, for each stereo channel, a pair of “Color range min” and “Color range max” buttons is provided, with which you open the standard “Color” settings for setting color characteristics (Fig. 9). After selecting the required color characteristics, pressing the “Ok” button in this window will automatically adjust the settings for the *HSV* interface parameters.

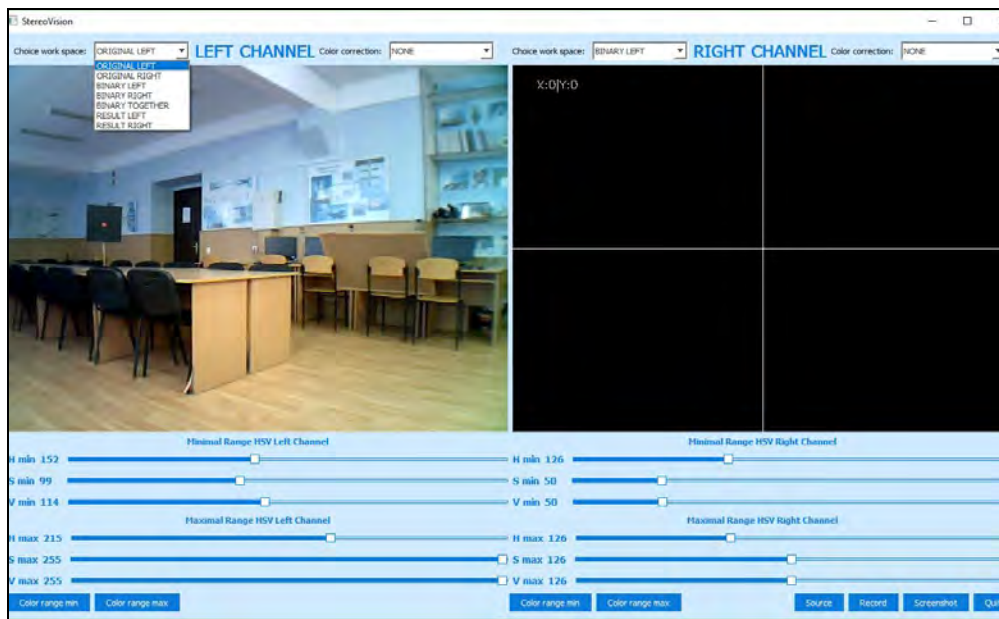


Fig. 6. Selecting the video viewing mode and processing results

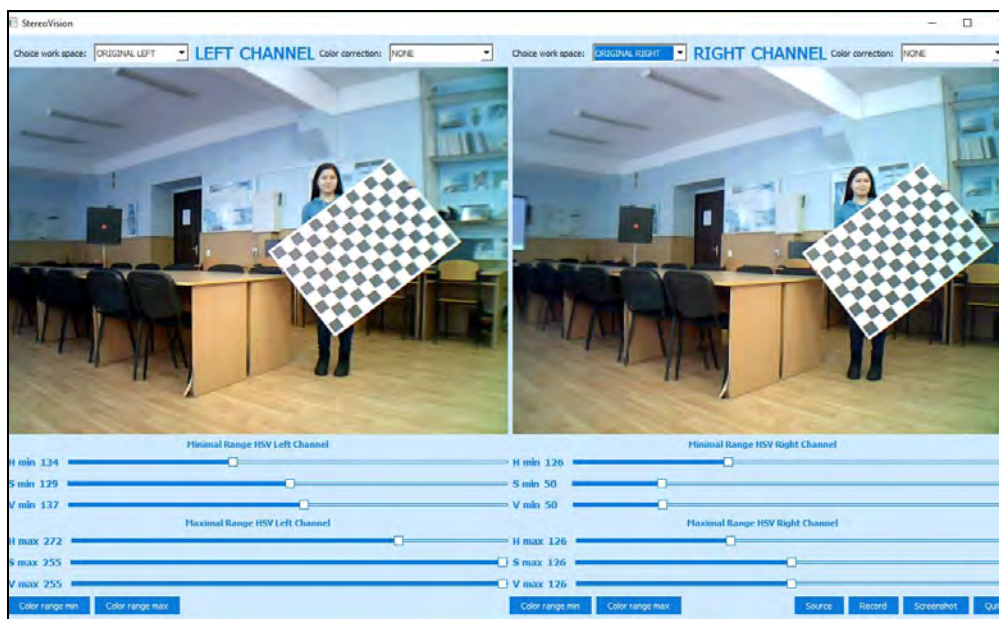


Fig. 7. An example of using the “Screenshot” option in the stereo calibration task

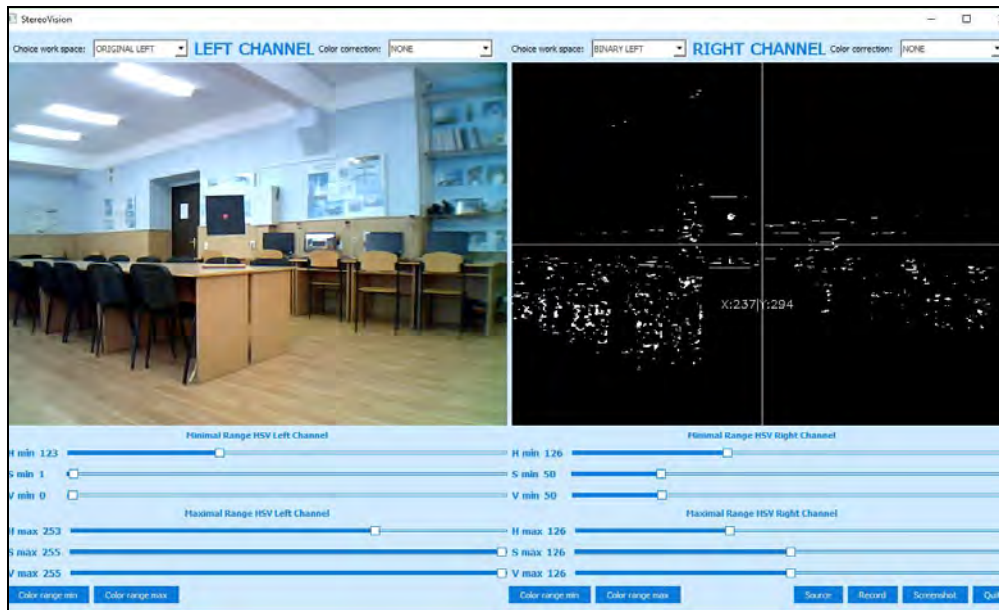


Fig. 8. Example of color filter parameters manual adjustment

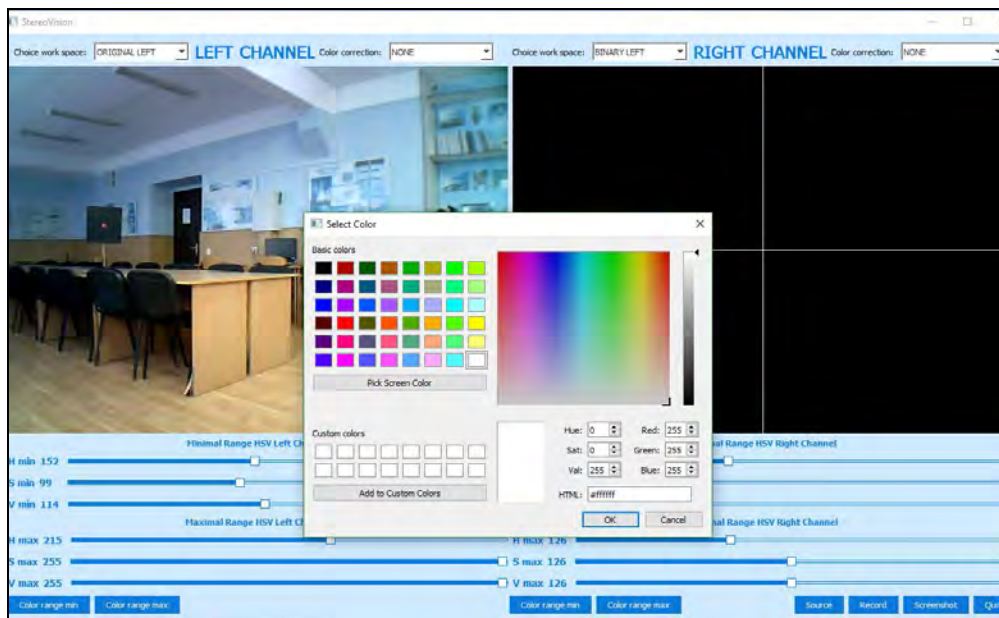


Fig. 9. Color Filter Preset Window

After presetting the parameters of the color filter, it is enough to make small adjustments to the position of the color adjustment sliders for user.

Note that the program has user-friendly and intuitive interface that does not require special user skills.

Conducting experimental studies and their main results

To assess the effectiveness of the proposed methods and algorithms for color correction of web cameras stereo pair, experimental studies were conducted, which can be divided into two main groups - analysis of algorithms quality for color correction of a separate stereo channel, and evaluation of algorithms quality used for joint color correction.

With the help of a laboratory stand and software, a stereoscopic shooting of a scene with a target mark on it was carried out (see Fig. 8). In Fig. 10 the results of the

calculation of Δx and Δy offsets, due to the displacement of the coordinates of the target mark in the left and right frames are shown.

The combination of channels by direct overlay increases visibility when evaluating Δx and Δy offsets, and advertise their values. This data is shown in the window `BINARY TOGETHER`. From the numerical data it is clear that the Δy offset equals 1 pixel. This should be taken into account when rectifying channels.

Let us further consider the method for determining the performance of color correction algorithms for individual channels. It consists of three components:

- visual assessment of processing source image I_{src} quality. Unfortunately, it is often subjective and depends on the quality of the experts' view. It does not always make it possible to correctly distinguish the differences between the output image and the source image visually.

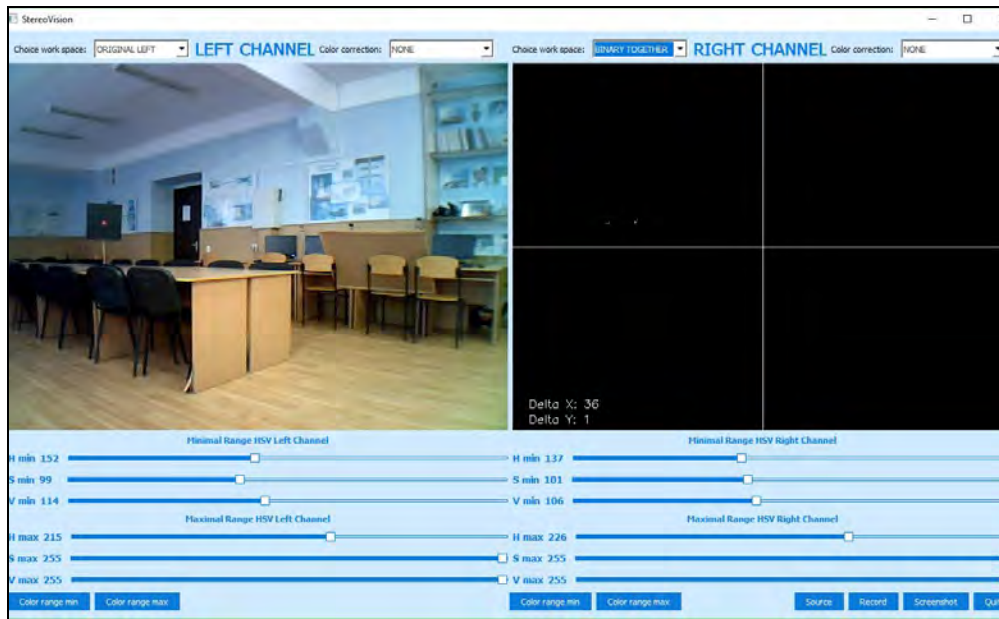


Fig. 10. Target mark offsets Δx and Δy in the left and right channels

- histograms of source images are displayed as solid curves, and histograms of output images are dotted. It clearly shows how the ratio of bright and dark pixels changes as a result of the action of the tested algorithm (see Tables 1 – 5).

- color correction scores.

For this, a special program block is used. It is called “Algorithm Tester” and allows you to evaluate the performance of left or right stereo channel color correction algorithms using individual frames received by the “Screenshot” command for the current conditions of the scene’s illumination.

As a generalized indicator of the efficiency of the algorithms used, the authors proposed a statistical discrepancy vector for the brightness σ_{Σ} of the original

image I_{src} and the transformed I_{dst} . Further we will call it as “**Color Balance Vector**”. It is presented in the orthogonal three-dimensional **RGB** color space. The color space in this case is usually represented in the form of an **RGB** color cube, shown in Fig. 11. It is known that the diagonal points of this cube, connecting two opposite vertices with coordinates (0, 0, 0) and (255, 255, 255), are called gray scale. They change their shade from black to bright white. In this coordinate system, the beginning of the vector σ_{Σ} is aligned with the zero (black) point, its module is determined by the set of dispersion indicators for each color component MSE_R, MSE_G and MSE_B used to describe the degree of conformity of the original and resulting images.

Table 1 – Left camera. Processing method «Gray World»

Channel name	Mean square error (MSE)	Standard deviation σ_{RGB}
Blue channel	15.98515	3.99814
Green channel	12.70391	3.56425
Red channel	36.23819	6.01982

Color balance vector $\sigma_{\Sigma} = 8.05774$

Table 2 – Right camera. Processing method «Gray World»

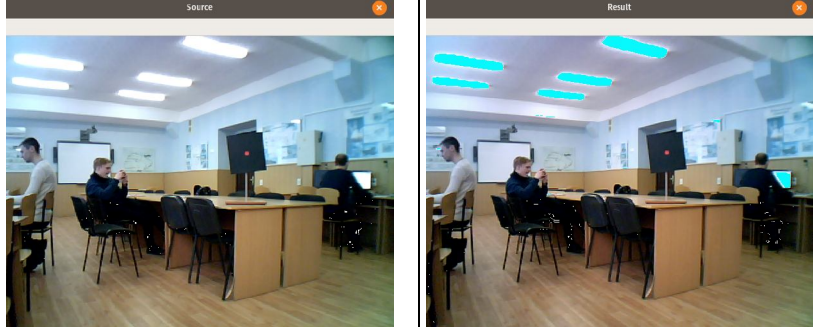
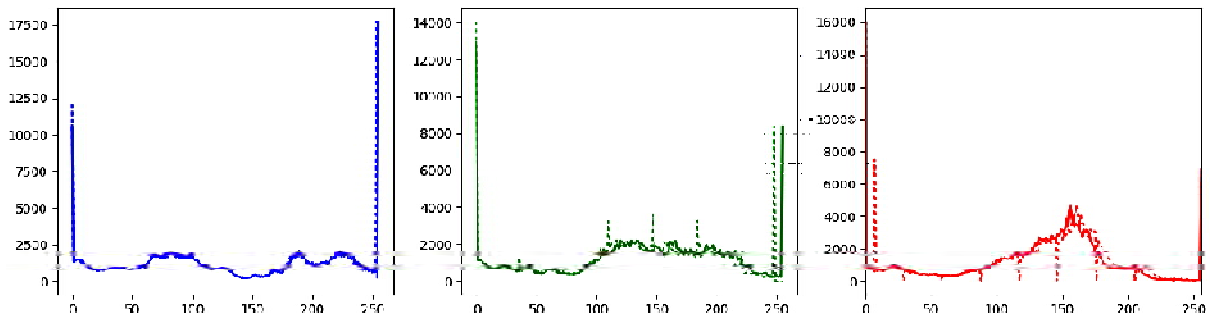
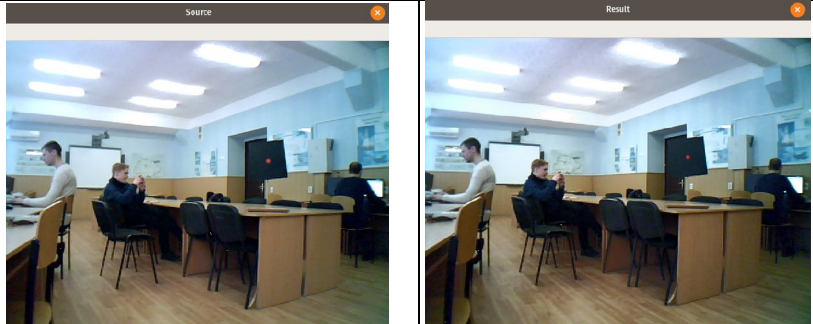
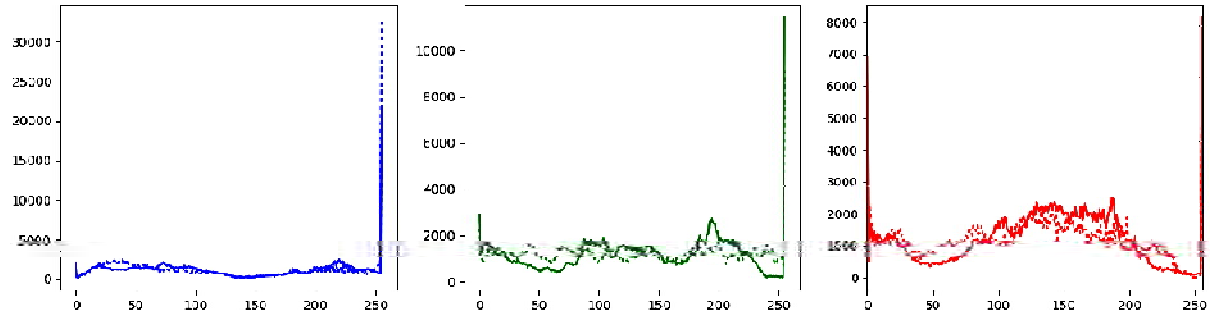
	Channel name	Mean square error (MSE)	Standard deviation σ_{RGB}
	Blue channel	2.29687	1.51554
	Green channel	19.58033	4.42496
	Red channel	19.93552	4.46492
			
Color balance vector $\sigma_{\Sigma} = 6.46627$			

Table 3 – Left camera. Processing method «Equalization»

	Channel name	Mean square error (MSE)	Standard deviation σ_{RGB}
	Blue channel	54.82341	7.40428
	Green channel	62.99173	7.93673
	Red channel	65.16232	8.07232
			
Color balance vector $\sigma_{\Sigma} = 13.52691$			

The formulas for calculating these variances are given above. The magnitude of the module "color balance vector" is determined by the formula

$$\sigma_{\Sigma} = \sqrt{MSE_R + MSE_G + MSE_B} ,$$

or by the sum of squared deviations (standard deviation $\sigma_{R,G,B}$) as follows:

$$\sigma_{\Sigma} = \sqrt{\sigma_R^2 + \sigma_G^2 + \sigma_B^2} .$$

If necessary, you can analyze the angular position of the color balance vector σ_{Σ} by determining these

angles in relation to the magnitude of the vector to its projections on the coordinate axes of the cube $(\sigma_R, \sigma_G, \sigma_B)$.

The results of experimental studies are presented below in the form of Tables 1-5, each of which contains data on the analysis of the one of color correction algorithms effectiveness. The source and transformed images, statistical processing data and histograms of the brightness distribution of the three color components are included in the table cells. Histograms of the input (solid line) and output (dashed line) images are arranged in pairs for each color component.

Table 4 – Right camera. Processing method «Equalization»

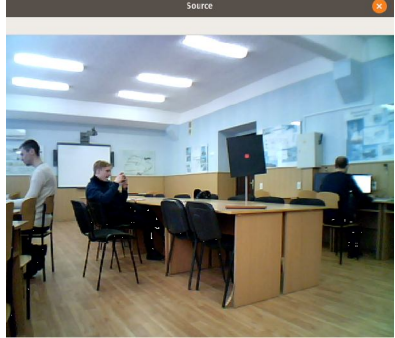
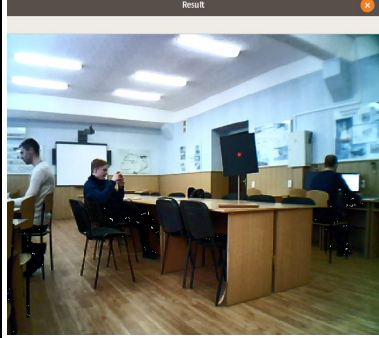
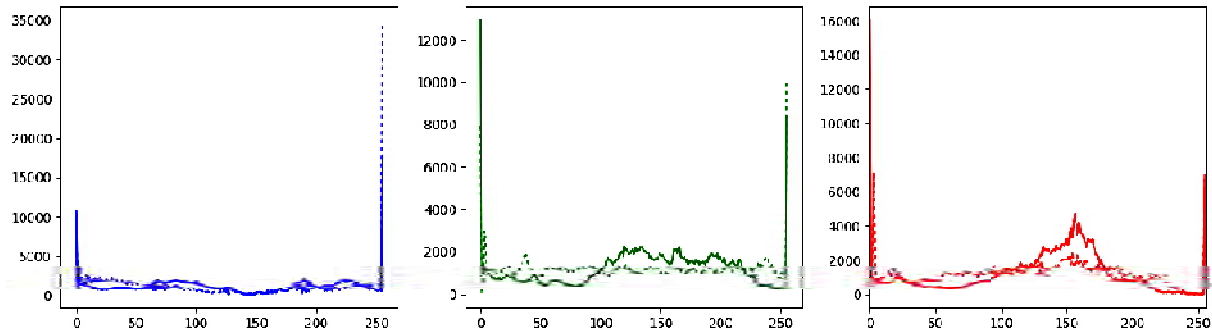
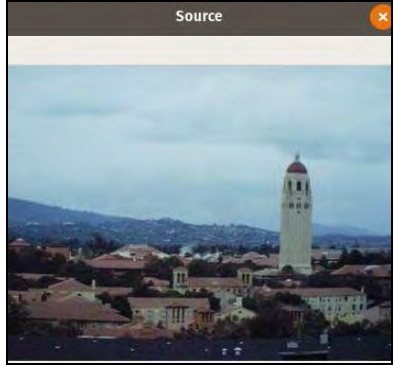
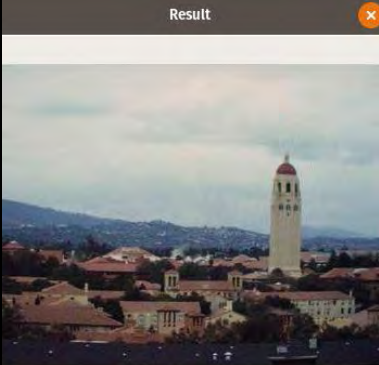
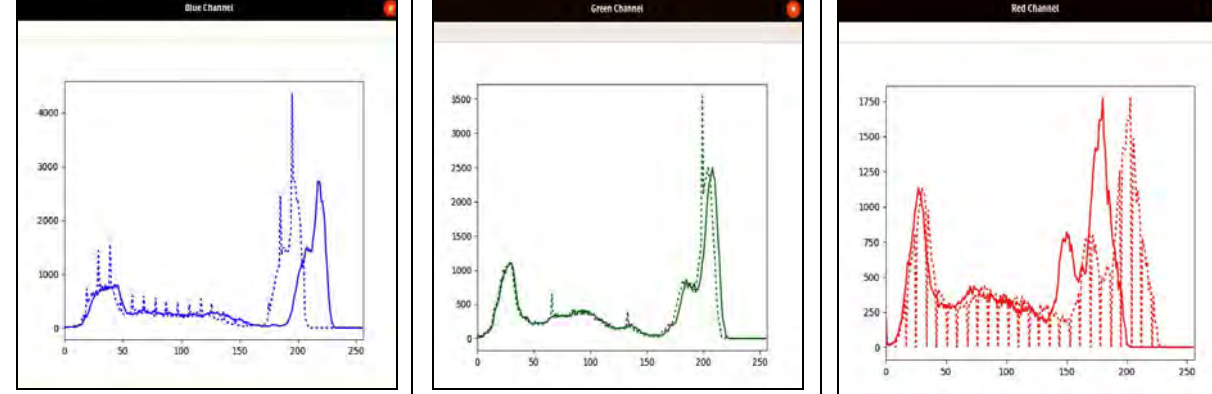
				Channel name	Mean square error (MSE)	Standard deviation σ_{RGB}
				Blue channel	77.15999	8.78407
				Green channel	85.27715	9.23456
				Red channel	83.63685	9.14532
						
Color balance vector $\sigma_{\Sigma} = 15.6867$						

Table 5 – Processing method «Gray World»

				Channel name	Mean square error (MSE)	Standard deviation σ_{RGB}
				Blue channel	110.11258	10.49345
				Green channel	7.71392	2.77739
				Red channel	82.72612	9.09539
						
Color balance vector $\sigma_{\Sigma} = 14.161$						

This creates convenience for analysis. Also, according to the results of statistical analysis, the module of the color balance vector is calculated and entered into the table σ_{Σ} .

Analysis of a particular of the investigated algorithms and comparative analysis of their quality should be carried out on the basis of an objective complex cri-

terion, which should take into account both the nature of image changes visual perception and the variability of histograms as a transformations result.

An equally important aspect of the analysis is the accounting of statistical indicators. It is also necessary to evaluate the module and the angular position of the color balance vector σ_{Σ} in orthogonal *RGB* color space.

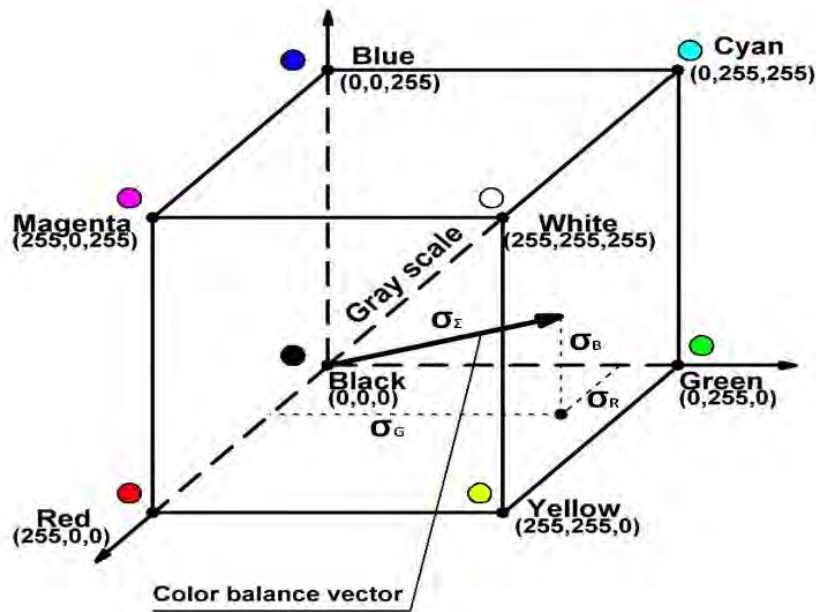


Fig. 11. Color balance vector in orthogonal *RGB* space

A comparative analysis of the four color correction algorithms showed that each of them has both advantages and disadvantages.

It should be noted that when processing the image using the “Gray World” algorithm, a significant disturbance of the color balance occurred (see Table 1). This means that such an algorithm is not effective under all conditions of scene lighting.

The example given in Table 5 shows that under certain conditions it is quite applicable and sufficiently effective. For convenience of analysis, the data on the magnitude of the modulus of the color balance vector σ_{Σ} are tabulated.

Table 6 – Color balance vector σ_{Σ}

Processing method	Color balance vector σ_{Σ}
Gray World	14.928
Perfect Reflector	12.619
Autolevels	12.494
Equalization	17.985

It is clear that in the absence of noticeable irregularities in the image of the white balance, the discrep-

ancy of the histograms of the original and processed images is reduced, and in the limit becomes zero. Therefore, a positive factor is the decrease in the modulus of the color balance vector. According to the authors, the use of the “Autolevels” and “Equalization” algorithms is almost most preferable.

The latter is most effective at low levels of scene illumination.

The described method for evaluating the effectiveness of color correction is fully applicable to the joint color balancing of stereopair cameras.

Conclusion

A new technique and color correction algorithms for stereo pair webcams in binocular vision systems are proposed. Experimentally analyzed the effectiveness of their use.

The software in the Python language was created using the resources of the OpenCV library, which allows realizing the operation of these algorithms in real time.

It is possible to adapt the program to various changes in the lighting conditions of the scene. The use of these results makes it possible to improve the quality of operation of the binocular vision systems in solving various problems of both stereo-visualization and stereo-matching.

REFERENCES

1. Protasov, S.I., Kurgalin, S.D. and Krylovetsky, A.A. (2011), “Using webcams as a source of stereo pairs”, *Vestnik VSU, Series: system analysis and information technologies*, No. 2, Voronezh, pp. 80–86.
2. Kruchinin A. (2018), *Digital image processing as signals. Computer graphics*, available at : <http://www.myshared.ru/slide/529382/>.
3. Lyudvichenko, V. (2018), *Stereo video color correction methods*, Video Group CS MSU Graphics & Media Lab (Video Group), available at : www.compression.ru/video/.
4. (2018), *Calibrate the camera in the Camera program “Calibration Toolbox for Matlab”*, available at : http://www.vision.caltech.edu/bouguetj/calib_doc/.
5. Linda G., Shapiro and George C., Stockman, (2001), *Computer Vision*, Prentice Hall, 580 p.
6. Konushin, A. (2008), *Stereoreconstruction*, MSU, Moscow, available at : <http://courses.graphicon.ru/main/vision2008>.

7. Kruchinin A. (2018), Stereo functions in OpenCV, available at : <https://docplayer.ru/53282398-Funkcii-stereozreniya-v-opencv.html>.
8. Chafonov, V.G. and Gazeeva I.V. (2014), "Methods of imaging a stereo pair with a given parallax value", *Scientific and Technical Journal of Information Technologies, Mechanics and Optics*, No. 6 (94), St. Petersburg, pp. 41–49.
9. Joseph, Howse and Joe, Minichino (2015), *Learning OpenCV 3 Computer Vision with Python - Second Edition*, September 2015, Packt Publishing, ISBN: 9781785289774.
10. Saurabh, Kapur (2017), *Computer Vision with Python 3*, Packt Publishing, August 2017, ISBN: 978-1-78829-976-3.
11. Prateek, Joshi (2015), *OpenCV with Python By Example*, Packt Publishing, September 2015, ISBN: 978-1-78528-393-2.
12. Computer vision library OpenCV available at : http://docs.opencv.org/trunk/doc/py_tutorial/py_objdetect/py_face_detection/py_face_detection.html.
13. Sivaram, M., Porkodi, V., Mohammed A.S. and Manikandan V. (2019), "Detection of Accurate Facial Detection Using Hybrid Deep Convolutional Recurrent Neural Network", *ICTACT Journal on Soft Computing*, Vol. 09, Issue 02, pp.1844-1850, DOI: <https://doi.org/10.21917/ijsc.2019.0256>
14. Ruban, I., Kuchuk, H., Kovalenko, A. (2017), "Redistribution of base stations load in mobile communication networks", *Innovative Technologies and Scientific Solutions for Industries*, No. 1 (1), P. 75–81, DOI: <https://doi.org/10.30837/2522-9818.2017.1.075>
15. Bovykin A.V. (2016), *Development of multimedia applications using OpenCV and IPP libraries*, INTUIT, Moscow, 515 p., available at : <http://www.iprbooksshop.ru/39564/>.
16. Evsegneev, O. (2018), *OpenCV on python: search for a color object*, available at : <http://robotclass.ru/tutorials/opencv-moments-color-object-search/>.
17. Qaru S. (2018), *Drawing histogram in OpenCV with Python*, available at : <http://qaru.site/questions/371882/drawing-histogram-in-opencv-python>.
18. Hyubel, D. (1990), *Eye, Brain, Vision*, Mir, Moscow, 172 p.

Received (Надійшла) 22.01.2019

Accepted for publication (Прийнята до друку) 27.02.2019

Метод колірної корекції web-камер стереопари та його практична реалізація

К. Ю. Дергачев, Л. О. Краснов, О. О. Челядин, О. В. Плахотный

Предмет вивчення. У статті запропоновано нові методи і засоби колірної корекції web-камер в системах стереозору з метою підвищення якості їх роботи. **Метою** є порівняльний аналіз показників якості відомих методів корекції кольору і розробка нової методики і робочих алгоритмів спільної процедури колірної корекції та ректифікації фреймів відеопотоку лівої і правої камер. **Завдання.** Ставилось завдання провести теоретичний аналіз показників якості відомих алгоритмів корекції кольору, розробити нові робочі алгоритми, програмні коди цих алгоритмів написати на мові Python з використанням необхідних функцій OpenCV. Провести експериментальні дослідження цих алгоритмів. Оцінити ефективність роботи стереосистеми в лабораторних умовах, а достовірність отриманих результатів перевірити методами статистичного аналізу. **Використовувані методи:** порівняльний аналіз відомих методів і алгоритмів шляхом статистичного моделювання, синтез нових алгоритмів і оцінка ефективності їх роботи шляхом проведення лабораторних натурних випробувань. **Отримані результати:** проведено порівняльний аналіз ефективності роботи відомих методів колірної корекції камер стереосистеми, запропоновані нові більш ефективні алгоритми для вирішення цього завдання. **Висновки.** Наукова новизна отриманих результатів: створено нові алгоритми корекції колірного балансу web-камер, використовуваних в стереоскопічних системах технічного зору, що відрізняються високою точністю колірної корекції і працюють в реальному масштабі часу з застосуванням функцій бібліотеки OpenCV в програмному середовищі Python.

Ключові слова: стереоскопічні системи технічного зору; колірна корекція зображень відеопотоку web-камер стереосистеми; спільна ректифікація і колірний баланс лівої і правої камери.

Метод цветовой коррекции web-камер стереопары и его практическая реализация

К. Ю. Дергачев, Л. А. Краснов, А. А. Челядин, А. В. Плахотный

Предмет изучения. В статье предложен новый метод цветовой коррекции web-камер в системах стереозрения для повышения качества их работы. **Целью** является сопоставительный анализ показателей качества известных методов коррекции цвета, разработка нового метода и рабочих алгоритмов совместной процедуры цветовой коррекции и ректификации фреймов видеопотока левой и правой камер. **Задачи.** Ставилась задача провести теоретический анализ показателей качества известных методов и алгоритмов коррекции цвета, разработать новые рабочие алгоритмы, программные коды этих которых написать на языке Python с использованием необходимых функций OpenCV. Провести экспериментальные исследования этих алгоритмов. Оценить эффективность работы стереосистемы в лабораторных условиях, а достоверность полученных результатов проверить методами статистического анализа. **Используемые методы:** сопоставительный анализ известных методов и алгоритмов путем статистического моделирования синтез новых алгоритмов и оценка эффективности их работы путем проведения лабораторных испытаний. **Полученные результаты:** проведен сопоставительный анализ эффективности работы известных методов цветовой коррекции камер стереосистемы, предложен новый более эффективный метод для решения этой задачи. **Выводы.** Научная новизна полученных результатов: создан новый метод коррекции цветового баланса web-камер, используемых в стереоскопических системах технического зрения, отличающийся высокой точностью цветовой коррекции и работают в реальном масштабе времени с применением функций библиотеки OpenCV в программной среде Python.

Ключевые слова: стереоскопические системы технического зрения; цветовая коррекция изображений видеопотока web-камер стереосистемы; совместная ректификация и цветовой баланс левой и правой камер.