Ujednačavanje osvjetljenja i boje u slikama dobivenim spajanjem slika s više različitih kamera i implementacija rješenja na realnu ADAS razvojnu platformu

Dhungana, Prakash

Master's thesis / Diplomski rad

2021

Degree Grantor / Ustanova koja je dodijelila akademski / stručni stupanj: **Josip Juraj Strossmayer University of Osijek, Faculty of Electrical Engineering, Computer Science and Information Technology Osijek / Sveučilište Josipa Jurja Strossmayera u Osijeku, Fakultet elektrotehnike, računarstva i informacijskih tehnologija Osijek**

Permanent link / Trajna poveznica: https://urn.nsk.hr/urn:nbn:hr:200:692691

JOSIP JURAJ STROSSMAYER UNIVERSITY OF OSIJEK
FACULTY OF ELECTRICAL ENGINEERING, COMPUTING AND INFORMATION TECHNOLOGY

GRADUATE STUDY

BRIGHTNESS AND COLOR EQUALIZATION IN IMAGES OBTAINED BY MERGING IMAGES FROM MULTIPLE DIFFERENT CAMERAS AND IMPLEMENTING SOLUTIONS ON A REALISTIC ADAS DEVELOPMENT PLATFORM

GRADUATION THESIS

PRAKASH DHUNGANA

OSIJEK, 2021

TABLE OF CONTENTS

1. INTRODUCTION

Vehicles play a significant role in daily activities around the world for commute and transportation. Passenger vehicles, as well as commercial vehicles, are crucial in today's world for humans and their daily tasks. The development of vehicular technologies began in the late 18th century, beginning with a vehicle powered with gas engines, till today, where a wide range of vehicular options like electric, hybrid electric, internal combustion-powered, fuel cell-powered vehicles, etc., are available. In the late 1900s, the advancement in the automotive industry is performed by the introduction of advanced electronics and software within the automotive systems. The involvement of powerful automotive electronics allows the concept of autonomous driving, which incorporates a wide range of software applications in the various automotive sector such as advanced driver assistance systems (ADAS), autonomous driving, connected transportation, etc. The concept of autonomous driving incorporates a solution for smart vehicles and provides a standardized platform for technological integration and advancement. Driving is the most significant task which should be performed in the vehicle. When humans drive the vehicle, he/she analyzes the vehicle environment through physical human senses and makes judgment for the action to be taken according to the perceived environment. Moreover, camera monitoring systems are mainly used to provide an overview of the surroundings to the driver. To assist drivers, various software and hardware modules are installed in the vehicle, enabling implementations of ADAS, which performs particular assistance to the driver like collision avoidance, parking assist, cruise control, lane departure warnings, etc. To provide the above-mentioned assistance algorithms, reliable presentation of surroundings and information is essential to achieve this goal.

In modern vehicles, the goal is to make the driving process as easy as possible for the driver and to make it safer for all traffic participants. One way to achieve this is to display the state of the vehicle's environment on a screen inside the vehicle. Moreover, to achieve that it is often necessary to combine images from multiple cameras into a single image because one camera does not cover the entire part of the space that should be displayed on the screen inside the vehicle. However, when combining the images from multiple cameras into a single image, a difference in brightness and color is often seen in a single panoramic image, due to the conditions in which the individual

images from a particular camera were taken. Therefore, it is necessary to equalize the brightness and color in the images obtained from multiple cameras, so that the unevenness of brightness and color is as low as possible. In this master thesis, four different methods were implemented for brightness and color equalization. Moreover, an evaluation of their performance is given using a subjective quality assessment. The solutions were implemented on a personal computer, and then on a realistic dedicated platform for ADAS systems. Optimization of the solutions was made through parallelization of certain subtasks on different processors of the ADAS development platform. Finally, performance in terms of execution speed and memory footprint was measured.

This thesis is organized as follows; in section II, a brief description of the state of the art is given. In section III, the methods for equalizations of brightness and color components are presented. In section IV, a description of the ADAS platform used for the implementation of the developed algorithm is provided. In section V, the algorithm for brightness and color equalization (BCE) is explained. Section VI presents the methods used for the validation and testing of the developed BCE algorithm. The summary of the thesis is presented in section VII.

2. LITERATURE REVIEW

In this section, a brief review of the problems, ideas, and solutions in the field of panoramic image formation, color balance, blending operations between two images in overlapping areas, and ADAS implementations of panoramic images is given.

A panorama image is a wide-angle presentation of a scene usually formed from stitching multiple images captured at a single location from different viewpoints, which share an overlapping region between two images. The obtained multiple captured images should be geometrically aligned and then processed with a stitching algorithm to form the final panoramic image. Geometric alignment algorithms transform the input images to generate an image with the transformed perspective that will serve as input to stitching algorithms. Stitching algorithms combine the images with transformed perspectives to form a panoramic image. The geometric alignment and stitching operations can be performed using various methods to obtain the panoramic image. As the panoramic image is formed by images from multiple cameras, there are variations in brightness and color within the panoramic image. In this chapter, previous researches and papers regarding contrast enhancements, panoramic image formation, and stitching operations for equalizing brightness and color are presented.

2.1. Previous Researches and Outcomes

Feature point identification and prioritization of such features according to their properties in an image is important to identify the overlapping parts of two images. After features are identified, mapping of these features between input images provides a homography matrix which is used to form a panoramic image. Speeded-Up Robust Features (SURF) [1] method is one of the available methods for feature point detection. SURF algorithm has three steps which are; points of interest detection, local neighborhood descriptors, and pairing of features having similar descriptors. For detecting features in an image, the Hessian matrix is used due to higher accuracy in the detection of features [1]. After the feature is obtained, a descriptor is assigned to it. For extracting features, square regions with 4x4 blocks are formed around the feature point. For each region, Haar wavelet responses are assigned for 16 regions around the feature point. SURF features can be matched by iterating through the detected features and comparing their descriptors.

SURF algorithm can be applied to create panoramic images as it is proposed in [2]. The proposed method also uses Scale Invariant Feature Transform (SIFT) features to create a panorama and compares the performance between these two features. For matching SURF features, K nearest neighbor (K-NN) and random sampling and consensus (RANSAC) algorithms are applied to remove incorrectly matched pairs. After appropriate matches are found, homography matric is calculated and the image is then transformed to match the perspective on the images being stitched. The authors have proposed blending between images to remove the seam formed due to differences in brightness and color in the formed panoramic images. Out of the 16 images used to create a single panoramic image, blending operations are carried in only 4 images that produced visible lines in the panoramic image. The testing results have shown that panoramic image formation with SURF was much faster than SIFT. The limitation is that the proposed method does not address the issue of difference in brightness and color created due to different photometric parameters of the camera, but rather applies blending operation in only four images in which there was a visible seam.

As demonstrated in [3], an improved gradual fusion algorithm is applied to achieve better results of the blending process. The method proposed by the authors focuses on improving the image stitching algorithm by eliminating the variations in brightness in the stitching process through an improved gradual fusion method. The gradual fusion algorithm assigns linear weights to the pixels in the overlapping region while the sum of the weights is always one. In an improved gradual fusion algorithm, the weights assigned to the pixels are non-linear and the sum of weights is equal to one. The operation of the improved gradual fusion algorithm is compared with the gradual fusion algorithm and the experiments were carried out to find which method provides an effective and smooth transition between the images in a panorama. The result of the comparison showed brightness and color transition in the resulting panoramic image formed using an improved gradual fusion algorithm was better than the gradual fusion algorithm.

There are various methods for brightness and color equalizing within an image such as histogram equalization, histogram specification (normalization), contrast enhancement, color balance, and gamma corrections. Histogram equalization is one of the first methods to be implemented for improving the brightness and contrast of an image. Most of the histogram equalization and specification methods increase the brightness as well as the contrast of the image. These methods are suited for grayscale images and have been implemented in a single-color space.

histogram equalization (HE) flattens the histogram in images with accumulated or skewed histograms. HE is performed through an intensity mapping function to ensure the resulting histogram to be flat. In doing so, the darker pixel values are increased and the brighter pixels values are decreased, which makes the pixel values to be evenly dispersed in the histogram [4]. There are versions of HE methods which can be chosen according to the required performance such as histogram specification, block histogram equalization, adaptive histogram equalization (AHE), and contrast limiting adaptive histogram equalization (CLAHE). Color balance is a method to perform color correction in an attempt to make the distorted white pixels closer to white. The simplest way to perform color balance is to apply diagonal matrix transformation to RGB values in an image.

As seen in [5], various linear and nonlinear contrast enhancement techniques are used to compare the performance of the enhancement of brightness of grayscale images. For comparing linear contrast enhancement methods, the authors have used minimum-maximum contrast stretching, percentage contrast stretching, and piecewise contrast stretching methods. For comparing non-linear contrast enhancement methods, the authors have used HE, AHE, homomorphic filter (low and high pass), and unsharp mask. The comparison presented, piecewise contrast stretching methods and unsharp mask as the most effective methods for contrast enhancement using linear and non-linear methods respectively [5].

Contrast enhancement techniques can be performed in the pixel domain as is shown in [6]. Pixel domain contrast enhancement is widely used in real-time applications due to low requirements in computation and parameters [6]. The proposed method uses two concepts for contrast enhancement by classifying the input image as bright and dimmed. For bright images, adaptive gamma correction (AGC) via negative image and for dimmed images, this paper proposes AGC via truncated CDF is proposed. A negative image is an image generated by replacing the pixel value by the maximum value that a pixel can be assigned subtracted by the actual pixel value (e.g., a negative image is generated using $I = 255 - I$, where I is an image presented by 8 bits). In the first method for bright images, a negative image is obtained. The performance of the methods proposed in [6] showed better performance of brightness and contrast enhancement than HE, AHE, AGCWD, and SECE methods. The method proposed by the authors focuses on a single image but the concept of gamma correction used in this method is used to develop the gamma correction method for brightness equalization.

Color balance can be used for panoramic images to equalize brightness and color in panoramic images as it is shown in [7]. The color correction is carried out in HSV color space. The reference image is chosen as a basis to match the color profile of the target image. The method used in this paper simply adds the difference in brightness pixel values between input images to the target picture if the pixel value of brightness is lower than the pixel value of brightness in the reference image otherwise the difference is subtracted. The image is then stitched using the gradual fusion method to obtain a panoramic image. The result of the method was improved color transition between the stitched images but still, the natural-looking panorama was not achieved.

There are well-developed panoramic image stitching systems for mobile devices and PCs that can perform reliable image stitching. Those implementations are not suitable for implementation in automotive applications due to limitations in computational resources. There are stitching applications in areas of blind spot eliminations using a stitched image from camera mirror replacement using two cameras on each side [8]. For blind spot elimination, the authors in [9] have also proposed a rear stitched panorama view obtained using four cameras (left, right, and two stereo cameras) to obtain an even wider range of view. Image stitching based on SURF features can be used to generate a wider view panoramic image to eliminate blind spots at the rear end of the car [8]. For this purpose, two cameras positioned at left and right-side mirrors were used. The images were then processed through algorithms for feature point detection, feature matching, and image stitching. The tests were carried out and effective creation of a panorama image was achieved with an 88.56% success rate [8]. Rear stitched view panorama (RSVP) improves on previous research by implementing three cameras to obtain a panoramic image which can increase the field of view of the resulting panoramic image [9].

2.2. Problem Statement

In the above-mentioned works, the brightness and color components are processed with the same algorithm. Moreover, in some methods, grayscale images are used for contrast enhancements. Since even a slight variation of the brightness component in an image is easily identified in the comparison with the same level of differences in color components, there should be different methods for equalization of brightness and color. Also, in normal camera imaging systems increase in brightness results in lower saturation and vice versa. The goal of this master thesis is to equalize

brightness and color components in an image through the implementation of different brightness and color equalization algorithms while using brightness as the base for the equalization of color components. The proposed solution for the equalization of brightness and color components is explained in detail in the next chapter.

3. PROPOSED METHOD FOR BRIGHTNESS AND COLOR EQUALIZATION

In this section, the ideas on how equalizations of brightness and color components can be achieved are discussed. In particular, brightness equalization can be achieved through linear and non-linear methods. Similarly, color components can be compensated through the adaptation of the proposed method for brightness equalization. The brightness and color components used for equalizations are in YUV color space, where Y is luminance and U and V are chrominance components. Luminance (Y) is used for equalization of brightness while chrominance (U and V) are used to equalize color. The linear and non-linear methods for brightness and color components can be applied in combination which results in four different methods in total. The methods proposed in [4] and [6] are used as a basis for the brightness and color equalization of two images to form a panoramic image in this thesis. The inverse gamma had to be slightly adjusted to fit the purpose of color correction.

Most of the state-of-the-art methods given in the literature address brightness and color enhancement through HE and its derivative approaches. Also, equalizations based on color balance and gamma correction were proposed for brightness equalization. In a panoramic image, color balance and blending operations are mainly used to obtain a smooth transition between stitched input images. The goal of this master thesis is to implement brightness equalization and color equalization methods and then to apply an appropriate blending algorithm to obtain a final natural-looking panoramic image. In doing so, the whole image that has a lower mean of brightness in the overlapping region will be transformed completely to match the brightness and color levels of the image having a higher mean of brightness in the overlapping region and blended to form a panoramic image. After the methods for brightness and color equalization have been developed, their implementation in ADAS hardware is carried out. In this section, the methods proposed to carry out brightness and color correction of one image based on the reference image and their blending are described. For brightness equalization, gamma correction using inverse gamma function is applied. Similarly, for color components, an adaptation of gamma correction to fit the color profile according to YUV color space is applied. Finally, to obtain a panoramic image improved gradual fusion as demonstrated in [3] is applied. The description of the methods for brightness and color equalization is explained with equations in this chapter.

3.1. Brightness Equalization

Brightness equalization can be described as the process of matching the brightness level of an image with the brightness level of the reference image. The brightness of an image can be given in various color spaces. In the red, green, and blue (RGB) image, the grayscale image obtained from the RGB image represents the brightness. In the hue, saturation, and value (HSV) image, V represents the brightness and S represents the saturation level/color of the image. In L*a*b* image (Perceptual brightness and Color), L represents the perceptual brightness, and a* and b* represent the color. In YUV colorspace, Y represents the luminance (brightness) and U(Cb), and V(Cr) represents the Chrominance (color) of an image. In this master thesis, YUV (YCbCr) color space is chosen to manipulate the brightness and color pixel values using different methods to obtain brightness and color correction. In a panoramic image, one of the input images that will be used to form a panoramic image is regarded as the reference image, while the rest of the images are passed through the brightness equalization process. Brightness equalization can be achieved through simple linear operations or non-linear operations of the brightness level values. Linear operations include additions, subtraction, multiplication, or a combination of mentioned operations. Through a linear approach, the brightness value of an image is treated equally irrespective of its value i.e., similar manipulation to lower and higher pixel brightness values are performed. Non-linear operations use polynomial functions of order other than one. In previous researches, gamma correction using inverse gamma is mostly used among non-linear methods while a color balance is used among linear methods. In the non-linear approach, the degree of change in brightness value depends on the original pixel brightness levels. This helps to increase the lower brightness and higher brightness values differently using higher changes for lower pixel values and lower changes for higher pixel values. The methods used in this thesis are based on linear and non-linear operations. In a linear method, the combination of multiplication and addition is used for the calculation of new pixel values of brightness and color whereas, in the non-linear method, gamma correction is applied for the calculation of a new level of brightness and color. The linear method proposed is also known as the basic linear transform method, while gamma correction based on inverse gamma function is also known as gamma correction [4].

3.1.1. Basic Linear Transform

The basic linear transform method is used as a combination of multiplication and addition operations on the pixel values to manipulate the brightness level of the image. This method is also known as brightness and contrast enhancement. This method has two operations combined in one while the parameter used for multiplication and addition determines the degree of change in the brightness and contrast of the image. The parameter for multiplication is often called a gain and for addition is called a bias parameter [4]. The relationship of the original and final brightness pixel value is expressed as shown in equation 3.1 [4].

$$g(i,j) = \alpha \times f(i,j) + \beta \tag{3.1}$$

where (i,j) represents values of the image pixel coordinates, $g(i,j)$ represents the output image pixel values for brightness, $f(i,j)$ represents the input image pixel values for brightness, α represents the gain parameter and β represents the bias parameter. The values of α and β determine the level of brightness change between the input and output image. There is no prior method for the calculation of gain and bias parameters for basic linear transform for correcting brightness and color components. Since these parameters dictate the degree of change in brightness levels, these parameters should also be determined by the brightness levels of the input images in a panorama. The gain and bias parameters are determined by the calculation of the mean brightness level of the left (referent) and right input images in the overlapping region. One of the examples of input images is shown in figure 1. Once the mean of the input images is calculated, the ratio of brightness of the darker image to brighter image is taken which is always less than one and is used as gamma (γ).

Figure 1: Sample of input images used for brightness and color equalization (a – left reference image, b – right transformed image to match the perspective of referent image)

The gain and bias parameters are calculated as shown in equations 3.2 to 3.4.

$$\gamma = (\overline{Y_{dark}} / \overline{Y_{bright}}) \tag{3.2}$$

$$\alpha = 2 - \gamma \tag{3.3}$$

$$\beta = \gamma\left(\overline{Y_{bright}} - \overline{Y_{dark}}\right)/2 \tag{3.4}$$

where $\overline{Y_{dark}}$ represents the mean value of brightness of the darker image in the overlapping area, $\overline{Y_{bright}}$ represents the mean value of brightness of the brighter image in the overlapping area, γ represents ratio gamma for brightness correction and determination of gain and bias parameters, α represents the gain parameter and β represents the bias parameter.

Figure 2: Brightness equalization of the darker image from figure 1 with gain equal to 1.413963 and bias equal to 13.771867 (a – Original image, b – Brightness equalized image using basic linear transformation)

Figure 2 presents an example of brightness correction of the darker image using basic linear transform with images from figure 1 as input images.

3.1.2. Gamma Correction

Gamma correction is one of the non-linear methods for brightness equalization in a panoramic image. The most significant parameter for gamma correction is the value of gamma. The value of gamma determines the increase and decrease in the brightness levels of the image. If the value of gamma is greater than 1, the brightness of the image decreases whereas the values less than one

increase the brightness levels. The relation between the original and final brightness level is shown in equation 3.5 [4].

$$g(i,j) = (f(i,j)/255)^{\gamma} \times 255 \tag{3.5}$$

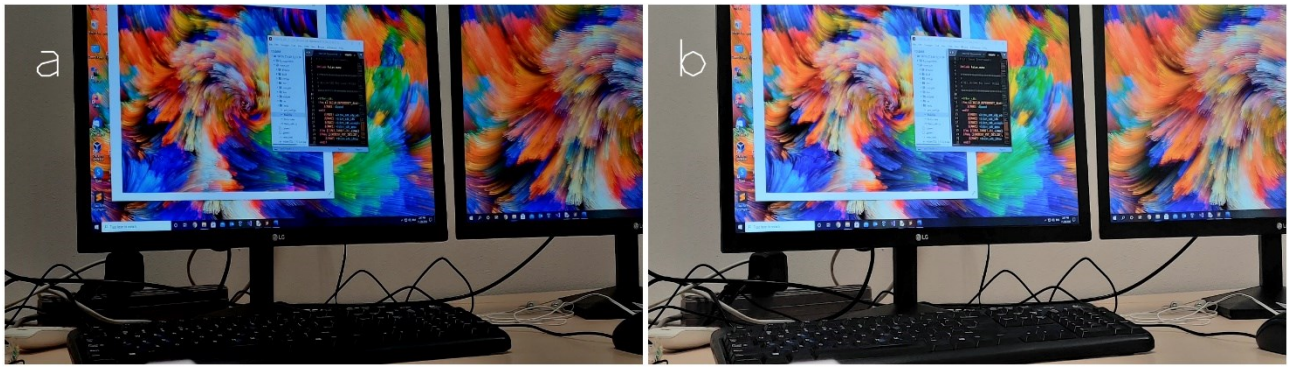


Figure 3: Brightness equalization of the darker image from figure 1 with gamma equal to 0.586037, (a – Original image, b – Brightness equalized image using gamma correction)

Figure 3 presents an example of brightness correction of the image using gamma correction with images from figure 1 as input images.

3.2. Color Equalization

Color equalization is carried out to match the saturation level and the color profile of the input images with the reference image while forming a panoramic image. Mostly color balance is used to equalize color as well as brightness to enable a smooth transition between image borders in a panoramic image. Color balance is implemented by the multiplication of the pixel with a diagonal matrix to obtain a transformed color value. Along with gain, addition can also be applied to obtain transformed color similar to basic linear transform which is used in color correction using basic linear transform. Also, color can be equalized using gamma correction similar to brightness in case of higher differences in saturation. The methods used in this master thesis are adapted from the above-mentioned linear and non-linear methods.

3.2.1. Color Correction using Basic Linear transform

If color correction using basic linear transform is carried with similar coefficients as brightness, the resulting pixel values of color components result in a distorted image. Calculations of gain, bias,

and gamma parameters are based on the brightness difference between the adjacent images in the overlapping region. So, to enable basic linear transform to fit for use in color components, the coefficients are compensated accordingly. For calculation of gamma for color components, compensation factor (δ) is calculated based on the original gamma for brightness. The calculation of gamma for color components, gain and bias parameters are shown in equations 3.6 to 3.8.

$$\delta = 1 + (1 - \gamma)/2 \tag{3.6}$$

$$\gamma_c = 1/\delta \tag{3.7}$$

$$\alpha_c = 1.3 - ((\gamma_c \times 0.3)/0.85) \tag{3.8}$$

$$\beta_c = \gamma_c(\overline{Y_{bright}} - \overline{Y_{dark}})/10 \tag{3.9}$$

where δ is the compensation factor for calculation of gamma for color component, γ_c is gamma for color components, α_c represents the gain parameter for color component, and β_c represents the bias parameter for the color component.

After calculation of the coefficients required for basic linear transform for color equalizations, the final color components can be calculated using equations 3.10 and 3.11.

$$g(i,j) = \alpha_c \times (f(i,j) - 128) + \beta_c + 128 \tag{3.10}$$

$$g(i,j) = 128 - \alpha_c \times (128 - f(i,j)) - \beta_c \tag{3.11}$$

Equations 3.10 and 3.11 complement each other. In an image, chrominance values vary from 0 to 255 but the values from 0 to 128 and 128 to 255 have significant implications in the resulting color and saturation level of the output image. Thus, the chrominance spectrum is divided into two halves with 128 being the midpoint. If the pixel value of color is greater than 128, then equation 3.10 is used to calculate the final pixel value otherwise equation 3.11 is used. Finally, the final values are clipped between 0 to 255 to prevent the overflow of colors.

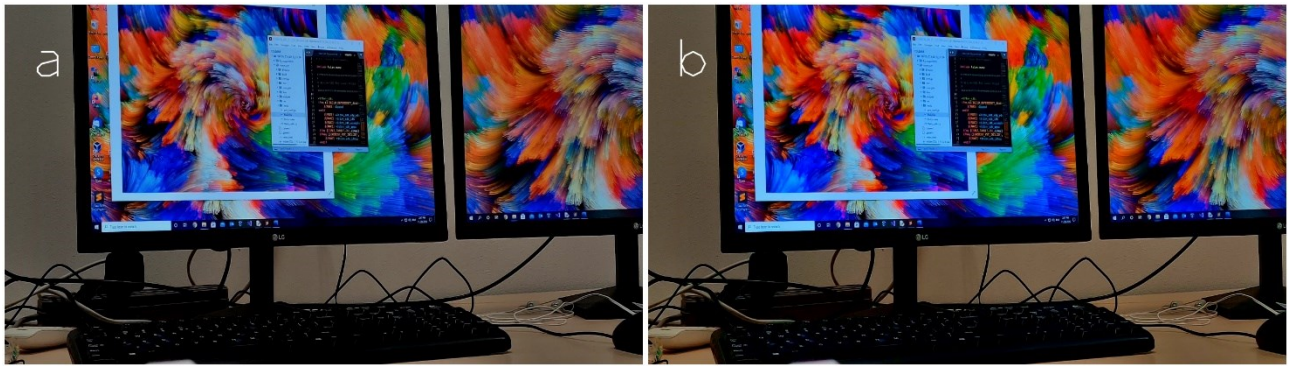


Figure 4: Color equalization of the darker image from figure 1 with gain equal to 1.207584 and bias equal to 3.894011, (a – Original image, b – Color equalized image using basic linear transformation)

Figure 4 presents an example of color equalization of the image using basic linear transform with images from figure 1 as input images.

3.2.2. Color Correction using Gamma Correction

Similar to color correction using basic linear transform, gamma correction can also be applied to equalize color in case of differences in color components. The correction of color components in this method is performed in a non-linear manner, which enhances the saturation level and provides a visually pleasant panoramic image. The gamma used for brightness has to be compensated according to equations 3.6 and 3.7 to obtain the final gamma value for color components. After the gamma parameter is calculated the new pixel value of color can be calculated using equations 3.12 and 3.13.

$$g(i,j) = 128 + (\,(f(i,j) - 128)/128)^{\gamma_c} \times 128 \tag{3.12}$$

$$g(i,j) = 128 - (\,128 - (f(i,j))/128)^{\gamma_c} \times 128 \tag{3.13}$$

As mentioned earlier, chrominance values from 0 to 128 and 128 to 255 have significant implications in the resulting color and saturation level of the output image. If the pixel value of color is greater than 128, then equation 3.12 is used to calculate the final pixel value otherwise equation 3.13 is used. Finally, the final values are clipped between 0 to 255 to prevent the overflow of colors.

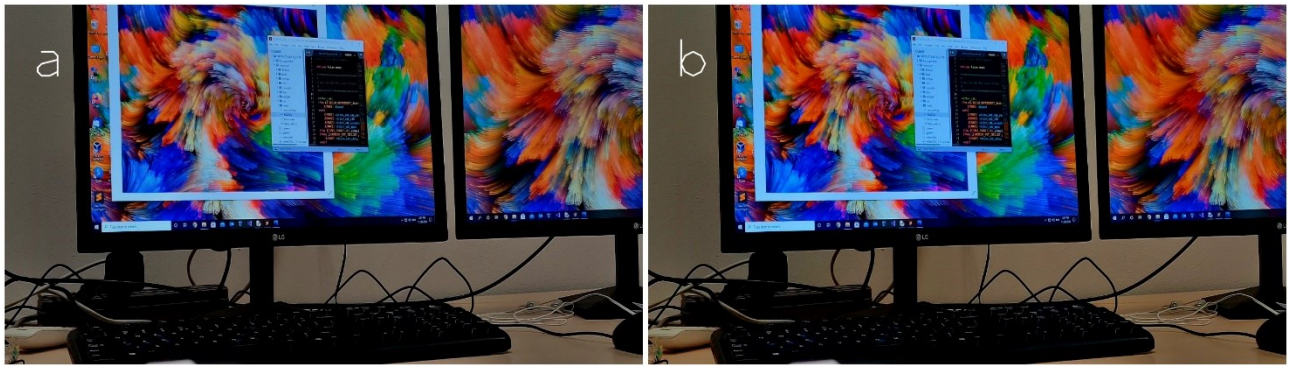


Figure 5: Color equalization of the darker image from figure 1 with gamma equal to 0.828513, (a – Original image, b – Color equalized image using gamma correction)

Figure 5 presents an example of color equalization of the image using gamma correction with images from figure 1 as input images.

3.3. Image Blending

A blending operation has to be carried out to obtain a panoramic image from input images in overlapping areas. Among blending operations mentioned in the literature review chapter, an improved gradual fusion method is chosen to obtain the final panoramic image after brightness and color equalizations. The comparison between the gradual fusion method and the improved gradual fusion method suggests that the improved gradual fusion method performed better [3]. The weights distribution in gradual and improved gradual fusion algorithm is shown in Figure 6.

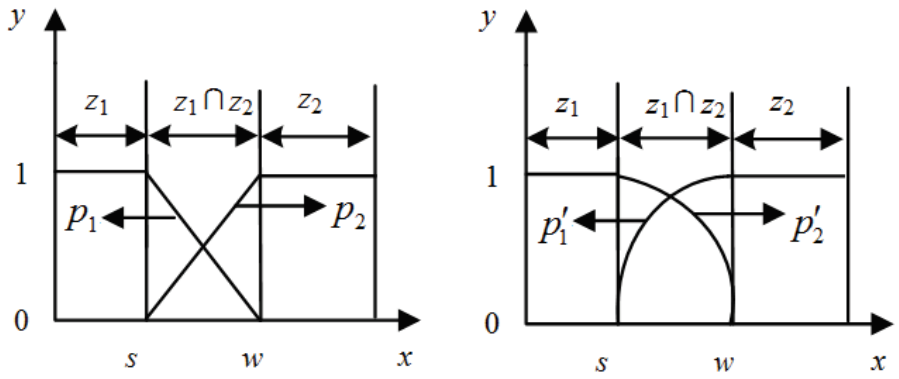


Figure 6: Left - Gradual Fusion Algorithm weight distribution in overlapping region, Right - Improved Gradual Fusion Algorithm weight distribution in the overlapping region. [3]

In the given figure, s denotes the beginning of the overlapping region, w denotes the width of the overlapping region, x represents the pixel position, $z_1$and z_2 represents the image pixel value for brightness for the input images, p_1 and p_2 are the weights of images $z_1$and z_2 for gradual fusion algorithm and $p_1{}'$ and $p_2{}'$ are the weights of images z_1 and z_2 for improved gradual fusion algorithm. The weights for pixel and final pixel values are calculated using equations 3.14 and 3.15 respectively.

$$p_1{}' = (w^2 - x^2)/w^2 \tag{3.14}$$

$$g(i,j) = l(i,j) \times p_1{}' + r(i,j) \times (1 - p_1{}') \tag{3.15}$$

where $l(i,j)$ represents the input left image pixel values, $r(i,j)$ represents the input right image pixel values, w denotes the width of the overlapping region, x represents the pixel position, $p_1{}'$ is left pixel weight parameter, and $(1-p_1{}')$ is the right pixel weight parameter.

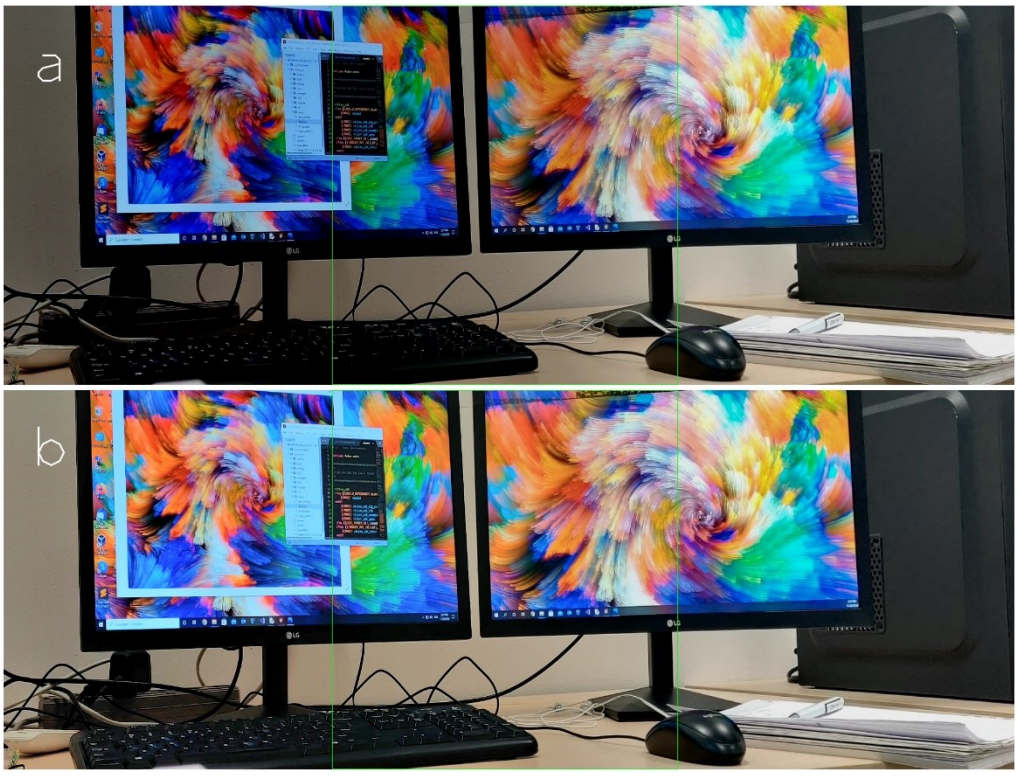


Figure 7: Comparison of the panoramic images with and without BCE equalization (a – Panoramic image formation without BCE equalizations, b – Panoramic image formation with BCE equalizations)

Figure 7 presents an example of blending operation before and after brightness and color equalization with images from figure 1 as input images. The area of blending is highlighted using green lines.

4. ADAS HARDWARE AND SOFTWARE DEVELOPMENT KIT

The Hardware and Software Development Kit for ADAS section describes the tools and technologies used in the development and implementation of the proposed algorithms on a real ADAS platform. Descriptions about hardware specifications of the ADAS development board regarding System on Chip (SoC) resources, different processors used, memory available to processors are provided in this chapter.

Advancement in the field of automotive electronics and software has caused the development of various ADAS platforms. Such ADAS systems require powerful hardware that can process information and execute complex algorithms. This leads to several platforms, specific to the application requirement because no ADAS platform supports all ADAS systems [10]. Therefore, the implementation of multiple ADAS platforms in a single-vehicle solves the issue related to system performance but increases the complexity of selection among existing ADAS platforms. The table below presents an overview of available autonomous vehicle hardware stacks available for ADAS systems. For this thesis, the ADAS platform designed and developed by RT-RK and TI is used for the development and optimization of brightness and color equalization.

Table 1: Autonomous Vehicle Hardware [10]

Company	Platform	Focus	Platform Type	Automation
Audi	zFAS	Piloted Driving	Production	L3-L5
Nvidia	DrivePX2	AI	Production/Development	L3-L5
TTTech	TTA Drive	Sensor	Development	L1-L5
Renesans	R-Car HAD Solution Kit	Vision	Development	L1-L2
Mentor	DRS360	Sensor	Development	L1-L5
Bosch	AI Car Computer	AI	Production	L3-L5
NXP	Blue Box	Vision, AI	Development	L1-L5
TI/RT-RK	AMV ALPHA	Vision	Development	L1-L2
Delphi	CSLP	Vision	Production	L1-L5
Intel	GO AV	Piloted Driving	Development	L3-L5

AMV ALPHA Board is an ADAS development board designed by the company RT-RK. The board is powered by TI TDA2xx SoCs and features various peripherals. Along with the board, VisionSDK is used as the software of choice for development purposes. This is a computer vision-oriented software package developed by Texas Instruments and further patched by RT-RK to make it compatible with the Alpha board out of the box. The details and descriptions regarding the AMV ALPHA board and VisionSDK are described in this chapter.

4.1. AMV ALPHA Board

AMV ALPHA is a reference design board based on Texas Instruments SoCs supporting basic and advanced warning systems, active control systems, and semi-autonomous operations for ADAS systems and automotive applications. A set of targeted applications covers different driver assistance algorithms as front view, surround view, a camera mirror system, driver monitoring, and night vision, with various features for increased comfort and overall driving experience. The hardware specification of the AMV ALPHA board is:

a) Processors: 3 TDA2x SoC
b) Load balancing between TDA2x SoCs via PCIe enabling optimal workload
c) Each SoC has HDMI out, DCAN, Ethernet, JTAG, UART interface, and Microsoft SD Card slot
d) Supports up to 10 cameras (Automotive cameras compatible with TI DS90UB964 and DS90UB914A deserializers)
e) Support for AVB Ethernet
f) The system is upgradable with daughterboard over connector where all TDA2x interfaces are available (Ethernet, USB, Can, SPI)

The TDA2x SoC incorporates a heterogeneous, scalable architecture that includes a mix of TI's fixed and floating-point TMS320C66x DSP generation cores, Vision Acceleration Pac (EVE), ARM Cortex-A15 MPCore™, and dual-Cortex-M4 processors. Application for front-camera, surround view and fusion algorithms are achieved through a dedicated set of heterogeneous processors, brought together in a scalable architecture with a rich set of integrated peripherals, providing an optimal mix of performance in a low power footprint for ADAS vision processing [11]. The appearance of the AMV ALPHA ADAS development board is shown in Figure 8.

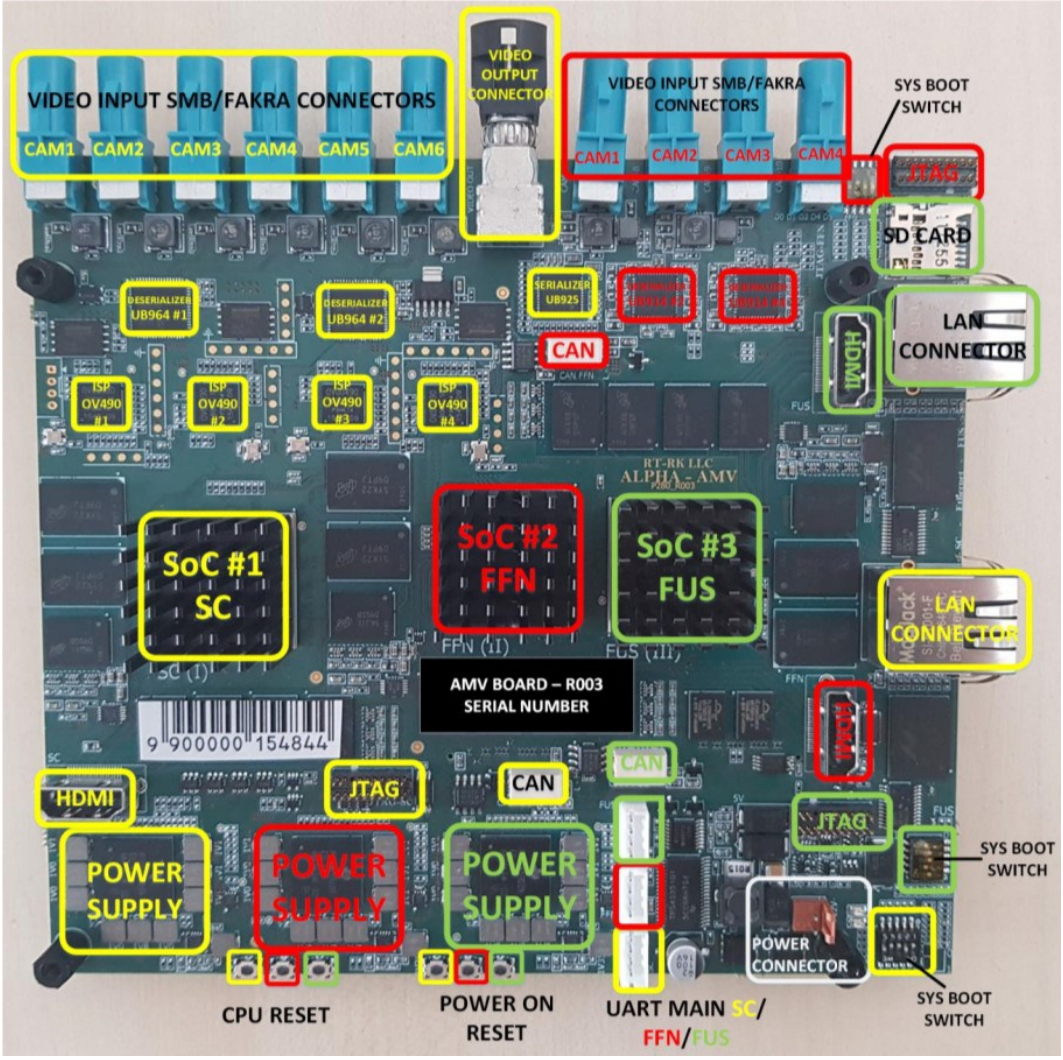


Figure 8: AMV ALPHA ADAS Development Platform Board Layout *[12]*

There are three SoCs namely SCV, FFN, and FUS in AMV ALPHA Board. Each of these SoCs has 2 ARM Cortex-A15, 2 ARM dual-Cortex-M4, 2 C66x DSP cores, and 4 EVE cores [10]. The first SoC is applied for Surround Camera View (SCV) applications. The second SoC is applied for Front View Camera Near Angle Stereoscopic View, Front View Camera Wide Angle, Night Vision Camera (FFN) algorithms. The last SoC is used for fusion (FUS). To distribute resources for data processing there is the possibility of load balancing between SCV and FFN SoC's using Peripheral Component Interconnect Express (PCIe) and Video link between FFN and FUS SoCs. Each of the SoCs has the following resources and peripherals available:

a) 2x ARM Cortex-A15 CPU for using network functionalities.

b) 2x ARM Cortex-M4 CPU for the general-purpose used most for use-cases and links.
c) 2x C66x DSP CPU for high-performance image processing.
d) 2x EVE CPU for vision-related purposes
e) 1.5 GB of RAM
f) Micro SD card slot for running a compiled image, reading data from the card, or writing.
g) HDMI port for showing video output on a screen.
h) UART port for communicating over a terminal on a PC.
i) JTAG port for connecting with the debug probe.
j) The boot mode switch is used to adjust which mode the SoC will boot in.
k) PCIe connection to other SoCs for sending data between SoCs.

There are few differences among these SoCs that determine specific application purposes of the SoC. SCV is primarily used for surround view, or use-cases that require a larger number of cameras. Cameras with labels RDACM24B-01 (fisheye) and RDACM24B-06 (narrow field) can be used on this SoC. This SoC has ethernet support as well. FFN is a general-purpose SoC used for use-cases for front view with narrow and wide FOV and night visions. A camera with the label RDACM23-06 (narrow field) can be used in this SoC. FUS is used mostly for fusion applications that enable data sharing from different SoCs. FUS does not have support for cameras and has ethernet to support network functionalities. When running the ALPHA board, three boot modes can be changed using the switches. Each SoC has its switches for boot mode configurations which are shown in Figure 9.

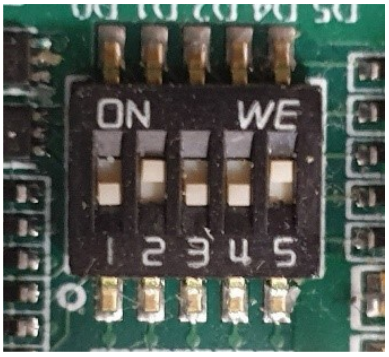


Figure 9: Boot Mode Configuration Switch

The available boot modes are mentioned in Table 2.

Table 2: Boot Configuration Switch States

Boot Mode	Switch Configuration
JTAG (Debug)	00000
SD Card	01001
QSPI	11111

The relationship and interactions between SoCs and peripherals are demonstrated in Figure 10.

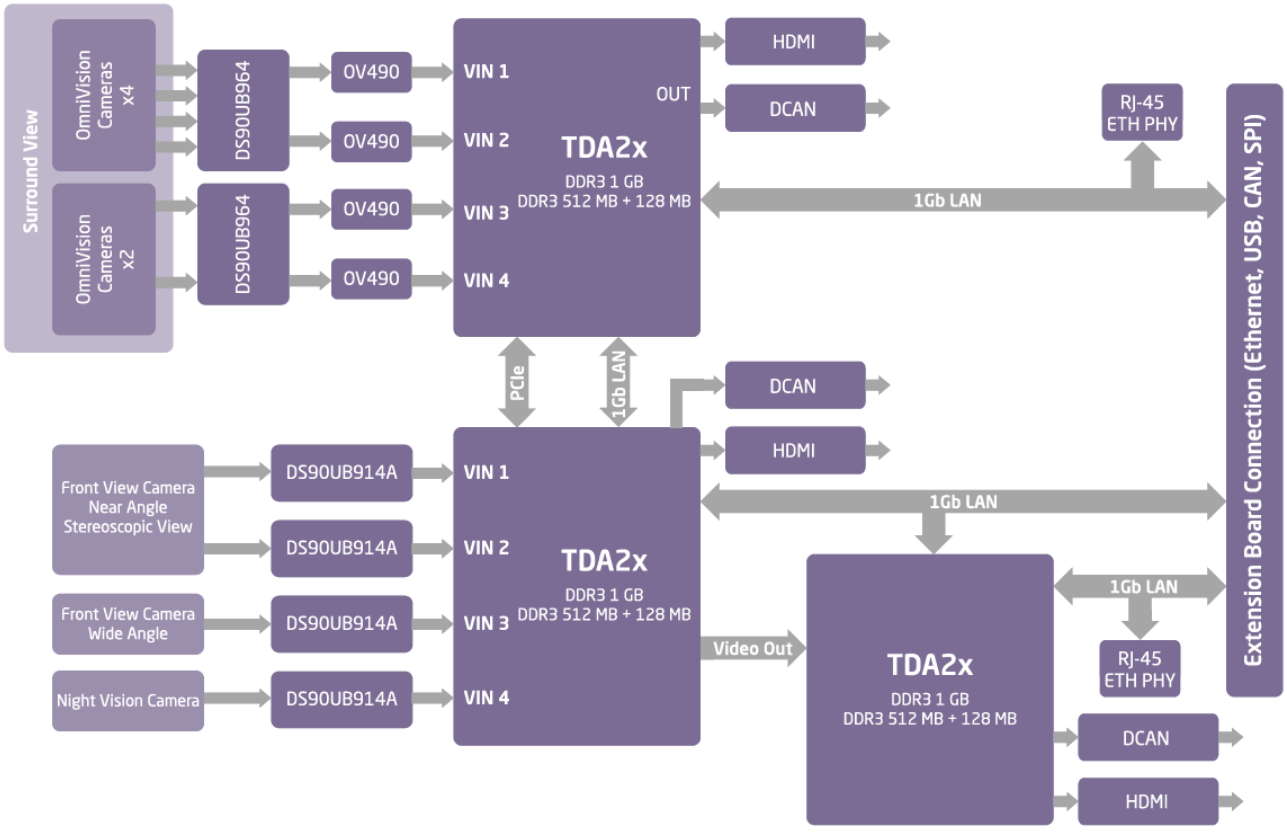


Figure 10: Block Diagram Showing interactions between SoCs [13]

4.2. Vision SDK

VisionSDK is used as the software for the development and implementation of algorithms. This is a computer vision-oriented software package developed by Texas Instruments and further patched by RT-RK to make it compatible with the ALPHA board out of the box. VisionSDK is a multi-processor software development platform designed for ADAS SoCs developed by TI. This software framework enables the creation of different ADAS application data flows involving video capture, video pre-processing, video analytics algorithms, and video display. VisionSDK is based on a framework named the "Links and Chains" framework while the user API for this framework is called "Link API". The SDK installer package includes all tools and components necessary to build such applications, including code generation tools, BIOS, BSP drivers, networking stacks, codecs, and kernels.

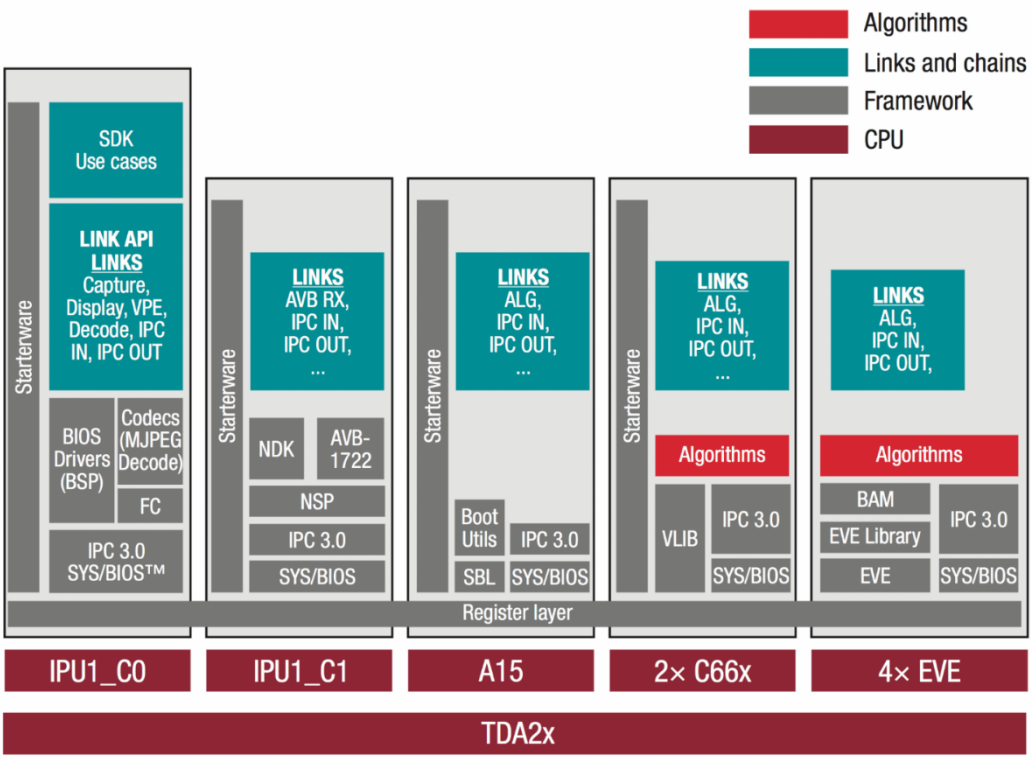


Figure 11: Vision SDK Software Architecture [14]

AppImage built using VisionSDK is run on the Alpha Board by placing the two built application components on the SD card, inserting them into the corresponding SD card slot on the board, and providing power to the board. In this master thesis, VisionSDK is used to develop a new

use-case application and a new algorithm link. After the required source files for use-cases and algorithms are added, the VisionSDK image is built by following steps.

Step1: Establishing dependencies between make files

The make files are processed using the command “gmake -s -j depend” which connects all the make files. It is necessary to run this command if any changes in the make file have been done. This enables switching between SoCs, adding new use-cases and algorithms.

Step2: Creating MLO

MLO file has to be generated that is copied along with AppImage to SD card when booting ALPHA board. This has to be generated only once. The MLO file is generated using the command “gmake -s -j sbl_sd”

Step3: Building object and binary files

This is the main build process. This build takes the longest time and performs the actual build process. The build process is carried out using the command “gmake -s -j”. This command has to run every time if any modifications have been carried out in any of the files.

Step4: Creating Application Image

AppImage is required to boot the ALPHA board using an SD card along with MLO. AppImage can be generated using the command “gmake -s -j appimage”. Similar to the build process, this command has to run every time if any modifications have been carried out in any of the files.

After all the build process is completed, MLO and AppImage are copied to the SD card, and the ALPHA board is booted to run the use-case and the algorithm generated.

5. ALGORITHM DEVELOPMENT AND IMPLEMENTATION

5.1. Objectives

Since images from multiple cameras are combined into a single image, the difference in brightness and color is often seen in a single panoramic image, due to the varying conditions in which the image from a single camera was taken. As part of this thesis, a solution to equalize the brightness and color on the images obtained by merging images from multiple cameras is proposed, so that the differences in brightness and color are eliminated. The specific objectives of the project are as follows:

a. Develop an algorithm for brightness and color equalization that can be applied separately.
b. Implement the designed algorithm in real embedded hardware used in automotive applications.
c. Optimize the algorithm to ensure real-life applications.

For methods for brightness and color equalizations on the ALPHA board, algorithm links were added using VisionSDK. Initially, the method for comparing the brightness level of input images to determine the darker image is performed. For this purpose, YUV color space is chosen as it is supported by the ALPHA board and brightness (Y) levels of the input images are taken as the reference for further calculations. To determine the darker image, the overlapping regions of the images are identified and the average brightness of a single image in the overlapping region is calculated. The image with a lower average brightness is the darker image. For the determination of overlapping regions, one of the input images has to be transformed to fit the perspective of the other image. After the image has been transformed, an iterative search of individual rows is carried out to locate the pixel index of the starting point of the overlapping region. The lowest pixel index value from all the rows is determined as the starting point of the overlapping region (represented by the tip of the blue arrow in figure 12b) and the endpoint of the overlapping region is the width of the reference image that is used for perspective transformation of the first image. An example of the reference image and the transformed image is shown in figure 12. In these input images, the right image has lower brightness and the BCE equalization is performed in the right image before forming the resulting panoramic image.

Figure 12: Sample input images for BCE algorithm to form a panoramic image

The algorithm for the calculation of starting point of overlapping area is called CALCULATEBLENDINGPOINT. Based on these calculated values, blending weights are calculated according to pixel position according to equations 3.14 and 3.15. The algorithm for the pixels weights calculation of the referent and transformed image for blending in the overlapping area is called CALCULATEBLENDINGWEIGHTS. Finally, an algorithm is devised to perform brightness equalization and color correction called YUVCORRECTION. After correction, the images are blended to obtain a single panoramic image with uniform brightness and color levels which is implemented in algorithm BLENDIMAGES. After BCE processes were completed, the implementation of the BCE algorithm was done in the ALPHA board and generation of the final panoramic image was carried out. The initial implementation of the BCE algorithms is illustrated in algorithm 1 and its flowchart is shown in figure 13.

Algorithm 1: Brightness and Color Equalization (BCE) Algorithm

Algorithm 1: BRIGHTNESS AND COLOR EQUALIZATION (BCE)		
Input	:	Reference image (first frame), transformed the image of the second frame according to the homography matrix calculated to match the perspective of the first frame (YUV format – YUV420 [NV12])
Output	:	Brightness and Color corrected stitched panoramic image of input frames
begin		
1		Calculate starting point for overlapping region: **CALCULATEBLENDINGPOINT** (RI,w,h)
2		Perform YUV correction process on darker image based on the mean of the Y channel of the overlapping region: **YUVCORRECTION** (LI,lw,lh, RI,rw,rh,sbp)
3		Calculate weights for blending from starting point of the overlapping region to the width of the reference frame: **CALCULATEBLENDINGWEIGHTS**(s,e)
4		Blend YUV corrected images according to the weights calculated: **BLENDIMAGES** (LI, RI, OI,sbp,lw,lh, RI,rw,rh)
5		Copy right image beyond the overlapping region and save output panoramic image
end		

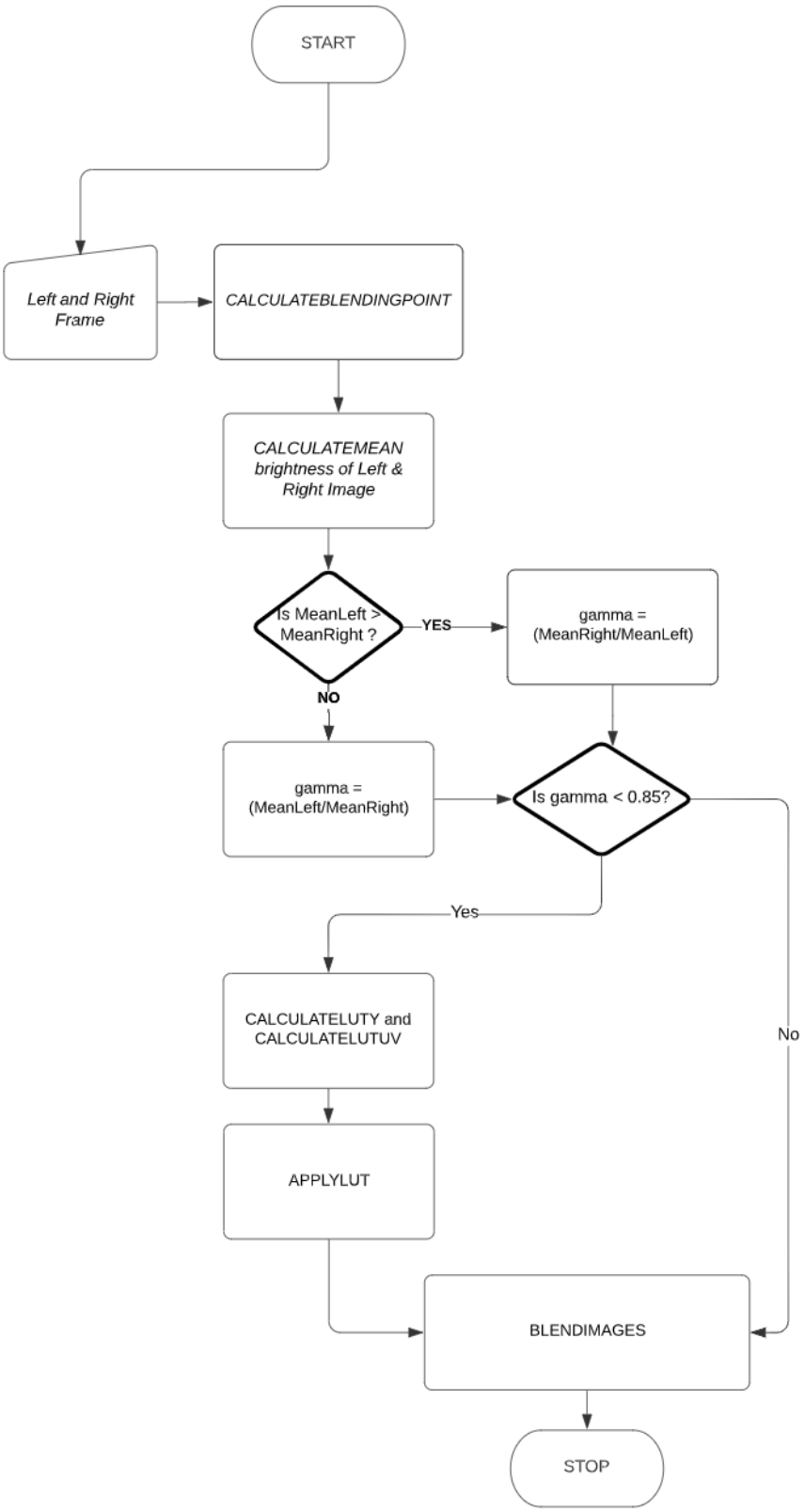


Figure 13: Flowchart of BCE Algorithm

5.2. Algorithm for Determination of Overlapping Region

The goal of the determination of the overlapping region algorithm is to find the overlapping regions. The input for this algorithm is the transformed image which matches the perspective of the reference image. The left image among the input images is taken as reference and the right image is transformed to match the perspective of the first image. The transformed image has black pixels in the left region and this fact is used for the determination of starting point of the overlapping region. This part of the algorithm has been carried out only once at the beginning of the implementation of the BCE algorithm or whenever the camera positions are altered.

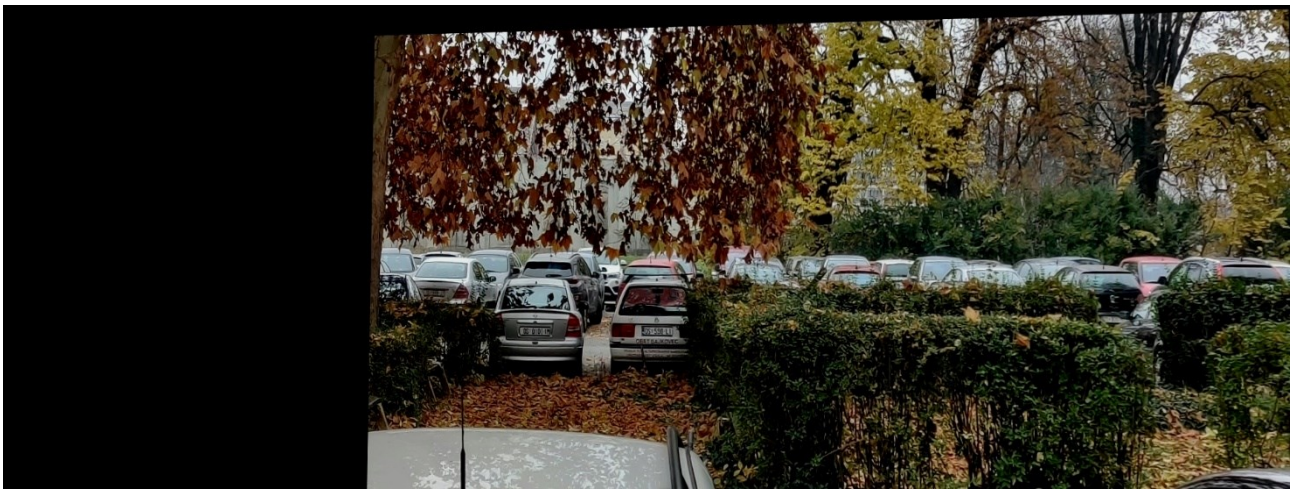


Figure 14: The transformed image of the right frame to match perspective to the left reference image

Towards the left side of the transformed image, there are black pixels until the perspective wrapped pixel values from the right image transformed using homography matrix are located as illustrated in figure 14. So, this algorithm searches for the first-pixel value other than a black pixel in each row of the image and checks if that is the minimum value in the transformed image. In YUV color space, the black pixels are illustrated with values Y = 0, U = 128 and V = 128. After the minimum value of the starting index in each of the rows of the image is obtained, the BCE algorithm progresses further. The starting point of the blending has a significant impact on the performance of the BCE algorithm as the calculation in the BCE algorithm is based on the mean of the brightness level of input images in the overlapping region. The starting point of the blending is calculated only once at the start of the BCE algorithm or whenever the camera positions are

changed. In the case of cameras that are constantly moving, calculation of the starting point of blending has to be carried out in every frame. The implementation of the process for the calculation of the blending point is shown in algorithm 2.

Algorithm 2: Algorithm for Determination of Overlapping Region (CALCULATEBLENDINGPOINT)

Algorithm 2: CALCULATEBLENDINGPOINT

Input : Transformed image (RI) of the second frame based on the reference image, the width of the image(w), the height of the image(h)

Output : Starting point from which overlapping region between two images begins

```
begin
1   Y ← 0 , U ← 0, V ← 0, position_luma ← 0, position_chroma ← 0
2   minimum ← w
3   for i : 0 → h
4       for j : 0 → w
5           position_luma ← i*w+j
5           position_chroma ←h*w+(i/2)*w+j
7           Y ← RI [position_luma]
8           U ← RI [position_chroma]
9           V ← RI [position_chroma + 1]
10          If (Y!= 0 && U!= 128 && V!= 128)
11              If (minimum > j)
12                  minimum ← j
13                  i ← i ++
14  return minimum
end
```

5.3. Algorithm for Brightness and Color Equalization

After starting point of the overlapping region is determined, the average brightness level of the left (reference) image and right (transformed) image in the overlapping region is calculated. The mean brightness is used as a basis for the BCE algorithm and is calculated using the algorithm CALCULATEMEAN. The darker image is processed through the BCE YUVCORRECTION algorithm to match the brightness and color profile of the brighter image. The calculations for equalization of brightness and color are described in section 3. Different brightness and color component equalization methods can be chosen to achieve equalization in a panoramic image. The first gamma value for brightness is calculated for darker images among input images and if the calculated value of gamma is lowers than 0.85, the brightness and color equalization method is

performed on the darker image. Thresholding of gamma value for brightness at 0.85 is carried out because the images with values between 0.85 to 1 can be blended directly to obtain a smooth panoramic image without the requirement for brightness and color equalizations. The threshold value of 0.85 is obtained using a blending of sample images with brightness differences which provides gamma values between 0.5 to 1 without BCE equalization to determine the threshold below which BCE equalization is required. The calculation of new pixel values of brightness and color components for the darker image is carried out only once at the beginning of YUVCORRECTION for a pair of images between 0 to 255 in an 8-bit image using the algorithm CALCULATELUTY for brightness and CALCULATELUTUV for color components. The implementation of the algorithm is shown in algorithm 3.

Algorithm 3: Algorithm for calculation of brightness and color pixel values

Algorithm 3: YUVCORRECTION

Input : The source image for the first frame(LI) and their dimensions(lw,lh), transformed image (RI) of the second frame based on the reference frame and their dimensions (rw,rh), the starting point for blending(sbp)

Output : Brightness and Color corrected frame (Darker Image gets corrected)

```
begin
1     left_mean ← CALCULATEMEAN(LI,lw,lh,sbp)
2     right_mean ← CALCULATEMEAN (RI,lw,lh,sbp)
3     method_Y, method_UV ← 0 or 1 (0 for Gamma , 1 for Linear Transformation)
3     if (left_mean > right_mean)
5           difference ← (left_mean - right_mean)
6           gamma ← right_mean/(left_mean+difference)
            if(gamma < 0.85)
7                 CALCULATELUTY (gamma,method_Y)
8                 gamma_chroma ← 1/(1 + (1-gamma)/2)
10                CALCULATELUTUV (gamma_chroma,method_UV)
11                APPLYLUT(RI,rw,rh)
12    else
13          difference ← (right_mean - left_mean)
14          gamma ← left_mean/(right_mean+difference)
            if(gamma < 0.85)
15                CALCULATELUTY (gamma,method_Y)
16                gamma_chroma ← 1/(1 + (1-gamma)/2)
18                CALCULATELUTUV (gamma_chroma,method_UV)
19                APPLYLUT (LI,lw,lh)
end
```

BCE processes CALCULATEMEAN, CALCULATELUTY, CALCULATELUTUV, and APPLYLUT are provided in Annex A, B, C, and D respectively. The final pixel values of the darker image are obtained through lookup tables for brightness and color component. Lookup tables are an array that is mostly used to replace runtime computations with simple index search operations in an array. For an 8-bit image, 256 values are possible. Lookup tables used for 8-bit images should have 256 elements which contain the value of the final brightness and color equalized pixel for the original value as an array element index. Lookup tables contain the original pixel value as array index and the array element is the final pixel value obtained after equalization. It reduces the computations to 512 computations, 256 each for luminance and chrominance irrespective of the image resolution. The chrominance components are applied with the same lookup table and a separate lookup table for brightness with algorithm APPLYLUT.

Figure 15: BCE brightness and color equalization for the darker image in figure 12 (a – Brightness equalization using gamma correction, b – Brightness equalization using a basic linear transformation, c – Color equalization using gamma correction, d – Color equalization using basic linear transformation).

Figure 15 demonstrates the application of BCE brightness and color equalization methods using available methods. The gamma, gain, and bias parameters during YUVCORRECTION for the images in figure 15 are mentioned below.

ColorSpace/Parameters	Gain	Bias	Gamma
Brightness	1.272753	9.090592	0.727247
Color	1.189415	2.199975	0.879990

5.4. Algorithm for Blending Operations

After carrying out YUVCORRECTION of the darker image, the images are passed through a blending process for creating a panoramic image. The implementation of the blending operations to obtain the panoramic image is shown in algorithm 4.

Algorithm 4: Algorithm for blending input images to form a panoramic image

Algorithm 4: BLENDIMAGES

Input : The source image for the first frame (LI) and their dimensions(lw,lh), transformed image(RI) of the second frame according to the homography matrix calculated to match the perspective of the first frame and their dimensions(rw,rh), the starting point for blending(sbp) from which pixel values overlap between two images and pointer to save the panoramic image (OI)

Output : Stitched panoramic image of input frames

```
begin
    for i : 0 → lh
        for j: sbp → lw
            Y_left ← LI [i* lw + j]
            U_left ← LI [lw*lh + (i/2)* lw + j]
            V_left ← LI [lw*lh + (i/2)* lw + j + 1]
            Y_right ← RI [i* rw + j]
            U_right ← RI [rw*rh + (i/2)* rw + j]
            V_right ← RI [rw*rh + (i/2)* rw + j +1]
            pix_black← (Y_left=0)*(U_left=128)*(V_left=128)
            wrap_black←(Y_right=0)*(U_ right=128)*(V_right=128)*(pix_black=0))
            if(wrap_black)
                (Y,U,V) ← (Y_left, U_left, V_left)
            else
                Y ← (Y_left * weights[j] + Y_right * (1-weights[j]))
                U ← (U_left * weights[j] + U_right * (1-weights[j]))
                V ← (V_left * lweights[j] + V_right * (1-weights[j]))
            clip(Y,U,V,(0,255))
            OI [i* rw + j] ← Y
            OI [rw*rh + (i/2)* rw + j] ← U
            OI [rw*rh + (i/2)* rw + j +1] ← V
end
```

In the blending process, the pixels in the overlapping areas are blended according to the weights assigned to the pixel position. During the blending process, the image pixels that are black due to perspective warping are copied from the left reference image to avoid dissimilarities in those regions while the rest are blended with weighted ratios. The algorithm for the calculation of weights

is carried out through CALCULATEBLENDWEIGHTS which is provided in Annex E. The pixels towards left and right from the overlapping region are copied from their respective counterparts. Finally, brightness and color equalized panoramic images are created.

The images in Figure 16 demonstrate the implementation of the developed blending algorithm with and without BCE equalization implementation. The resulted panoramic images are displayed with green lines in the overlapping region to highlight the blending area. Also, all the possible combinations of BCE methods (GG, GLT, LTG, and LTLT) and the resulting panoramic image sequences obtained are displayed in figure17 for input images shown in the figure.

Figure 16: Blending Images before and after BCE equalization

Figure 17: Examples of implementation of BCE using combination of basic linear transform and gamma correction

[a - left reference frame, b – right transformed frame, c,d,e,f – panoramic image generated after BCE algorithm using methods GG, GLT, LTG, LTLT, respectively, where G represents BCE correction using gamma correction, and LT represents BCE correction using a basic linear transformation, the first letter denotes method for correction of brightness and the second letter denotes method for correction of color components.]

5.5. Optimization of the Algorithms

Implementation of the above algorithms was carried out in a single algorithm link using VisionSDK. The developed algorithm link was used to develop use-cases for the ALPHA board. For optimizations, the execution time of the algorithm was measured in seconds(s), milliseconds (ms), and microseconds (μs).

The first implementation was carried out using IPU which supports reading and writing operations from SD cards. The input frames (left reference frame and the transformed second frame according to the homography matrix) were copied to the SD card and the algorithm reads the input frames from the SD card. The perspective wrapping of the right frame based on the reference frame is carried out in the PC before copying the images so that only the BCE algorithm is performed in

the ALPHA board. Function pointers were used for reading and writing images as algorithm links do not support reading and writing operations in VisionSDK. The resulting panoramic image from the algorithm link was saved to the SD card. In the initial implementation, the execution speed includes reading and writing operations from and to SD card, due to which another implementation using network functionalities were used to avoid reading and writing to SD card and the resulting panoramic image was saved to PC. The use-case flowchart of algorithm implementation using IPU is shown in Annex F.

The optimization was carried out for the first few versions of implementation in IPU and the algorithm was shifted to DSP to enable faster execution of the algorithm. The algorithm implementation in DSP showed faster performance capabilities than IPU as expected. This is due to the architecture behind the DSP processor which runs on the C66x library that has dedicated support for floating-point operations. Implementation in DSP was carried out using network functionalities and both of the input frames were passed at once horizontally stacked together as seen in Figures 1 and 12. The resulted panoramic image was saved to PC using the network functionalities. After optimization in DSP, to enable much faster execution, both of the DSP processors (DSP1 and DSP2) were used. The flowchart of the use-case for implementation of the algorithm using both DSP is shown in Annex G.

Finally, the algorithm was shifted to an A15 processor to test the execution speed. The result obtained in the A15 processor was by far the best performance enabling the BCE algorithm to obtain the lowest execution time among all the processors. The boost obtained in execution can be explained by the L2 cache memory available to the processors. The DSP cores have 256 KB of L2 cache memory enabling a total of 512 KB of L2 cache in case of multiple processor implementation. The A15 core has 2 MB of combined L2 cache memory available for ARM cores (2xA15 and 2xM4). The implementation of Neon Optimization is built-in in the VisionSDK platform for ARM cores. The flowchart of the use-case for implementation of the algorithm using A15 is shown in Annex H. The implementation in A15 core was carried out similar to the implementation in DSP core using network functionalities.

There are initial and final versions of algorithm implementations for each of the processors before and after optimizations. The initial version of IPU is the first implementation of the BCE algorithm in the ALPHA board or transport the previous final version to the next processor (i.e. the

final version of implementation in IPU is the initial version of implementation in DSP and so on). The final version of the implementation is the execution of the algorithm in the same processor after certain optimization steps are carried out.

The optimization steps carried during algorithm implementation in IPU are listed below:

a) Assumed the camera positions to be constant so that the algorithm for the determination of starting point (CALCULATEBLENDINGPOINT) can be avoided and replaced with pixel position assignment.
b) Calculation of blending weights (CALCULATEBLENDINGWEIGHTS) carried out in a separate algorithm is place inside the BLENDIMAGES algorithm.
c) Two multiplication, one addition and subtraction are replaced with one operation each of addition, subtraction, and multiplication. i.e., Y_left*W_left + Y_right*(1-W_left) replaced with (Y_left-Y_right) *W + Y_right for blending pixels.
d) Reduced looping by decreasing height iterator (720 to 360) in APPLYLUT and BLENDIMAGES to operate two rows simultaneously.

The optimization steps carried during algorithm implementation in DSP are listed below:

a) Using multiplication inside BLENDIMAGES instead of nested if-else for determining black pixels and pixels obtained from perspective wrapping in areas of the overlapping region.
b) Using multiple DSP to blend images. Blending operation is divided into upper and bottom half of the image and each processor carries out blending simultaneously.
c) The data type used for calculation of sum in CALCULATEMEAN replaced from double to int.
d) Instead of copying each pixel through the nested loop, the right part of the image is copied using *memcpy* operation.

The optimization steps carried during algorithm implementation in A15 are listed below:

a) Blending pixel iteration inside BLENDIMAGES increased to 100 pixels from 8 pixels (ie. loop iterator step size for height and width increased). Blending weights are calculated at 10-pixel width and blending occurs at 10x10 pixel area in a single pass of the loop.

b) Replaced positional index calculation in each function call with single calculation on posLuma and posChroma wherever applicable in functions BLENDIMAGES, APPLYLUT, and copying right part.

The comparison regarding execution parameters of the algorithm during the optimization phase is mentioned in detail in Table 3. For this comparison, 50 frames each having different overlapping areas are taken from the video sequence. The final version of the implementation of the algorithm is provided in ANNEX F.

Table 3: Comparison of Algorithm Implementation in various CPUs

Overlapping Area		670 pixels (x720 pixels)			440 pixels (x720 pixels)		
S.N.	Tasks Performed	AET	MET	SD	AET	MET	SD
1	Initial Algorithm (IPU)	14.12 s	14.68 s	0.169	10.17 s	10.999 s	0.158
2	Final IPU Implementation	5.62 s	6.26 s	0.184	4.53 s	5.025 s	0.112
3	Initial DSP Implementation	1.62 s	1.63 s	0.005	1.48 s	1.498 s	0.005
4	Final DSP Implementation	163.99 ms	164.04 ms	0.023	119.84 ms	119.86 ms	0.014
5	Initial A15 Implementation	25.33 ms	25.59 ms	0.252	18.46 ms	18.87 ms	0.157
6	Final A15 Implementation	20.70 ms	20.76 ms	0.027	15.54 ms	15.99 ms	0.068

AET[1]-Average Execution Time, MET[2]-Maximum Execution Time, SD[3]-Standard Deviation in Execution Time

6. VALIDATION AND TESTING OF PROPOSED SOLUTION

In this section, the validation and testing of the proposed solution are given. After the algorithms for brightness and color equalizations are developed and implemented, the resulted panoramic image is then used for evaluations. Since the lack of appropriate databases for testing in a real traffic situation, subjective quality testing is carried out. The Blind/Referenceless Image Spatial Quality Evaluator (BRISQUE) is calculated and the subjective quality was observed. The obtained values of the quality of the resulted panoramic images provide an approximate prediction of the image quality. In addition to BRISQUE values, a subjective quality evaluation among the developed methods for brightness and color equalizations with a sample size of 180 frames is carried out to determine the optimal solution with 15 user preference entries.

During the development and implementation of the BCE methods, implementation in the PC and ALPHA board was carried out simultaneously using the frames illustrated in Figures 1 and 12. Different values of gamma, bias, and gain parameters were calculated to achieve the natural-looking panoramic image. After continuous modification of gain and bias parameters for brightness and color components, the relationship of compensation of parameters between brightness and color components was achieved which enables the creation of the natural-looking panoramic image. The implementation between the PC and ALPHA board was compared and all the pixel values were checked to ensure that the BCE algorithm equalizes brightness and color with the same degree of increment/decrement in both platforms. As a result, no differences in the panoramic image were achieved between PC and ALPHA board implementation. After the implementation of the algorithm was achieved, validation of the BCE algorithms was carried out using an image sequence from video recorded from three cameras. The cameras are placed at different distances from which the variation in overlapping regions between adjacent cameras was achieved. From the video, 50 frames were selected, 25 image pairs each from left and middle cameras and middle and right cameras, respectively.

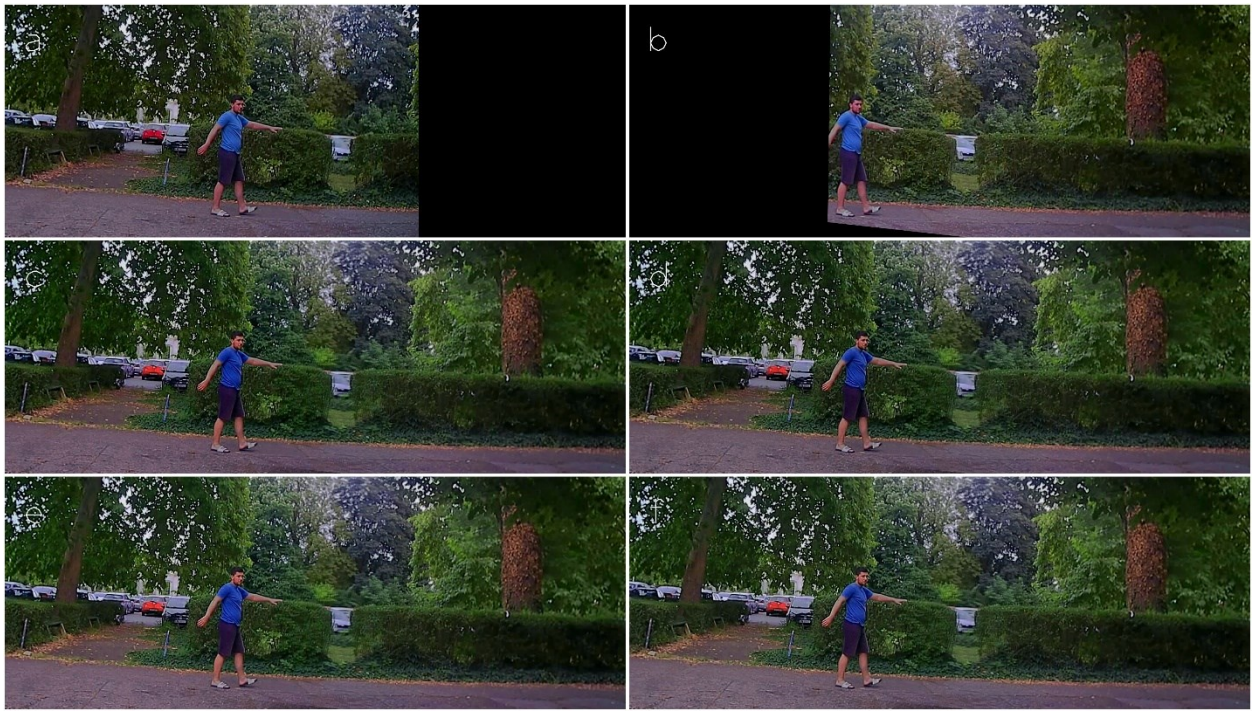


Figure 18: Panoramic images passed through BCE algorithms for frame obtained from left and middle camera (Overlapping Region from 610 to 1280)

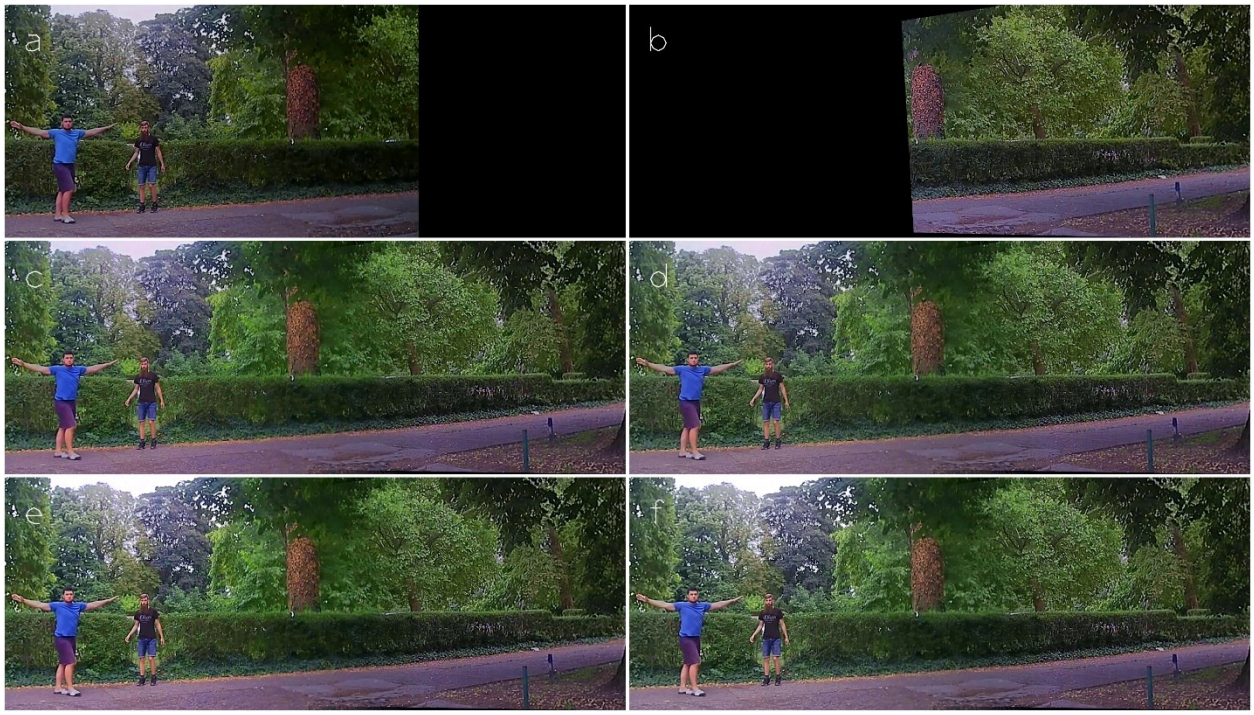


Figure 19: Panoramic images passed through BCE algorithms for frame obtained from the middle and right camera (Overlapping Region from 840 to 1280)

[a - left reference frame, b – right transformed frame, c,d,e,f – panoramic image generated after BCE algorithm using methods GG, GLT, LTG, and LTLT respectively]

The input frames were passed through to all four combinations of BCE algorithms (GG, GLT, LTG, and LTLT methods) to generate a panoramic image. The left frame was taken as reference and the right frame was transformed according to the calculated homography matrix i.e. for left and right cameras, the frame from left cameras was taken as a reference frame and for the middle and right cameras, the frame from the middle cameras was taken as reference. Out of the 50 images, 19 images that were fed to the BCE algorithm did not undergo brightness and color correction as the ratio of mean brightness between input images was higher than 0.85. Those images were blended without brightness and color correction. The rest of 31 images underwent brightness and color correction which results in different panoramic images. The examples of implementation of BCE methods from left and middle cameras and middle and right cameras can be seen in Figures 18 and 19 respectively. Since there was a lack of testing dataset for testing the BCE algorithm, non-reference quality assessment using the BRISQUE method and subjective image quality testing was carried out. Lower the value of BRISQUE, higher the image quality, and vice versa. The BRISQUE scores obtained for 30 images that underwent the BCE algorithm are presented in ANNEX J. The scores of the images for different methods were within the competitive range and didn't show any significant variation to derive a conclusive result. However, among the four methods developed (GG, GLT, LTG, and LTLT), the LTG method had the lowest BRISQUE score indicating higher image quality. The same images used for calculations of BRISQUE scores were used for subjective image quality assessment to determine the most preferred method of brightness and color correction among available methods in the BCE algorithm.

The comparison of the images was done using two panoramic images stacked vertically that were formed using different BCE methods. From 30 images, using comparisons among four methods, a total of 180 comparisons were created. The testing was carried out in two stages comprising of 90 images each from 15 frames with the random occurrence of the test sequences. The testing tool was created using GUI application in C# and random occurrence of the test sequence was achieved using random shuffling of 90 elements in an array. The score of the test sequence was then recorded using an array index to register the score to the appropriate test sequence. The user was able to score the top iteration of the image with comparison to the bottom

image according to their preference of brightness and color. The scores recorded from 15 users are provided in ANNEX K.

The comparisons were then analyzed using the Bradley-Terry (BT) model of paired comparisons. Using the BT model, previous scores can be used to estimate the future probability of the method favorable to be chosen between the comparison methods. The possible probability of the method to beat another method in paired comparisons can be represented by equation 7.1 [15].

$$P(i\ beats\ j) = \frac{w_i}{w_i + w_j} \tag{7.1}$$

where $w_i \geq 0$ is the preference of method i over j, $w_j \geq 0$ is the preference of method j over i and P is the probability of preference of method i over j.

In the above equation, the comparison is done between only two methods. In addition to a paired comparison between two methods, this BT model can also be applied to multi-class probability estimates by combining all pairwise comparison results [16]. The multi-class comparison involves paired comparisons between any two methods among all the available combinations. In this thesis, there are four methods in total for the BCE algorithm which generates six combinations of comparisons among the available methods. Multi-class probability estimates are described using different approaches which are explained in [16]. Regarding the Bradley-Terry model for multi-class probability estimates, weighted individual skill, ties, advantages, and multiple team comparisons are presented in [16]. The multiple class comparisons are important to our subjective image quality assessment for different BCE algorithms. The model presented in [16] is based on a general model derived from [17]. The comparison in subjective image quality assessment is based on the general model which is presented in equation 7.2.

$$P(a(1) \rightarrow a(2) \rightarrow \cdots \rightarrow a(n)) = \prod_{i=1}^{n} \frac{p_{a(i)}}{p_{a(i)} + p_{a(i+1)} + \cdots + p_{a(n)}} \tag{7.2}$$

In above equation 7.2, $P(a(i))$ represents the probability of class i to be chosen over all the other classes, $p_{a(i)}$ represents the total number of occurrences in which class i was favored over all other classes. In this thesis, since there are only four BCE methods in comparison, $= 4$.

Since the input is the number of wins or preferences of one method over the other, the user data has to be sorted according to wins. For analysis of the data entry from users, if the score for the top frame is positive the corresponding method of the BCE algorithm of the top frame is considered as preferred or winner. If the score for the top frame is negative, the corresponding method of the BCE algorithm of the bottom frame is considered as preferred or winner, and if the score is zero it is considered tied. One of the examples of user scores in subjective quality testing is shown in figure 20.

Figure 20: Subjective image quality testing user interface

The ties are not the priority of this thesis and we will be focusing only on wins among multiple comparisons. The user data from ANNEX K are sorted according to wins, losses, and inconclusive in case of a draw. The table for wins or losses for comparisons from user scores is presented in ANNEX L. The table presented in ANNEX L is parsed using a MATLAB script for calculation of wins according to BCE methods to calculate the final 4X4 matrix(M) that summarizes the subjective image quality testing comparisons. The MATLAB script is provided in ANNEX M. The total occurrences of wins using comparison with other BCE methods are provided as 4x4 matrix M.

$$M = \begin{bmatrix} 0 & 235 & 153 & 210 \\ 174 & 0 & 253 & 164 \\ 144 & 138 & 0 & 226 \\ 206 & 128 & 176 & 0 \end{bmatrix}$$

where M_{ij} represents the number of user preferences for method i over method j for BCE methods. The first, second, third, and fourth rows and columns are GG, GLT, LTG, and LTLT methods respectively.

MATLAB scripts for probability estimation based on the BT models are available online from [18] and can be used directly in MATLAB. From the link, the BT model for Expectation-Maximization (EM) and the Gibbs BT model are used in this thesis. The details about EM and Gibs MT model can be found [19]. The final probabilities are calculated from EM and the Gibbs model extensions presented in [19] using the matrix M. This matrix M is used as input for calculations of the final probability using the EM and Gibbs extensions for BT model. The resulting scores are mentioned in Table 4.

Table 4: Methods Comparison between BCE methods

Method	EM(Mean)	Gibbs(Mean)	Gibbs(SD)
GG	0.2756	0.2758	0.0120
GLT	0.2832	0.2829	0.0118
LTG	0.2228	0.2225	0.0107
LTLT	0.2184	0.2188	0.0101

From the above comparisons, it can be stated that out of four BCE methods, the most preferable method is brightness correction using gamma correction and color correction using a basic linear transformation. We can observe from matrix M that BCE using GG has the highest number of wins beating BCE using GLT with 7 wins. But due to the higher standard deviation of wins probability of BCE using GG method, BCE using GLT wins over BCE using GG with a slight margin. Also, the result suggests that the selection of the BCE method is dependent on user preference of brightness and color levels.

7. CONCLUSION

The methods for brightness enhancements can be classified as linear and non-linear operations. This thesis proposes one method for each of the linear and non-linear methods for brightness and color correction. BCE method using basic linear transform and gamma correction is developed using the mean brightness level of the input images as the reference. Implementation of the developed BCE algorithm was carried out using the ALPHA board. During the development of the algorithm, three sequences of images were used to calibrate the parameters of the BCE algorithm such as gamma factor for gamma correction, bias, and gain factors in basic linear transformation. The parameters for all color space components are based on the mean brightness level of the input images are calculated considering only the overlapping area between the images. Since the perception of the brightness is much more sensitive to the human eye than the colors, the parameters were adjusted to perform color equalization with relation mentioned in Equations 3.6 to 3.13. The developed methods for the BCE algorithm were validated using 50 frames from video sequences from three cameras. The execution time of the BCE algorithm in different CPUs in the ALPHA board was measured and optimization was carried out to enable faster execution of the algorithm. The fastest implementation was achieved in the A15 processor of the ALPHA board.

Finally, the panoramic images were subjected to the BRISQUE model, and respective scores were recorded. The variations in BRISQUE scores of the BCE methods were not significant due to which subjective image quality assessment had to be carried out. In subjective image quality assessment, two panoramic images resulted from different BCE method is stacked vertically and shown to the user. The user had to rate the top image between 1 to 3 if the upper image is preferred else the rating should be -1 to -3. The test was carried out using 30 images and 4 different BCE methods which resulted in 180 test sequences. The scores were recorded and analyzed using the Bradley-Terry model using EM and Gibbs extension for determination of the possibility of certain BCE methods to be favored. After comparison, the BCE method using GLT was found to be most favorable to be picked. The second likely to be preferred was GG and the least favorable was found to be LTLT.

REFERENCES

[1] H. Bay, A. Ess, T. Tuytelaars, and L. Van Gool, "Speeded-Up Robust Features (SURF)," *Comput. Vis. Image Underst.*, vol. 110, no. 3, pp. 346–359, Jun. 2008, doi: 10.1016/j.cviu.2007.09.014.

[2] L. Juan and G. Oubong, "SURF applied in panorama image stitching," in *2010 2nd International Conference on Image Processing Theory, Tools and Applications*, Paris, France, Jul. 2010, pp. 495–499. doi: 10.1109/IPTA.2010.5586723.

[3] C. Xiu and Y. Ma, "Image Stitching Based on Improved Gradual Fusion Algorithm," in *2019 Chinese Control And Decision Conference (CCDC)*, Nanchang, China, Jun. 2019, pp. 2933–2937. doi: 10.1109/CCDC.2019.8832814.

[4] R. Szeliski, *Computer Vision*. London: Springer London, 2011. doi: 10.1007/978-1-84882-935-0.

[5] M. Salem, S. Al-Amri, N. Kalyankar, and Khamitkar, "Linear and Non-linear Contrast Enhancement Image," *IJCSNS Int. J. Comput. Sci. Netw. Secur.*, vol. 10, Jan. 2010.

[6] G. Cao, L. Huang, H. Tian, X. Huang, Y. Wang, and R. Zhi, "Contrast enhancement of brightness-distorted images by improved adaptive gamma correction," *Comput. Electr. Eng.*, vol. 66, pp. 569–582, Feb. 2018, doi: 10.1016/j.compeleceng.2017.09.012.

[7] A. A. Mohammed, F. Ming, and R. Zhengwei, "Color Balance for Panoramic Images," *Mod. Appl. Sci.*, vol. 9, no. 13, Art. no. 13, Nov. 2015, doi: 10.5539/mas.v9n13p140.

[8] K.-C. Hu, F.-Y. Lin, C.-C. Chien, T.-S. Tsai, C.-H. Hsia, and J.-S. Chiang, "Panoramic image stitching system for automotive applications," in *2014 IEEE International Conference on Consumer Electronics - Taiwan*, Taipei, Taiwan, May 2014, pp. 203–204. doi: 10.1109/ICCE-TW.2014.6904058.

[9] J. Pan, V. Appia, J. Villarreal, L. Weaver, and D.-K. Kwon, "Rear-Stitched View Panorama: A Low-Power Embedded Implementation for Smart Rear-View Mirrors on Vehicles," in *2017 IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, Honolulu, HI, USA, Jul. 2017, pp. 1184–1193. doi: 10.1109/CVPRW.2017.157.

[10] N. Perkovic, M. Pranjkic, I. Kolak, and G. Velikic, "System for automotive machine vision," in *2017 IEEE 7th International Conference on Consumer Electronics - Berlin (ICCE-Berlin)*, Berlin, Sep. 2017, pp. 243–245. doi: 10.1109/ICCE-Berlin.2017.8210638.

[11] J. Sankaran and N. Zoran, “TDA2X, a SoC optimized for advanced driver assistance systems,” in *2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, Florence, Italy, May 2014, pp. 2204–2208. doi: 10.1109/ICASSP.2014.6853990.

[12] B. Kragulj, S. Laza, M. Milosevic, and G. Velikic, “One solution of complex ADAS HW system testing,” in *2017 25th Telecommunication Forum (TELFOR)*, Belgrade, Nov. 2017, pp. 1–4. doi: 10.1109/TELFOR.2017.8249477.

[13] “Automotive Machine Vision ALPHA reference board on Texas Instruments SoCs.” Accessed: Jul. 01, 2021. [Online]. Available: https://www.rt-rk.com/download/rt-rk_ALPHA_ADAS_board.pdf

[14] “TI Vision SDK, Optimized Vision Libraries for ADAS Systems,” p. 12, 2014.

[15] R. A. Bradley and M. E. Terry, “Rank Analysis of Incomplete Block Designs: I. The Method of Paired Comparisons,” *Biometrika*, vol. 39, no. 3/4, p. 324, Dec. 1952, doi: 10.2307/2334029.

[16] T.-K. Huang, R. C. Weng, and C.-J. Lin, “Generalized Bradley-Terry Models and Multi-Class Probability Estimates,” *J. Mach. Learn. Res.*, vol. 7, no. 4, pp. 85–115, 2006.

[17] R. L. Plackett, “The Analysis of Permutations,” *Appl. Stat.*, vol. 24, no. 2, p. 193, 1975, doi: 10.2307/2346567.

[18] “Matlab/Octave package BayesBT.” http://www.stats.ox.ac.uk/~caron/code/bayesbt/ (accessed Jul. 04, 2021).

[19] F. Caron and A. Doucet, “Efficient Bayesian Inference for Generalized Bradley—Terry Models,” *J. Comput. Graph. Stat.*, vol. 21, no. 1, pp. 174–196, 2012.

ABSTRACT

Brightness and color equalization in images obtained from multiple cameras to form a panoramic image is carried out in this thesis. One of the linear and non-linear methods was proposed for carrying out brightness and color transformation of the darker image considering the overlapping area between the input images. In addition to brightness and color equalizations, blending operations were also proposed to obtain a smooth transition at the overlapping region between the input images. All the processes combined form an algorithm for brightness and color equalization (BCE). GG, GLT, LTG, and LTLT are the four methods of the BCE algorithm that can be applied to form a panoramic image. After the development of the BCE algorithm, implementation of the algorithm using the ADAS development board was carried out. After implementation, optimization of the algorithm was performed to ensure real-time application. The implementation using the A15 processor was found to be suitable than the rest of the processors on the ALPHA board. The panoramic images formed after BCE operations were evaluated using BRISQUE scores and subjective image quality assessment. The best image could not be selected using BRISQUE scores because the scores of the panoramic images from different BCE methods were within the limits of 43 and 44. Subjective Image quality assessment was carried out using the same panoramic images. The BTEM model was used for analyzing the user scores and prediction of the most preferable BCE method. The BCE using the GG method was found to be the most favorable among all BCE methods.

Keywords: ADAS, Basic Linear Transformation, Gamma Correction, Gain, Bias, Gamma, BCE, GG, GLT, LTG, LTLT, BRISQUE, BTEM

BIOGRAPHY

PRAKASH DHUNGANA

He was born on 10th April 1994 in Manahari, Makwanpur district from Nepal. He completed his primary school at United Academy in Manahari and secondary school at Ideal Model School in Lalitpur, Nepal.

In 2012, he completed his higher secondary education at Prasadi Academy in Lalitpur, Nepal. In the same year, he enrolled in the undergraduate study of Mechanical Engineering at Pulchowk Campus, Institute of Engineering (IOE) which he completed in 2016. In December 2017, he was awarded a "Bachelor's Degree in Mechanical Engineering" by Tribhuvan University. After completing his studies, he worked in the field of automotive as a service advisor for three years.

In 2019, he enrolled in graduate study of automotive computing and communications.

Osijek, 15th July 2021

LIST OF FIGURES

LIST OF TABLES

LIST OF ALGORITHMS

ABBREVIATIONS

ADAS	Advanced Driver Assistance Systems
SDK	Software Development Kit
HE	Histogram Equalization
AHE	Adaptive Histogram Equalization
CLAHE	Contrast Limiting Adaptive Histogram Equalization
RGB	Color Space (R – Red, G – Green, B – Blue)
YUV	YCbCr Color Space (Y – Luminance, Cb, Cr – Chrominance)
HSV	Color Space (H – Hue, S – Saturation, V – Brightness/Value)
SoC	System on Chip
DSP	Digital Signal Processor
EVE	Embedded Vision Engine
BRISQUE	Blind/Referenceless Image Spatial Quality Evaluator
SIFT	Scale-Invariant Feature Transform
SURF	Speeded Up Robust Features
K-NN	K Nearest Neighbors
RANSAC	Random Sample Consensus
CE	Contrast Enhancement
SECE	Spatial Entropy-based Contrast Enhancement
AGC	Adaptive Gamma Correction
AGCWD	Adaptive Gamma Correction with Weighting Distribution

CDF	Cumulative Distribution Function
RSVP	Rear Stitched View Panorama
TI	Texas Instruments
AMV	Automotive Machine Vision
ISP	Image Signal Processor
SECDED	Single Error Correct and Double Error Detect
SER	Soft Error Rate
ECC	Error Correcting Code
SCV	Surround Camera View
FFN	Front View Camera Near Angle Stereoscopic View, Front View Camera Wide Angle, Night Vision Camera
FUS	Fusion
CMS	Camera Mirror System
PCIe	Peripheral Component Interconnect Express
DDR3	Double Data Rate Type 3
RAM	Random Access Memory
SDRAM	Synchronous Dynamic RAM
SD card	Secure Digital Card
ROM	Read-Only Memory
EEPROM	Electrically Erasable Programmable ROM
API	Application Programming Interface
BSP	Build Support Package
AppImage	Application Image

BCE	Brightness and Color Equalization
AET	Average Execution Time
MET	Maximum Execution Time
SD	Standard Deviation
GG	BCE Using Gamma Correction for both Luminance and Chrominance
GLT	BCE Using Gamma Correction for Luminance and Basic Linear Transformation for Chrominance
LTG	BCE Using Basic Linear Transformation for Luminance and Gamma Correction for Chrominance
LTLT	BCE Using Basic Linear Transformation for both Luminance and Chrominance
EM	Expectation Maximization
BT	Bradley – Terry Model
BTEM	Expectation Maximization using BT model

ANNEXES

ANNEX A: Algorithm for calculation of the mean value of brightness in an overlapping region

Algorithm:	**CALCULATEMEAN**
Input :	Image (IM) frame for which mean of Y values is to be calculated, width of image(w), height of the image(h), the starting point of overlapping pixels for blending(sbp)
Output :	Mean value of the brightness in the overlapping area

```
begin
1   sum ← 0, mean ← 0, counter ← 0
2   for i : 0 → h
3       for j: sbp → w
4           Y ← IM [i*w + j]
5           if ( Y!= 0)
6               sum ← sum + Y
7               counter ← counter + 1
8   mean ← (sum/counter)
9   return mean
end
```

ANNEX B: Algorithm for lookup table calculation of pixel value for brightness

Algorithm:	**CALCULATELUTY**
Input :	Value of ratio(gamma) for calculation of Gamma values in case of gamma correction(method = 0), alpha and beta values in case of Linear Transformation (method = 1)
Output :	Corrected brightness values saved to global variable LUT_Y[256]

```
begin
1   alpha ← 2 - gamma
2   beat ← 20 * alpha
3   if(method = 0)
4       for j : 0 → 255
5           LUT_Y[j] ← pow(j/255, gamma)* 255
6   else
7       for j : 0 → 255
8           LUT_Y[j] ← clip ((alpha * j + beta), (0,255))
end
```

ANNEX C: Algorithm for lookup table calculation of pixel value for brightness

Algorithm: CALCULATELUTUV

Input : Value of ratio(gamma) for calculation of Gamma values in case of gamma correction(method = 0), alpha and beta values in case of Linear Transformation (method = 1)

Output : Corrected color values saved to global variable LUT_UV[256]

```
begin
ratio ← 0
alpha ← 2 - gamma
beat ← 20 * alpha
if(method = 0)
    for j : 0 → 255
        if( j > 128)
            ratio ← pow((j-128)/128, gamma)
            final ← (ratio*128  + 128)
        else
            ratio ← pow((128-j)/128, gamma)
            final ← (128 - ratio*128)
        LUT_UV[j] ← final
else
    for j : 0 → 255
        if( j > 128)
            ratio ← ((j - 128)* alpha + beta + 128)
        else
            ratio ← (128 – (128 - j)* 128 - beta)
        ratio ← clip (ratio, (0,255))
        LUT_UV[j] ← final
end
```

ANNEX D: Algorithm for application of lookup tables for YUV components

Algorithm: APPLYLUT

Input : Image(IM) frame for which brightness and color components is to be corrected, width of the image(w), height of the image(h)

Output : Corrected brightness and color values updated in the received image (IM)

```
begin
for i : 0 → h
    for j : 0 → w
        IM [i * w + j] ← LUT_Y [IM [i * w + j]]
for i : 0 → (h/2)
    for j : 0 → w
        IM [h*w + i * w + j] ← LUT_Y [IM [h * w + i * w + j]]
end
```

ANNEX E: Calculation of blending weights for pixels

Algorithm:	**CALCULATEBLENDINGWEIGHTS**
Input :	Starting point for pixels in the transformed image(s), the width of the image according to which the first image is transformed to match perspective(e)
Output :	Blending weights saved to global variables

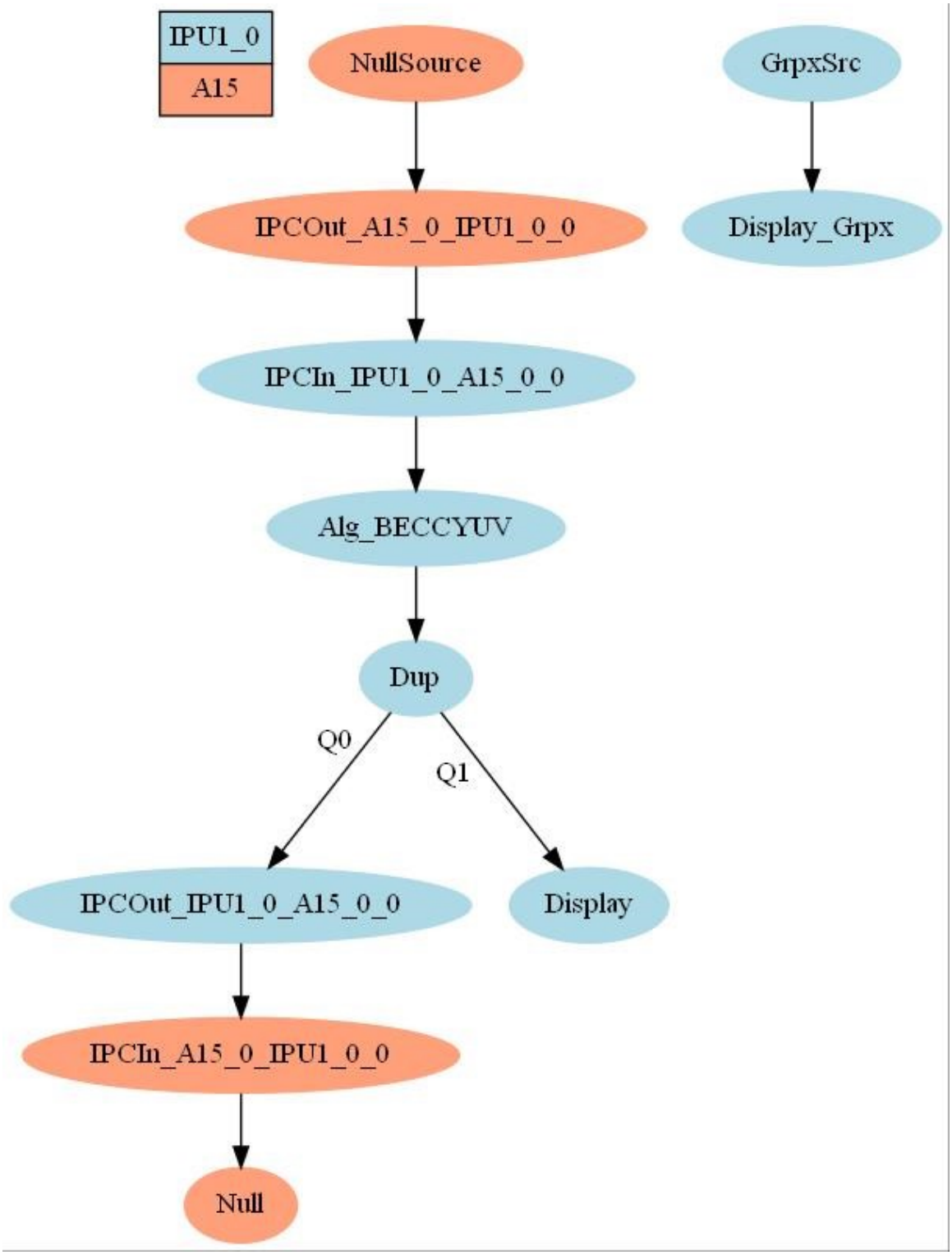
```
begin
1    w ← (e – s)
2    ratio ← 0
3    for j : 0 → w
4        ratio ← (w*w – j*j)/(w*w)
5        weights[s + j -1] ← ratio
end
```

ANNEX F: Optimized algorithm for brightness and color equalization

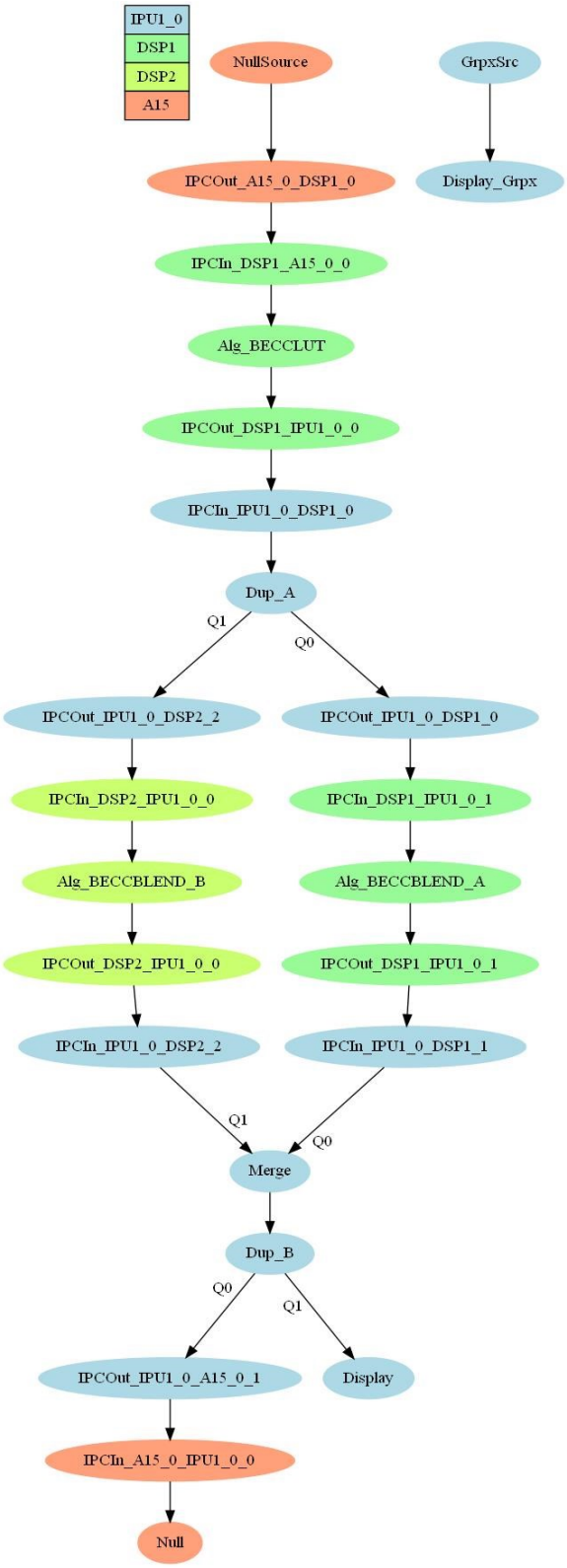
Algorithm:	**OPTIMIZED BRIGHTNESS AND COLOR EQUALIZATION (BCE)**
Input :	Reference image (first frame), transformed the image of the second frame according to the homography matrix calculated to match the perspective of the first frame (YUV format – YUV420 [NV12])
Output :	Brightness and Color corrected stitched panoramic image of input frames

```
begin
1    Starting point for blending: sbp ← assignmentvalue
2    Perform YUV correction process on darker image based on the mean of the Y channel
     of the overlapping region: YUVCORRECTION(LI,lw,lh, RI,rw,rh,sbp)
4    Blend YUV corrected images after calculation of the weights: BLENDIMAGES(LI,
     RI, OI,sbp,lw,lh, RI,rw,rh)
5    Copy right image beyond the overlapping region and save output panoramic image
end
```

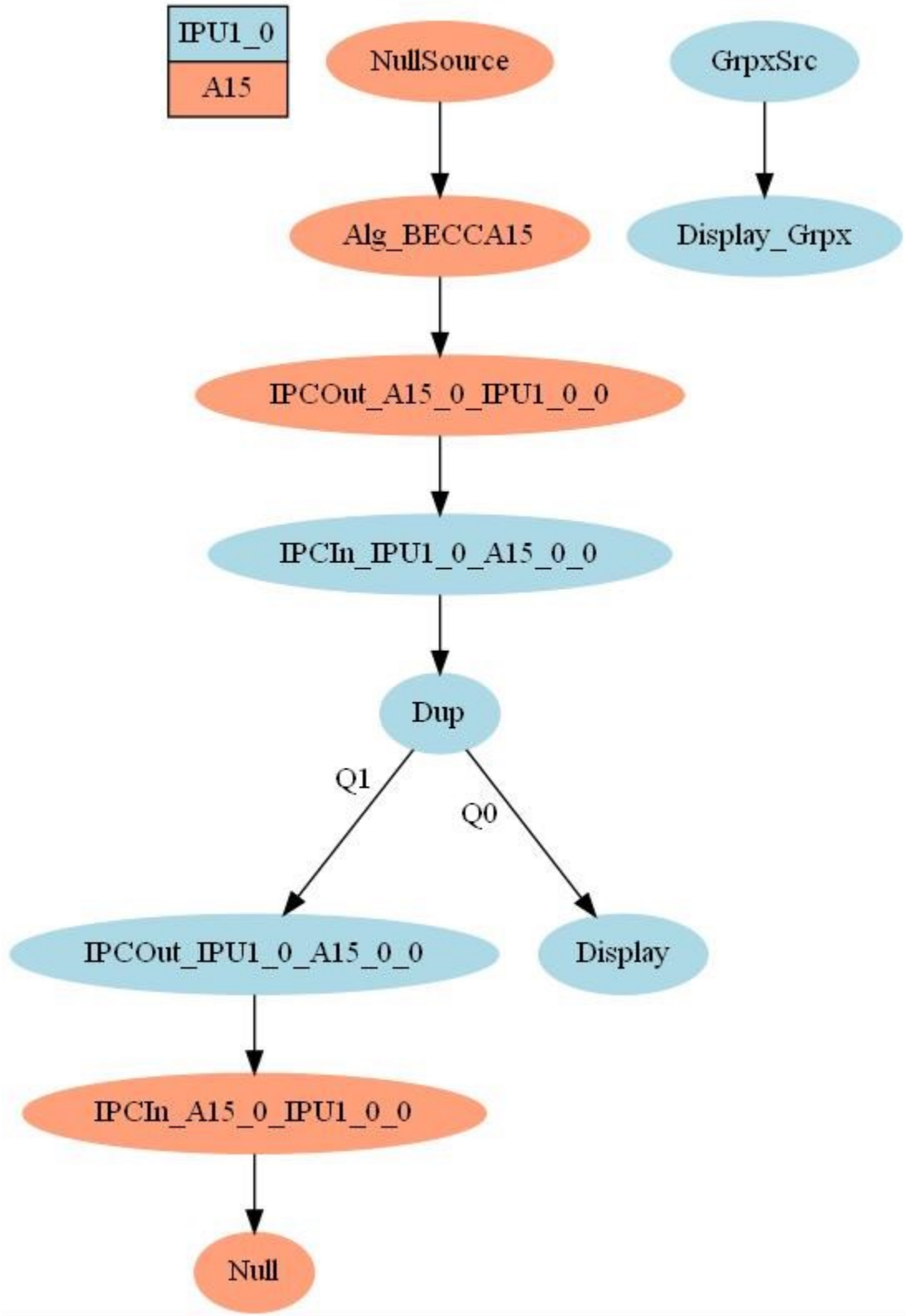
ANNEX G: BCE Implementation using IPU processor



ANNEX H: BCE Implementation using multiple DSP processors



ANNEX I: BCE Implementation using A15 processor



ANNEX J: BRISQUE Scores for 30 Images

Frame/ Method	BRISQUE scores			
	GG	GLT	LTG	LTLT
0	42.91663	42.94721	42.8402	42.91064
1	42.91785	42.96571	42.98074	42.98593
2	42.91663	42.94721	42.8402	42.91064
3	43.60437	43.58393	43.70227	43.64772
4	41.86818	41.89858	41.90843	41.89703
5	45.74631	45.79106	45.89758	45.91778
6	44.52456	44.59031	44.32411	44.55708
7	44.63417	44.68841	44.48017	44.6968
8	44.39432	44.40976	44.15181	44.3547
9	44.58822	44.60446	44.36573	44.58899
10	44.51284	44.52858	44.32191	44.55768
11	44.33725	44.40656	44.24503	44.39954
12	44.32686	44.373	44.12134	44.2612
13	44.25922	44.33186	44.17517	44.32348
14	44.45693	44.53534	44.3831	44.66651
15	44.2819	44.36575	44.31276	44.4969
16	44.37698	44.45049	44.19398	44.33619
17	44.08374	44.13757	43.86161	43.99708
18	44.80199	44.83393	44.767	44.83353
19	44.51278	44.59749	44.50311	44.69559
20	44.62959	44.7078	44.59131	44.79976
21	44.2819	44.36575	44.31276	44.4969
22	44.62643	44.71043	44.39332	44.58835
23	44.54493	44.53897	44.34066	44.49365
24	44.21893	44.32493	44.36137	44.47686
25	44.66154	44.72769	44.43929	44.69684
26	44.4068	44.49066	44.38142	44.5007
27	44.34124	44.40366	44.1479	44.38117
28	43.94855	44.08494	44.06042	44.17554
29	44.20442	44.28546	44.00554	44.28606

ANNEX K: Subjective Image Quality Test Scores (Raw Scores Data)

Image	User Test Scores(Varies from -3 to 3)																													
Frame0	3	2	1	-1	-1	0	1	-1	0	-1	1	-2	1	-1	-1	-2	-1	-1	-1	-1	2	3	2	1	2	2	1	2	2	-2
Frame1	3	2	1	0	0	0	1	-1	-1	0	-1	-2	1	1	0	0	0	1	0	0	1	0	0	1	2	2	2	0	-1	-1
Frame2	2	2	1	1	-1	0	-1	-1	1	-1	-1	-2	2	1	-1	-1	-1	1	-2	3	2	1	3	1	3	-2	1	2	2	2
Frame3	2	1	1	1	1	1	1	0	0	1	1	-2	-1	1	2	2	1	1	2	2	-1	0	2	2	2	-2	0	-1	-2	1
Frame4	2	1	2	3	0	0	1	0	0	0	1	-1	1	-1	0	-2	0	1	2	2	0	0	1	1	1	3	1	0	-2	-1
Frame5	2	0	1	2	-1	1	1	-1	0	0	-1	-1	2	-1	-1	-1	-1	1	-1	-2	2	2	2	1	2	2	1	2	2	2
Frame6	2	3	2	-1	0	0	0	0	1	0	-1	-2	2	1	-1	-1	0	-2	2	1	0	1	2	2	2	2	2	1	1	2
Frame7	1	2	2	-1	0	0	-1	-1	0	0	-2	-1	-1	3	2	-1	1	1	1	0	0	0	0	0	2	-1	1	0	-1	0
Frame8	2	2	1	-2	-1	0	1	-1	0	0	-1	-2	1	2	1	-2	-1	1	-1	-1	2	1	0	2	-1	2	2	2	2	-2
Frame9	1	1	1	-1	1	1	0	0	2	1	-2	1	-1	-1	1	2	2	1	1	2	-1	0	2	1	2	2	1	-2	-2	1
Frame10	2	1	3	0	0	0	-1	-1	1	0	2	0	0	1	2	-1	-1	1	1	2	0	0	2	1	2	2	1	0	0	-2
Frame11	1	3	1	-1	-1	0	-1	-1	1	0	-1	-2	1	-1	-2	-1	-1	1	-2	2	2	1	1	1	1	3	2	2	2	1
Frame12	0	1	0	1	0	-1	-1	1	0	0	-2	2	1	1	-1	-1	2	-1	-1	-1	0	2	1	2	1	-3	1	1	-1	-2
Frame13	1	1	1	0	0	0	-1	-1	0	0	-1	-2	1	0	-1	-1	0	-1	1	-2	0	0	1	2	1	-1	2	0	-2	2
Frame14	2	1	1	-2	-1	-1	-1	-1	-1	-2	-1	-1	1	-1	-1	-2	-1	-2	-1	2	2	1	2	1	1	1	1	2	-2	-2
Frame15	-1	1	0	-1	1	1	1	-1	1	0	1	1	-1	2	2	1	1	2	1	1	-1	-1	0	2	2	-2	-1	-2	-2	-2
Frame16	1	1	1	2	0	0	-2	-1	0	0	-1	-3	-2	-1	-3	0	0	-2	1	2	0	0	0	1	1	2	1	0	0	-1
Frame17	2	1	1	1	-1	-1	-2	-1	0	-1	-1	-1	2	1	-2	-1	-1	-1	-1	-2	2	2	2	1	-1	2	2	2	1	-1
Frame18	3	1	1	3	0	-1	-1	0	0	0	-2	2	-1	1	1	0	-1	-1	0	0	1	-1	1	1	2	3	2	2	-2	2
Frame19	3	0	0	3	0	0	-1	0	0	0	-1	1	1	1	0	-2	-1	-1	1	2	0	0	2	1	-2	2	1	-1	0	1
Frame20	2	0	1	2	-1	-1	-1	-1	-1	-1	-1	-1	1	2	-2	-1	-1	-2	-1	-2	2	2	1	0	2	2	2	2	-2	1
Frame21	2	1	0	3	1	1	-1	0	0	1	1	1	-2	-1	3	1	-1	1	2	0	-1	-1	1	1	2	1	-1	-2	-1	-1
Frame22	3	0	1	3	0	0	-1	-1	-1	0	-2	1	-1	0	-1	-1	-1	1	0	0	0	0	1	1	2	-1	0	0	-1	-2
Frame23	2	1	0	2	-1	-1	1	1	-2	0	-3	1	1	1	-2	-1	-1	-2	-2	-2	2	2	0	2	2	3	2	2	2	-2
Frame24	2	0	1	2	-1	-1	0	0	0	-1	-1	-2	1	-1	-1	-2	-2	-1	1	1	2	1	0	2	2	2	1	2	1	1
Frame25	2	1	0	-1	0	0	-1	-1	0	0	-2	0	0	0	-1	-1	-1	-1	0	-1	0	0	2	2	2	1	1	0	-2	-1
Frame26	2	1	1	-1	-1	-1	-2	-1	-1	-1	-2	-2	2	2	-1	-1	-1	-1	-1	-1	2	1	1	1	1	1	2	2	-2	-2
Frame27	0	0	0	1	1	1	1	1	0	1	2	-2	-2	1	2	1	2	1	1	2	-1	-1	1	0	2	-2	-1	-2	-2	-1
Frame28	1	2	0	1	0	0	-1	-1	0	0	-1	1	-2	1	-2	-3	-1	-1	1	1	0	0	1	2	2	2	1	0	2	-2
Frame29	3	1	0	1	-1	-1	1	-1	0	0	-1	-1	3	1	-2	-2	-1	-2	-1	-1	1	2	0	1	2	2	2	2	-2	-1
Frame30	2	1	2	-1	-1	-1	-2	-1	0	-1	-2	-2	1	1	-2	-2	-1	-1	0	1	1	2	0	2	1	2	2	2	2	1
Frame31	3	1	2	-1	0	0	-1	-1	0	0	-1	-1	-1	-1	0	-1	-1	2	0	0	0	0	1	2	2	2	1	0	0	-2
Frame32	1	2	3	3	-1	-1	-2	-1	-1	-1	-2	-1	1	1	-2	-1	-1	-2	-1	-1	2	2	2	2	2	2	2	2	-1	-2
Frame33	0	0	0	-1	1	1	-1	0	1	1	-2	1	-2	-1	2	2	1	2	1	1	-1	-1	0	1	2	-2	-2	-2	-2	-1
Frame34	2	1	1	0	0	0	-1	-1	-1	0	-1	0	1	1	-2	-2	1	-2	0	2	0	0	1	1	1	2	1	0	-1	-2
Frame35	3	1	1	1	-1	-1	-1	-1	0	0	-1	-3	2	2	-3	-1	-1	-2	-1	2	2	2	1	1	2	1	2	1	2	-2
Frame36	1	1	2	1	-1	-1	-1	0	-1	0	-1	-2	2	1	-2	-1	-1	-1	-1	0	2	2	1	1	2	2	2	2	1	1
Frame37	3	3	1	2	0	0	-1	-1	-1	0	-3	-1	-1	-1	-3	-1	-2	-2	0	1	0	1	1	1	1	2	1	0	-2	-2
Frame38	3	3	3	2	-1	-1	-2	0	-1	-1	-3	-2	-1	-1	-1	-1	-1	-1	-1	2	3	2	1	1	3	2	2	2	-2	-2
Frame39	2	1	3	2	1	1	2	0	1	1	1	1	-2	2	1	1	1	1	1	-2	-1	-1	2	1	3	2	-2	-2	-2	-2

Frame40	1	3	3	2	0	0	-1	0	0	0	-3	-1	1	0	-1	-1	-1	-1	0	1	2	0	1	1	2	2	0	-1	-1	-2
Frame41	3	3	3	-1	-1	-1	-2	-2	0	0	-2	-2	2	1	-3	-2	-1	-2	-1	1	2	2	2	2	2	2	1	2	2	2
Frame42	2	3	3	-1	-1	-1	1	0	-1	0	-2	-1	2	3	-1	-2	1	1	2	-1	2	2	2	1	1	3	2	2	2	-2
Frame43	2	2	3	-1	0	0	-2	-1	-1	0	-1	0	1	2	-1	-2	0	1	0	-1	0	0	1	2	2	2	0	1	-2	-2
Frame44	2	3	3	-1	-1	-1	-2	-1	-1	0	-3	-2	1	2	-2	-2	-1	-1	-1	-2	2	2	1	1	2	2	2	2	-2	-2
Frame45	1	2	3	3	1	1	-2	1	0	1	1	-2	-1	-1	2	1	-1	2	2	2	-2	-1	1	0	2	3	-1	-2	-1	-2
Frame46	2	2	3	-1	0	0	-2	-1	0	0	-2	-1	-2	2	0	-2	1	-2	0	-1	0	0	0	1	2	3	1	0	0	-2
Frame47	3	1	3	-1	-1	-1	-1	-1	0	-1	-2	-2	2	2	-1	-3	-1	-1	-1	0	2	2	1	2	1	2	2	2	2	-1
Frame48	1	2	0	1	-1	-1	-1	0	-1	0	-3	-1	1	1	-2	-2	-1	1	-1	2	2	2	1	1	1	2	2	2	1	-1
Frame49	2	3	3	0	0	0	1	0	0	0	-2	-2	-1	2	1	2	0	-1	1	1	0	0	1	2	2	2	2	-1	0	-2
Frame50	2	2	3	2	0	-1	-1	1	-1	0	-2	-2	1	1	-2	-1	1	-2	-1	-1	1	2	2	1	2	3	2	2	2	-2
Frame51	2	1	3	0	0	1	-1	0	1	0	2	1	2	1	3	1	1	2	1	2	0	-1	0	1	2	2	-1	-1	1	-2
Frame52	2	2	0	1	0	0	-1	0	0	0	-2	0	0	2	-1	0	1	-2	-1	1	0	0	1	2	2	-2	1	1	-1	-2
Frame53	2	3	2	-1	0	-1	-2	0	-1	0	-3	-1	1	2	-1	-1	1	-2	0	2	1	2	1	2	2	2	2	1	-1	2
Frame54	2	2	1	-1	-1	-1	1	-1	-1	-1	-1	-1	1	3	-2	-2	-1	-1	1	1	2	2	1	2	2	1	1	2	-1	1
Frame55	1	0	1	-1	0	0	-1	-2	0	0	0	1	0	-1	-1	2	-1	-1	-1	0	0	2	0	1	1	2	0	0	-1	-2
Frame56	1	2	2	-1	0	-1	-1	-1	-1	0	-1	0	-2	1	-2	0	1	-1	2	-2	0	2	0	1	1	2	2	1	2	-2
Frame57	0	2	2	0	0	1	1	1	1	0	1	-2	1	2	3	1	-1	2	2	1	-1	-1	1	1	2	2	-1	-2	1	-1
Frame58	1	1	0	0	0	0	-2	-1	0	0	-1	1	0	0	-2	-2	1	-1	0	2	0	0	0	1	-1	3	1	1	0	-2
Frame59	1	0	3	0	0	-1	-1	0	-1	0	-1	-1	1	1	-1	-1	1	2	0	-1	1	2	1	2	1	3	2	1	-1	2
Frame60	2	3	3	-1	-1	-1	1	-1	-1	0	-2	-2	1	2	-2	-1	-1	-2	-1	-1	1	2	2	1	2	3	2	1	2	2
Frame61	3	1	1	2	0	0	-1	0	0	0	0	-2	1	0	0	1	-1	1	-1	-1	0	0	1	-1	-2	2	1	1	-2	-1
Frame62	3	0	1	1	0	-1	-1	0	0	0	-1	-2	1	3	-2	-1	-1	-1	-1	-1	0	1	2	1	2	3	2	2	0	1
Frame63	3	-1	0	3	0	1	-1	0	0	0	1	-1	1	-1	2	1	-1	-2	0	3	0	-1	1	-1	-2	2	-2	-2	-2	-2
Frame64	3	2	0	-1	0	0	-1	0	0	0	-1	1	-2	-1	-2	2	-1	-1	-1	-1	0	0	0	1	2	3	1	1	-2	-2
Frame65	3	2	1	1	0	-1	1	0	0	0	-2	-2	1	-1	1	1	-1	-1	-2	-1	1	2	1	1	2	2	2	2	0	-2
Frame66	2	3	1	-1	-1	-1	-1	0	-1	-1	-2	-2	1	3	-2	-2	-1	-1	2	-2	2	2	1	2	-1	3	2	1	2	-2
Frame67	3	2	0	-1	0	0	-2	-1	0	0	-2	2	2	-1	-2	-2	0	-1	0	1	0	0	1	1	2	2	1	0	-2	-2
Frame68	2	2	1	-2	-1	-1	-2	0	-1	-1	-1	1	1	-1	-2	-2	1	-1	-1	-1	2	1	1	1	2	3	1	2	1	-1
Frame69	3	0	0	1	1	1	-2	-1	1	1	1	-2	-1	-1	1	1	1	2	1	1	-1	-1	1	2	2	2	-1	-2	-2	1
Frame70	2	0	0	3	0	0	-2	-1	0	0	-1	-2	3	0	0	-1	-1	-1	0	1	0	0	2	1	2	2	1	0	2	-2
Frame71	3	2	2	-1	-1	-1	-2	0	0	0	-2	-1	2	1	1	-1	-1	-2	0	1	2	2	1	1	3	2	2	2	2	-1
Frame72	3	1	2	1	-1	-1	-2	-1	-1	0	-2	-2	1	1	-2	-2	-2	-1	-1	-1	2	2	2	1	2	2	2	2	1	-1
Frame73	2	1	0	-1	0	0	-1	-1	0	0	-1	-1	-1	-2	2	-1	-1	-1	0	3	0	0	2	2	1	2	1	1	-2	2
Frame74	2	1	0	-1	-1	0	0	-1	0	-1	-1	-2	2	1	1	-1	-1	-1	-1	1	2	2	0	1	2	2	1	2	-2	-1
Frame75	2	0	0	3	1	0	2	-1	0	1	-1	-2	-1	1	1	2	-1	-1	1	1	-1	0	1	1	1	-2	-1	-2	-2	1
Frame76	1	1	0	0	0	0	0	-1	1	0	-1	1	-2	2	1	-1	0	1	0	0	0	-1	0	1	2	1	1	0	-2	-1
Frame77	3	0	0	1	-1	0	-2	0	0	0	-2	-1	1	0	-2	-1	1	-1	2	-1	1	2	0	2	2	-1	1	2	-2	-1
Frame78	3	3	2	2	-1	-1	1	-1	-1	-1	-2	-1	1	1	-1	-1	-2	-1	-1	2	2	2	1	1	2	3	2	2	2	-1
Frame79	2	2	1	-1	0	0	-1	-1	0	0	-1	2	2	-1	-2	0	0	-2	0	0	0	2	2	2	2	2	2	0	-2	-1
Frame80	1	1	1	-1	-1	-1	-1	-1	-1	0	-2	-2	2	1	-1	-3	-1	-1	-1	1	2	2	0	1	1	2	2	1	-2	-2

Frame81	2	-1	2	2	1	1	0	1	0	0	1	1	-1	-1	1	1	-1	1	0	2	-1	-1	0	0	2	2	-2	-1	-2	-2
Frame82	2	2	3	-1	0	0	-1	-1	0	0	-2	0	1	-1	-3	0	-1	-1	0	2	0	0	2	1	2	3	-1	1	-1	-1
Frame83	2	2	2	-1	-1	-1	-1	-1	-1	0	-2	-2	2	1	-2	-2	1	-1	-1	1	2	1	1	2	1	2	2	2	2	-2
Frame84	1	2	3	2	-1	-1	1	-1	-1	-1	-3	-1	3	1	-2	-2	-2	-2	-1	2	2	1	1	1	2	3	2	2	-2	-2
Frame85	1	0	1	-1	0	0	-1	-1	-1	0	-2	-2	-2	0	-1	-1	0	-2	0	0	0	0	1	2	-1	3	0	1	-2	-2
Frame86	1	2	1	-1	-1	-1	-1	-1	-1	0	-3	1	2	2	-2	-1	-1	-1	-1	0	2	2	1	1	2	2	2	2	-1	-2
Frame87	1	1	0	1	1	1	1	1	1	1	1	0	1	-1	2	1	1	2	2	2	-1	-1	2	2	-1	2	-2	-2	-2	-2
Frame88	1	0	0	1	0	0	-1	1	0	0	-1	1	-1	-1	0	-1	0	1	1	1	0	0	0	1	1	2	1	0	-1	0
Frame89	2	1	0	-3	-1	-1	-2	-1	-1	-1	-2	-2	1	1	-2	-1	-1	-2	-1	-1	3	2	1	1	1	2	2	2	1	-1

ANNEX L: Win (1)/Loss (0)/Inconclusive (2) Assignment to the top frame

Image	User Votes(Sorted by Win/Loss/Inconclusive)																												
Frame0	1	1	1	0	0	2	1	0	2	0	1	0	1	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	0
Frame1	1	1	1	2	2	2	1	0	0	2	0	0	1	1	2	2	2	1	2	2	1	2	2	1	1	1	1	2	0	0
Frame2	1	1	1	1	0	2	0	0	1	0	0	0	1	1	0	0	0	1	0	1	1	1	1	1	1	0	1	1	1	1
Frame3	1	1	1	1	1	1	1	2	2	1	1	0	0	1	1	1	1	1	1	1	0	2	1	1	1	0	2	0	0	1
Frame4	1	1	1	1	2	2	1	2	2	2	1	0	1	0	2	0	2	1	1	1	2	2	1	1	1	1	1	2	0	0
Frame5	1	2	1	1	0	1	1	0	2	2	0	0	1	0	0	0	0	1	0	0	1	1	1	1	1	1	1	1	1	1
Frame6	1	1	1	0	2	2	2	2	1	2	0	0	1	1	0	0	2	0	1	1	2	1	1	1	1	1	1	1	1	1
Frame7	1	1	1	0	2	2	0	0	2	2	0	0	0	1	1	0	1	1	1	2	2	2	2	2	1	0	1	2	0	2
Frame8	1	1	1	0	0	2	1	0	2	2	0	0	1	1	1	0	0	1	0	0	1	1	2	1	0	1	1	1	1	0
Frame9	1	1	1	0	1	1	2	2	1	1	0	1	0	0	1	1	1	1	1	1	0	2	1	1	1	1	1	0	0	1
Frame10	1	1	1	2	2	2	0	0	1	2	1	2	2	1	1	0	0	1	1	1	2	2	1	1	1	1	1	2	2	0
Frame11	1	1	1	0	0	2	0	0	1	2	0	0	1	0	0	0	0	1	0	1	1	1	1	1	1	1	1	1	1	1
Frame12	2	1	2	1	2	0	0	1	2	2	0	1	1	1	0	0	1	0	0	0	2	1	1	1	1	0	1	1	0	0
Frame13	1	1	1	2	2	2	0	0	2	2	0	0	1	2	0	0	2	0	1	0	2	2	1	1	1	0	1	2	0	1
Frame14	1	1	1	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	1	1	1	1	1	1	1	1	0	0	
Frame15	0	1	2	0	1	1	1	0	1	2	1	1	0	1	1	1	1	1	1	1	0	0	2	1	1	0	0	0	0	0
Frame16	1	1	1	1	2	2	0	0	2	2	0	0	0	0	0	2	2	0	1	1	2	2	2	1	1	1	1	2	2	0
Frame17	1	1	1	1	0	0	0	0	2	0	0	0	1	1	0	0	0	0	0	0	1	1	1	1	0	1	1	1	1	0
Frame18	1	1	1	1	2	0	0	2	2	2	0	1	0	1	1	2	0	0	2	2	1	0	1	1	1	1	1	1	0	1
Frame19	1	2	2	1	2	2	0	2	2	2	0	1	1	1	2	0	0	0	1	1	2	2	1	1	0	1	1	0	2	1
Frame20	1	2	1	1	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	1	1	1	2	1	1	1	1	0	1
Frame21	1	1	2	1	1	1	0	2	2	1	1	1	0	0	1	1	0	1	1	2	0	0	1	1	1	1	0	0	0	0
Frame22	1	2	1	1	2	2	0	0	0	2	0	1	0	2	0	0	0	1	2	2	2	2	1	1	1	0	2	2	0	0
Frame23	1	1	2	1	0	0	1	1	0	2	0	1	1	1	0	0	0	0	0	0	1	1	2	1	1	1	1	1	1	0
Frame24	1	2	1	1	0	0	2	2	2	0	0	0	1	0	0	0	0	0	1	1	1	1	2	1	1	1	1	1	1	1
Frame25	1	1	2	0	2	2	0	0	2	2	0	2	2	2	0	0	0	0	2	0	2	2	1	1	1	1	1	2	0	0

Frame26	1	1	1	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	1	1	1	1	1	1	1	0	0
Frame27	2	2	2	1	1	1	1	1	2	1	1	0	0	1	1	1	1	1	1	1	0	0	1	2	1	0	0	0	0	0
Frame28	1	1	2	1	2	2	0	0	2	2	0	1	0	1	0	0	0	0	1	1	2	2	1	1	1	1	1	2	1	0
Frame29	1	1	2	1	0	0	1	0	2	2	0	0	1	1	0	0	0	0	0	0	1	1	2	1	1	1	1	1	0	0
Frame30	1	1	1	0	0	0	0	0	2	0	0	0	1	1	0	0	0	0	2	1	1	1	2	1	1	1	1	1	1	1
Frame31	1	1	1	0	2	2	0	0	2	2	0	0	0	0	2	0	0	1	2	2	2	2	1	1	1	1	1	2	2	0
Frame32	1	1	1	1	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	1	1	1	1	1	1	1	1	0	0
Frame33	2	2	2	0	1	1	0	2	1	1	0	1	0	0	1	1	1	1	1	1	0	0	2	1	1	0	0	0	0	0
Frame34	1	1	1	2	2	2	0	0	0	2	0	2	1	1	0	0	1	0	2	1	2	2	1	1	1	1	1	2	0	0
Frame35	1	1	1	1	0	0	0	0	2	2	0	0	1	1	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	0
Frame36	1	1	1	1	0	0	0	2	0	2	0	0	1	1	0	0	0	0	0	2	1	1	1	1	1	1	1	1	1	1
Frame37	1	1	1	1	2	2	0	0	0	2	0	0	0	0	0	0	0	0	2	1	2	1	1	1	1	1	1	2	0	0
Frame38	1	1	1	1	0	0	0	2	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	0	0
Frame39	1	1	1	1	1	1	1	2	1	1	1	1	0	1	1	1	1	1	1	0	0	0	1	1	1	1	0	0	0	0
Frame40	1	1	1	1	2	2	0	2	2	2	0	0	1	2	0	0	0	0	2	1	1	2	1	1	1	1	2	0	0	0
Frame41	1	1	1	0	0	0	0	0	2	2	0	0	1	1	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1
Frame42	1	1	1	0	0	0	1	2	0	2	0	0	1	1	0	0	1	1	1	0	1	1	1	1	1	1	1	1	1	0
Frame43	1	1	1	0	2	2	0	0	0	2	0	2	1	1	0	0	2	1	2	0	2	2	1	1	1	1	2	1	0	0
Frame44	1	1	1	0	0	0	0	0	0	2	0	0	1	1	0	0	0	0	0	0	1	1	1	1	1	1	1	1	0	0
Frame45	1	1	1	1	1	1	0	1	2	1	1	0	0	0	1	1	0	1	1	1	0	0	1	2	1	1	0	0	0	0
Frame46	1	1	1	0	2	2	0	0	2	2	0	0	0	1	2	0	1	0	2	0	2	2	2	1	1	1	1	2	2	0
Frame47	1	1	1	0	0	0	0	0	2	0	0	0	1	1	0	0	0	0	0	2	1	1	1	1	1	1	1	1	1	0
Frame48	1	1	2	1	0	0	0	2	0	2	0	0	1	1	0	0	0	1	0	1	1	1	1	1	1	1	1	1	1	0
Frame49	1	1	1	2	2	2	1	2	2	2	0	0	0	1	1	1	2	0	1	1	2	2	1	1	1	1	1	0	2	0
Frame50	1	1	1	1	2	0	0	1	0	2	0	0	1	1	0	0	1	0	0	0	1	1	1	1	1	1	1	1	1	0
Frame51	1	1	1	2	2	1	0	2	1	2	1	1	1	1	1	1	1	1	1	1	2	0	2	1	1	1	0	0	1	0
Frame52	1	1	2	1	2	2	0	2	2	2	0	2	2	1	0	2	1	0	0	1	2	2	1	1	1	0	1	1	0	0
Frame53	1	1	1	0	2	0	0	2	0	2	0	0	1	1	0	0	1	0	2	1	1	1	1	1	1	1	1	0	1	
Frame54	1	1	1	0	0	0	1	0	0	0	0	0	1	1	0	0	0	0	1	1	1	1	1	1	1	1	1	0	1	
Frame55	1	2	1	0	2	2	0	0	2	2	2	1	2	0	0	1	0	0	0	2	2	1	2	1	1	1	2	2	0	0
Frame56	1	1	1	0	2	0	0	0	0	2	0	2	0	1	0	2	1	0	1	0	2	1	2	1	1	1	1	1	1	0
Frame57	2	1	1	2	2	1	1	1	1	2	1	0	1	1	1	1	0	1	1	1	0	0	1	1	1	1	0	0	1	0
Frame58	1	1	2	2	2	2	0	0	2	2	0	1	2	2	0	0	1	0	2	1	2	2	2	1	0	1	1	1	2	0
Frame59	1	2	1	2	2	0	0	2	0	2	0	0	1	1	0	0	1	1	2	0	1	1	1	1	1	1	1	0	1	
Frame60	1	1	1	0	0	0	1	0	0	2	0	0	1	1	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1
Frame61	1	1	1	1	2	2	0	2	2	2	2	0	1	2	2	1	0	1	0	0	2	2	1	0	0	1	1	1	0	0
Frame62	1	2	1	1	2	0	0	2	2	2	0	0	1	1	0	0	0	0	0	0	2	1	1	1	1	1	1	1	2	1
Frame63	1	0	2	1	2	1	0	2	2	2	1	0	1	0	1	1	0	0	2	1	2	0	1	0	0	1	0	0	0	0
Frame64	1	1	2	0	2	2	0	2	2	2	0	1	0	0	0	1	0	0	0	0	2	2	2	1	1	1	1	1	0	0
Frame65	1	1	1	1	2	0	1	2	2	2	0	0	1	0	1	1	0	0	0	0	1	1	1	1	1	1	1	1	2	0
Frame66	1	1	1	0	0	0	0	2	0	0	0	0	1	1	0	0	0	0	1	0	1	1	1	1	0	1	1	1	1	0
Frame67	1	1	2	0	2	2	0	0	2	2	0	1	1	0	0	0	2	0	2	1	2	2	1	1	1	1	1	2	0	0
Frame68	1	1	1	0	0	0	0	2	0	0	0	1	1	0	0	0	1	0	0	0	1	1	1	1	1	1	1	1	1	0
Frame69	1	2	2	1	1	1	0	0	1	1	1	0	0	0	1	1	1	1	1	1	0	0	1	1	1	1	0	0	0	1

Frame70	1	2	2	1	2	2	0	0	2	2	0	0	1	2	2	0	0	0	2	1	2	2	1	1	1	1	1	2	1	0
Frame71	1	1	1	0	0	0	0	2	2	2	0	0	1	1	1	0	0	0	2	1	1	1	1	1	1	1	1	1	1	0
Frame72	1	1	1	1	0	0	0	0	0	2	0	0	1	1	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	0
Frame73	1	1	2	0	2	2	0	0	2	2	0	0	0	0	1	0	0	0	2	1	2	2	1	1	1	1	1	0	1
Frame74	1	1	2	0	0	2	2	0	2	0	0	0	1	1	1	0	0	0	0	1	1	1	2	1	1	1	1	0	0
Frame75	1	2	2	1	1	2	1	0	2	1	0	0	0	1	1	1	0	0	1	1	0	2	1	1	1	0	0	0	0	1
Frame76	1	1	2	2	2	2	2	0	1	2	0	1	0	1	1	0	2	1	2	2	2	0	2	1	1	1	1	2	0	0
Frame77	1	2	2	1	0	2	0	2	2	2	0	0	1	2	0	0	1	0	1	0	1	1	2	1	1	0	1	1	0	0
Frame78	1	1	1	1	0	0	1	0	0	0	0	0	1	1	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	0
Frame79	1	1	1	0	2	2	0	0	2	2	0	1	1	0	0	2	2	0	2	2	2	1	1	1	1	1	1	2	0	0
Frame80	1	1	1	0	0	0	0	0	0	2	0	0	1	1	0	0	0	0	0	1	1	1	2	1	1	1	1	1	0	0
Frame81	1	0	1	1	1	1	2	1	2	2	1	1	0	0	1	1	0	1	2	1	0	0	2	2	1	1	0	0	0	0
Frame82	1	1	1	0	2	2	0	0	2	2	0	2	1	0	0	2	0	0	2	1	2	2	1	1	1	1	0	1	0	0
Frame83	1	1	1	0	0	0	0	0	0	2	0	0	1	1	0	0	1	0	0	1	1	1	1	1	1	1	1	1	1	0
Frame84	1	1	1	1	0	0	1	0	0	0	0	0	1	1	0	0	0	0	0	1	1	1	1	1	1	1	1	0	0
Frame85	1	2	1	0	2	2	0	0	0	2	0	0	0	2	0	0	2	0	2	2	2	2	1	1	0	1	2	1	0	0
Frame86	1	1	1	0	0	0	0	0	0	2	0	1	1	1	0	0	0	0	0	2	1	1	1	1	1	1	1	1	0	0
Frame87	1	1	2	1	1	1	1	1	1	1	1	2	1	0	1	1	1	1	1	1	0	0	1	1	0	1	0	0	0	0
Frame88	1	2	2	1	2	2	0	1	2	2	0	1	0	0	2	0	2	1	1	1	2	2	2	1	1	1	1	2	0	2
Frame89	1	1	2	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	0

ANNEX M: MATLAB Script for parsing sorted user scores

```
Line  Code
1     [num,txt,raw] = xlsread("votes.xlsx");
2     CompareBCE = zeros(90,4);
3     CompareBCEAll = zeros(30,90,4);
4     dataSave = zeros(30,4);
5     Matrix = zeros(4,4);
6     for iterUsers = 1:30
7     for iterComparisons = 1:15
8     temp = 6*(iterComparisons-1)+1;
9     if(num(temp,iterUsers) == 1)
10    CompareBCE(temp,1) = CompareBCE(temp,1)+1;
11    CompareBCEAll(iterUsers,temp,1) = CompareBCEAll(iterUsers,temp,1)+1;
12    Matrix(1,2) = Matrix(1,2) + 1;
13    elseif (num(temp,iterUsers) == 0)
14    CompareBCE(temp,2) = CompareBCE(temp,2)+1;
15    CompareBCEAll(iterUsers,temp,2) = CompareBCEAll(iterUsers,temp,2)+1;
16    Matrix(2,1) = Matrix(2,1) + 1;
17    end
18    temp = temp +1;
19    if(num(temp,iterUsers) == 1)
20    CompareBCE(temp,1) = CompareBCE(temp,1)+1;
21    CompareBCEAll(iterUsers,temp,1) = CompareBCEAll(iterUsers,temp,1)+1;
22    Matrix(1,3) = Matrix(1,3) + 1;
23    elseif (num(temp,iterUsers) == 0)
24    CompareBCE(temp,3) = CompareBCE(temp,3)+1;
25    CompareBCEAll(iterUsers,temp,3) = CompareBCEAll(iterUsers,temp,3)+1;
26    Matrix(3,1) = Matrix(3,1) + 1;
27    end
28    temp = temp +1;
29    if(num(temp,iterUsers) == 1)
30    CompareBCE(temp,1) = CompareBCE(temp,1)+1;
31    CompareBCEAll(iterUsers,temp,1) = CompareBCEAll(iterUsers,temp,1)+1;
32    Matrix(1,4) = Matrix(1,4) + 1;
33    elseif (num(temp,iterUsers) == 0)
34    CompareBCE(temp,4) = CompareBCE(temp,4)+1;
35    CompareBCEAll(iterUsers,temp,4) = CompareBCEAll(iterUsers,temp,4)+1;
36    Matrix(4,1) = Matrix(4,1) + 1;
37    end
38    temp = temp +1;
39    if(num(temp,iterUsers) == 1)
40    CompareBCE(temp,2) = CompareBCE(temp,2)+1;
41    CompareBCEAll(iterUsers,temp,2) = CompareBCEAll(iterUsers,temp,2)+1;
42    Matrix(2,3) = Matrix(2,3) + 1;
43    elseif (num(temp,iterUsers) == 0)
44    CompareBCE(temp,3) = CompareBCE(temp,3)+1;
45    CompareBCEAll(iterUsers,temp,3) = CompareBCEAll(iterUsers,temp,3)+1;
46    Matrix(3,2) = Matrix(3,2) + 1;
47    end
48    temp = temp +1;
49    if(num(temp,iterUsers) == 1)
50    CompareBCE(temp,2) = CompareBCE(temp,2)+1;
51    CompareBCEAll(iterUsers,temp,2) = CompareBCEAll(iterUsers,temp,2)+1;
52    Matrix(2,4) = Matrix(2,4) + 1;
53    elseif (num(temp,iterUsers) == 0)
```

```
    CompareBCE(temp,4) = CompareBCE(temp,4)+1;
    CompareBCEAll(iterUsers,temp,4) = CompareBCEAll(iterUsers,temp,4)+1;
    Matrix(4,2) = Matrix(4,2) + 1;
    end
    temp = temp +1;
    if(num(temp,iterUsers) == 1)
    CompareBCE(temp,3) = CompareBCE(temp,3)+1;
    CompareBCEAll(iterUsers,temp,3) = CompareBCEAll(iterUsers,temp,3)+1;
    Matrix(3,4) = Matrix(3,4) + 1;
    elseif (num(temp,iterUsers) == 0)
    CompareBCE(temp,4) = CompareBCE(temp,4)+1;
    CompareBCEAll(iterUsers,temp,4) = CompareBCEAll(iterUsers,temp,4)+1;
    Matrix(4,3) = Matrix(4,3) + 1;
    end
    end
    if (iterUsers==0 )
    dataSave(iterUsers,:) = sum(CompareBCE,1);
    else
    dataSave(iterUsers,:) = sum(CompareBCE,1) - sum(dataSave,1);
    end
    end
    outcombined = sum(CompareBCE,1);
    outindividual = sum(CompareBCEAll,[1 2]);
    filename = 'testdata.xlsx';
    writematrix(dataSave,filename,'Sheet',1,'Range','A1')
    writematrix(Matrix,filename,'Sheet',2,'Range','A1')
    a = 1;
    prec = 1e-8;
    btem(Matrix,a,prec);
    N_Gibbs = 1000;
    N_burn = 100;
    btgibbs(Matrix,1,N_Gibbs,N_burn);
```

ANNEX N: Bradley-Terry Model Prediction for User Preference Probabilities

Entry		Wins				Probabilities			
	Set	GG	GLT	LTG	LTLT	GG	GLT	LTG	LTLT
User1	Test1	44	26	16	0	0.5116	0.3023	0.1860	0.0000
	Test2	39	21	14	0	0.5270	0.2838	0.1892	0.0000
User2	Test1	38	15	11	0	0.5938	0.2344	0.1719	0.0000
	Test2	17	25	19	19	0.2125	0.3125	0.2375	0.2375
User3	Test1	0	24	0	24	0.0000	0.5000	0.0000	0.5000
	Test2	0	27	1	24	0.0000	0.5192	0.0192	0.4615
User4	Test1	9	15	23	37	0.1071	0.1786	0.2738	0.4405
	Test2	2	13	16	30	0.0328	0.2131	0.2623	0.4918
User5	Test1	2	19	5	18	0.0455	0.4318	0.1136	0.4091
	Test2	0	17	0	11	0.0000	0.6071	0.0000	0.3929
User6	Test1	1	28	16	43	0.0114	0.3182	0.1818	0.4886
	Test2	8	26	17	31	0.0976	0.3171	0.2073	0.3780
User7	Test1	34	10	32	9	0.4000	0.1176	0.3765	0.1059
	Test2	30	15	25	11	0.3704	0.1852	0.3086	0.1358
User8	Test1	6	31	10	35	0.0732	0.3780	0.1220	0.4268
	Test2	3	30	11	39	0.0361	0.3614	0.1325	0.4699
User9	Test1	6	25	17	30	0.0769	0.3205	0.2179	0.3846
	Test2	9	31	15	35	0.1000	0.3444	0.1667	0.3889
User10	Test1	10	26	3	27	0.1515	0.3939	0.0455	0.4091
	Test2	17	30	11	19	0.2208	0.3896	0.1429	0.2468
User11	Test1	27	1	28	0	0.4821	0.0179	0.5000	0.0000
	Test2	32	1	27	1	0.5246	0.0164	0.4426	0.0164
User12	Test1	36	20	12	0	0.5294	0.2941	0.1765	0.0000
	Test2	42	26	17	0	0.4941	0.3059	0.2000	0.0000
User13	Test1	40	28	19	3	0.4444	0.3111	0.2111	0.0333
	Test2	41	24	21	4	0.4556	0.2667	0.2333	0.0444
User14	Test1	42	13	28	1	0.5000	0.1548	0.3333	0.0119
	Test2	34	4	32	1	0.4789	0.0563	0.4507	0.0141
User15	Test1	16	8	35	22	0.1975	0.0988	0.4321	0.2716
	Test2	13	12	27	36	0.1477	0.1364	0.3068	0.4091