Received March 13, 2014, accepted April 8, 2014, date of current version May 21, 2014.

Digital Object Identifier 10.1109/ACCESS.2014.2322101

Enhancing Stereo Matching With Classification

MOHAMMED BAYDOUN AND MOHAMAD ADNAN AL-ALAOUI, (Senior Member, IEEE)
Department of Electrical and Computer Engineering, American University of Beirut, Beirut 1107 2020, Lebanon

Corresponding author: M. Baydoun (mhb13@aub.edu.lb)

This work was supported in part by the Lebanese National Council for Scientific Research and in part by Nvidia through the GPUs.

ABSTRACT This paper presents a novel approach that employs classification to enhance the accuracy of the stereo matching problem. First, the images are treated in order to improve their pixel to pixel correspondence and reduce illumination differences. After that, stereo matching is addressed using different methods with emphasis on local ones like the sum of absolute distances and normalized cross correlation. Other state-of-the-art approaches are also considered. Then, and for every pixel, different features are computed from the input stereo image and the initially found depth map. Afterward, boosting and neural networks, as classification methods, are used to handle occlusion and enhance stereo matching by finding the erroneous disparity values. These values are then corrected through a completion stage. The accuracy of the proposed implementation improves on the problem in an efficient manner. A timing analysis of the method is provided to validate the real time performance. This paper further clarifies some of the possible developments based on various discussions.

INDEX TERMS Stereo matching, classification, real time.

I. INTRODUCTION

STEREO matching is one of the most addressed problems in computer vision and image processing [1]. Stereo matching usually utilizes a stereo camera, which is made of two or more lenses producing images to simulate the binocular human vision and estimate 3D positions [2]. There are many possibilities for the lenses, as in the optical axes which can be parallel or converging, or the distance between the lenses which is usually similar to the distance between the human eyes, etc. . .

The aim of stereo matching is to compute the disparity value of each pixel. The disparity is the horizontal difference in pixel positions between matching pixels. Using basic geometry, according to the stereo setup and camera parameters, the disparity leads to the depths of the pixels. The disparity is limited by a certain range or a maximum displacement that differs for different stereo pairs.

Besides stereo matching, there are many methods for determining depth. These include laser scanning [3] and structured light [1], [3]. There are other methods such as 3D from video, using multiple cameras or combinations of several methods, etc. . .

The methods differ in aspects such as setup and accuracy. The laser scanning and structured light methods require an enhanced setup and achieve relatively high accuracy. These methods are used to check the viability of the different algorithms of stereo matching [1]. Thus, while stereo matching is the best in regards to cost, it is the least accurate. Fig. 1 shows the Middlebury [1] evaluation images which are used to check the accuracy of stereo matching algorithms.

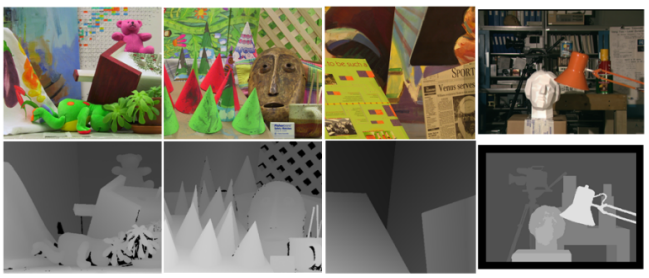


FIGURE 1. Middlebury evaluation images (Top Row), from left to right, Teddy, Cones, Venus, Tsukuba [1], Corresponding disparity map (bottom row).

The main faced problem in stereo matching is occlusion where some pixels appear in one image and disappear in the other(s) due to varying depths. This is generally solved by computing all the disparity maps and validating the matching pixels in the images. In the case of a stereo pair, which is the target here and in considered literature, this is called left to right cross-checking (LRC). The main ideas related to occlusion are summarized in [4].

Other stereo matching problems are mainly related to color inconsistency. For example, the different lenses in the camera could differ in their properties. Another issue lies in the discrete nature of images that does not allow for exact matching, etc. . .

In regards to classification, it is defined as the process of determining the class of a certain input based on the input's features. In this work, we consider supervised classification where the features and classes of the training data are known and used to determine the classifier parameters.

This work presents a new approach that uses classification as a post processing stage to enhance stereo matching by remedying the erroneous disparities. The main contributions and benefits can be summarized as follows:

1) The developed method is generic since it can be applied to any stereo matching algorithm.
2) The approach enhances the matching in real-time.
3) The technique handles incorrect and occluded pixels with or without LRC.
4) The presented ideas are parallelizable enabling fast parallel computations in addition to the existence of several possibilities for further enhancing the system.

This is motivated by several incentives:

1) The first idea is that the implementation of binary classification is fast.
2) Another motivation is that many of the stereo matching methods, especially local ones, usually present a systematic error since errors repeat themselves. This led to amending the errors through the systematic approach of classification.
3) Third, classification can be adjusted to certain scenarios, which renders the approach tweak-able to excel in known conditions.
4) The last major incentive is that classification can be used to target the problem of occlusion. This can reduce the computations since occlusion handling generally requires the computation of two maps to perform LRC.

In essence, classification can and is proven to combine a certain set of measurements for the purpose of improving stereo matching proficiently. And in order to validate the main ideas, a Matlab demo code is provided with the paper.

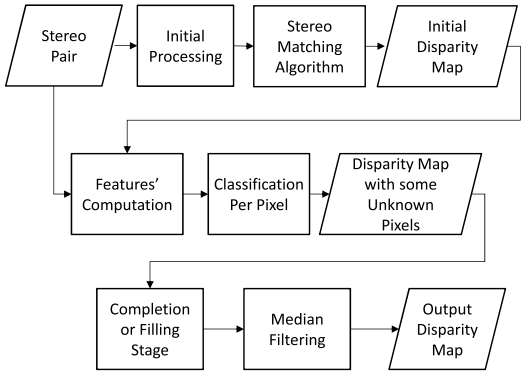


FIGURE 2. Flowchart showing the stages of the proposed system.

The suggested system follows the steps shown in the flowchart of Fig. 2. The method starts by performing initial processing to the stereo pair. Then, a stereo matching algorithm is performed to obtain the required disparity map. After that, the computed map along with the input images are used to compute the chosen features for every pixel. The pixels are then classified as acceptable or not which leads to a disparity map with unknown values. The unidentified values are then filled using a completion stage followed by median filtering to produce the final disparity map.

The rest of the paper is organized as follows; section II reviews related literature. Section III presents the initial approach for stereo matching including the preprocessing and the used methods. Then, section IV considers the contributions of this work in regards to classification and relating it to matching. Section IV also explains the features. Then, section V presents the results. Section VI discusses several issues related to the proposed work such as timing, advantages and limitations. Then, conclusions are presented in section VII.

II. REVIEW

There are many methods to tackle the stereo matching problem and summarizing related work is not possible in a single paper due to the popularity and scatter of the literature [1], [4]. In general, the methods are divided into three types; local, global, and semi-global ones that combine the other two.

The local methods usually implement a measure of the difference between the pixels and are generally the fastest. The three basic local methods are the Sum of Absolute Differences (SAD), the Sum of Squared Differences (SSD), and the Normalized Cross Correlation (NCC).

Fig. 3 presents an example of stereo matching showing the corresponding left and right SSD disparity images of the Teddy [1] stereo pair.

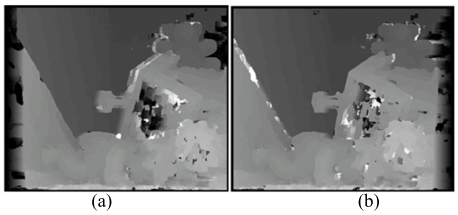


FIGURE 3. Example of obtained disparity map using SSD with *Radius* = 5 (a) left map, (b) right map.

Besides the elementary methods, others employ improved distances such as the Birchfield Tomasi (BT) measure in [5]. Also, many local methods use adaptive measurements like different windows with (SAD) [6] or adaptive support weights (ASW) [7], and combinations of local methods [7], [8]. Some of these are compared in [8]. It is worth mentioning [9], which reformulated the cost aggregation problem of matching from a histogram perspective.

The global matching methods mainly use an implementation of different optimization techniques. These are usually time demanding because they consider the pixel's disparity with the neighboring ones. Some of these rely on graph theory [10]. Others use cooperative optimization [11], [12], nonlinear optimization techniques [13], [14], Markov Chain based methods [15] and Belief Propagation (BP) as in [16]

and [17]. Moreover, [18] presented a combination of BP and Markov methods, with other works like [19], [20] also combining different global approaches.

Additionally, there are semi-global approaches as in [21] in which Mutual Information was used. Also, the work in [22] implemented a new distinctive similarity measure that combined local and global properties.

So, and as noted in [1], [3], [4], [6]–[8] and others, the two key properties of a stereo matching method are accuracy and speed with constantly updated research dealing with these aspects.

Concerning accuracy, many methods were proposed in the literature for enhancing accuracy such as [7]–[9], [22] amongst others. Another example is [23], where the accuracy was improved by accounting for the edge points which can be problematic due to disparity alterations.

As already mentioned, the main problems that reduce accuracy in stereo matching are occlusion and illumination differences. In regards to occlusion, there are many works that concentrated on it like [18] and [21]. Also, [24] dealt with large occlusion stereo, and [25] used disparity alterations and the uniqueness properties of different pixels for occlusion handling. All of the previous methods relied on LRC while in this work, we handle occlusion through classification.

Concerning the illumination issues, these are addressed here through histogram information according to [26]. The work in [26] further summarizes some of the illumination related work in stereo matching.

In regards to speed, a key necessity for the stereo matching system is to be as fast as possible. This is greatly targeted through parallel hardware and software whether that is applied on the Multi Core CPU, the FPGA or the GPU as in [27]. Definitely, the algorithm itself must be fast and parallelizable to obtain a real time implementation. There are many works that solve the stereo problem on GPU devices such as [28], which discussed a local algorithm based on bitwise voting and applied it on the GPU. Moreover, [29] presented a GPU implementation of exponential ASW and exponential step message propagation. Also, [30] proposed an iterative refinement method for ASW based matching. In any case, the CUDA related literature presented approaches that are slower than the SAD algorithm [31], [32].

Concerning classification, it is constantly updated with applications in fields including Image Processing and Computer Vision. In this work, classification is first addressed using boosting in the form of Adaboost [33], [34] and RUSBoost [35]. Of course, there are many other boosting methods such as Linear Programming boosting [36], Gentle boosting [37], etc. . .

Besides boosting, Multi-layer perceptron feed forward neural networks (NN) are used as another method of classification. NN are a field on their own and are widely used for pattern recognition and classification. Definitely, there are many other classification techniques.

Since the work enhances stereo matching through classification, it is necessary to review relevant related works. The work in [38] differentiated between reliable and other pixels according to a set of measurements. This is different from our work since [38] did no classification or training. Also, [38] is constricted to the single method of matching used in [39] whereas the proposed work can be used with various techniques as a verification stage.

Moreover, there are other works that used classification and feature matching between pixel pairs to find the disparity which is not the approach adopted here. The work in [40] discussed integrating feature matching with disparity estimation and contour detection. Also, the works in [41]–[43] aimed at performing stereo matching through supervised learning, Support Vector Machines (SVM) and Hopfield neural networks. Additionally, the work in [44] used learning and feature selection for stereo matching, with [45] discussing the application of genetic algorithms to matching. Furthermore, the work in [46] implemented features based on Gabor filters and a neural model to estimate the disparity on the GPU. Also, [47] used biologically and psychologically inspired features in an ASW based algorithm. It is important to mention that all these related works did not evaluate their work using Middlebury [1] which renders comparing our work to theirs impractical.

III. INITIAL STEREO MATCHING

This section presents the first stages of the system. This work considers solving the stereo matching problem starting from any possible approach. But at first, we preprocess the images to reduce the illumination variances. Afterwards, matching is applied on the processed images.

A. INITIAL PROCESSING

The preprocessing method is based on the work in [26]. This basically involves finding the histograms of each stereo image. Then, a set of peaks of the two histograms are found. After that, the absolute differences between the corresponding peaks of each histogram are found. If the median difference is relatively high (≥ 2 in a 0-255 quantized image), histogram matching, or specification, is applied to match the image having the higher peak with the other. This leads to a slightly modified pair that are better matched than the original ones as the results already validated [26].

It is possible to use additional preprocessing techniques as in [26] or [48] but this is out of scope here.

It is worth mentioning that the proposed initial processing affects only the Teddy and Cones pairs. The Tsukuba and Venus pairs are found with close peaks (<2) which are conditions that stop this preprocessing.

Fig. 4 shows an example of the obtained SAD disparity maps with and without the proposed initial processing.

B. LOCAL STEREO MATCHING

Local methods usually use a direct measure of the pixels' difference. And as already noted, the main three measures are (SAD), (SSD), and (NCC). These can be defined using all color scales or just the gray scale which is adopted here

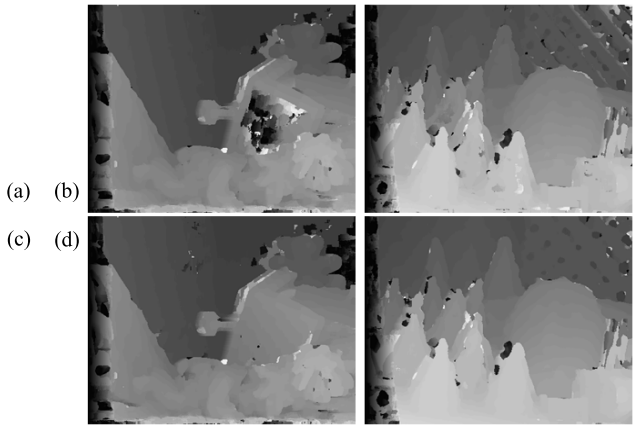


FIGURE 4. (a) and (b) Teddy and Cones SAD disparity maps, (c) and (d) Teddy and Cones SAD disparity maps after initial processing.

according to the following equations:

$$SAD_{x,y} = \sum_{i=x-Rad}^{x+Rad} \sum_{y-Rad}^{y+Rad} |L(i,j) - R(i,j)| \quad (1)$$

$$SSD_{x,y} = \sum_{i=x-Rad}^{x+Rad} \sum_{y-Rad}^{y+Rad} (L(i,j) - R(i,j))^2 \quad (2)$$

$$NCC_{x,y} = \frac{1}{n-1} \sum_{i=x-Rad}^{x+Rad} \sum_{y-Rad}^{y+Rad} \frac{(L(i,j)-\bar{L})(R(i,j)-\bar{R})^2}{\sigma_L \sigma_R} \quad (3)$$

L is the left gray scale image, R is the right image, n is the number of pixels in the selected square neighborhood, $I(i,j)$ is a pixel in the corresponding image, $\bar{I}$ is the mean of image I, σ is the standard deviation and Rad is the radius of the square neighborhood around the pixel.

As already mentioned, these are not the only ones and perhaps the most additional famous one is the (BT) measure in [5]. Moreover, the work in [4] lists various distances that can be used for image matching.

In this work, we initially consider using SAD and NCC for validating the proposed ideas, but any other method can be used. SSD is not tested since it achieves similar results as SAD. BT is not reported since it required additional calculations although it presented similar improvements.

C. OTHER STEREO MATCHING TECHNIQUES

In addition to the main local approaches, other new ones are tested. These validate that the proposed ideas are possible to implement on different matching approaches.

The first two are the works in [16] and [17]. These use BP and are readily available through OpenCV [49] GPU implementations which favored their implementation.

The other two are the methods of [9] and [22] since these are of high accuracy with [9] being state of the art. The output disparity maps of these are available, but their GPU code is not available, so their time results are not reported.

IV. CLASSIFICATION AND STEREO MATCHING

This section presents the main contributions of this work. First, the methods used for classification are briefly presented. Then, stereo matching is related to classification. Afterwards, the different features are described.

A. CLASSIFICATION METHODS

This work uses classification for stereo matching in an original manner. The classification methods include two Boosting schemes and neural networks. These are briefly described next.

1) BOOSTING

AdaboostM1 is probably the most popular binary boosting algorithm because it is straightforward and with acceptable accuracy. Since this work uses binary classification, AdaboostM1 was chosen.

The algorithm basically starts with a set of samples with equal weighting and a single weak learner. Then, the learner is trained while finding the error degree $e(i)$with (i) being the number of the learner. The error rate of each learner should be less than 50% since a greater rate halts the algorithm.

After any iteration, on the creation of a new learner, the weighting of each sample is modified based on being correct or not. The true patterns' weights are multiplied by

$$\beta(i) = \frac{e(i)}{1 - e(i)} \quad (4)$$

So, AdaboostM1 reduces the weights for the correct patterns with a post normalization step for all the patterns.

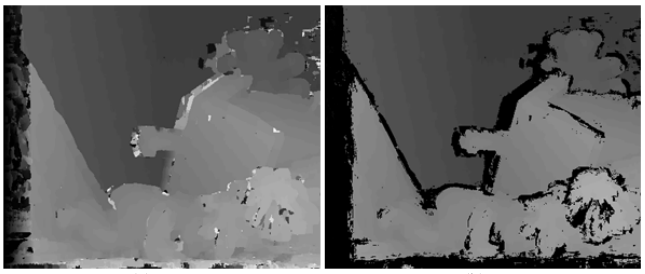
This modification for the weighting leads to obtain a new classifier through training. This is a process that is carried out for a number of steps which is the number of weak learners. Afterwards, all the classifiers are combined together to create a strong classifier that performs classification by weighted voting. The weighting of each weak classifier is set to:

$$W(learner\ i) = 0.5\ x\ \log(\frac{1}{\beta(i)}) \quad (5)$$

Besides AdaboostM1, another method called RUSBoost [35] is considered here. This is used because the involved classes were generally imbalanced with one class in more abundance than the other. RUSBoost mainly uses random under sampling (RUS) which arbitrarily removes data from the major class until an acceptable distribution is found. In the other points, it is rather similar to AdaboostM1.

RUSBoost can be modified according to a ratio that sets the percentage of the minority and the majority classes in the (RUS) process. In this work, this ratio varied according to the initial stereo matching method to ensure better results.

It is important to note that decision trees are the chosen weak learner for both boosting methods. Decision trees are the most common weak learners with each tree consisting of a condition that sets the output according to one of the features. Another important note is that 50 weak learners were used for each of the boosting methods.



(a) (b)

FIGURE 5. (a) Initial disparity map and (b) obtained disparity map.

2) NEURAL NETWORKS

Neural Networks are inspired from the human neural system and basically consist of a set of layers of neurons. Each layer performs a basic operation according to:

$$y = f\Big(\sum_i (w_i x_i + b)\Big) \tag{6}$$

x_i corresponds to the input of the neuron, w_i is a weighting factor, b is a constant bias parameter, and f is the activation function that is usually sigmoidal.

There are several ways for training neural networks and in this work, they are trained using the Levenberg-Marquardt backpropagation algorithm.

Concerning the activation function, it is chosen randomly for each layer of the NN amongst two possibilities. The first one is the hyperbolic tangent-sigmoid:

$$\tan sig(n) = \frac{2}{1 + e^{-2n}} - 1 \tag{7}$$

The second possible activation function is the log-sigmoid:

$$\log sig(n) = \frac{1}{1 + e^{-n}} \tag{8}$$

Moreover, the NN is set to consist of three hidden layers besides the input and output layers. The number of nodes in each layer is set to a random number between 3 and 6.

B. STEREO MATCHING USING CLASSIFICATION

After presenting the classification techniques, we need to relate them to stereo matching according to the proposed approach.

Computing the disparity is first performed using a selected technique amongst the selected matching methods. After that, it is possible to check whether the found disparity is correct or not, which is a binary classification problem. This makes the proposed method a verification stage. Other possibilities are discussed later.

Using this verification approach directly means the existence of another filling stage to complete the pixels since some are left with unknown disparities.

And although the features are not yet detailed, Fig. 5 shows the obtained map through NCC and AdaboostM1, where the black pixels are set to unknown. The map shows that the errors, especially due to occlusion, are reduced.

C. PROPOSED FEATURES

Perhaps the most important block of building a classifier is selecting the features that well suit the subject of study, since this determines the accuracy of the system. So, this section presents the suggested features.

The main initially available data are the stereo images and these are used to compute a preliminary disparity map using any of the matching methods handled here. Thus, the data becomes the image (I) or the left one (L) in the Middlebury images, with the initial corresponding disparity map (D).

In any case, there are many possible features, since it is possible to use the data in many ways. The proposed features were chosen to be simple and fast to compute. Also, the selected features try to account for local and global effects.

Moreover, the selected set of features varied according to the initial stereo matching algorithm to ensure better results. The different selected sets are shown in the results section.

The total features are eleven and they are described next with additional discussions later.

1) FEATURE 1: MEAN INTENSITY DIFFERENCE

The first feature is selected out of the image itself. The idea is to account for the discrepancies between the pixel (x, y) and its surroundings. This feature finds the difference of the intensity value of the pixel with all the pixels in a square neighborhood with radius *Rad*. This difference constitutes a small matrix that one can select different measures including the mean, median, standard deviation and others. The mean was selected for ease of computation. The mean being

$$f_1 = \frac{1}{n} \sum_{i=x-Rad}^{x+Rad} \sum_{j=y-Rad}^{y+Rad} |I(i,j) - I(x,y)| \tag{9}$$

2) FEATURE 2: MEAN EDGE DISPARITY VALUE

The second feature is depicted from the disparity map. This is not direct however, since it is based on finding the edge disparity map (E_D) according to the Al-Alaoui edge detection filter [50]. An example of the edge map for the NCC obtained map is shown in Fig. 6.

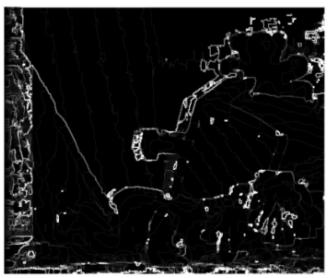


FIGURE 6. Example of the edge image of the disparity map.

Afterwards, for every pixel, the average edge disparity value is found. This is important since the edge map indicates the regions where the problems arise and the areas with the

highest concentration of errors especially due to occlusion.

$$f_2 = \frac{1}{n} \sum_{i=x-Rad}^{x+Rad} \sum_{j=y-Rad}^{y+Rad} E_D(i,j) \tag{10}$$

3) FEATURES 3 AND 4: NUMBER OF SIMILAR DISPARITIES AND MEAN DISPARITY DIFFERENCE

It is highly likely that if most surrounding pixels have a different disparity from the current one, the disparity is incorrect. So, the disparity of the current pixel is checked with its neighbors. At first, the number of pixels that have a dissimilar disparity in the current neighborhood is found. In the kernel, only pixels that are close in intensity (difference in intensity $\leq i_D$) to the current pixel are considered. The radius here is set to *Rad*2* instead of *Rad* since that led to better results.

The idea here is analogous to segmentation, since neighboring similar intensity value pixels usually belong to same segments,

$$f_3 = length(\sum_{i=x-Rad*2}^{x+Rad*2} \sum_{j=y-Rad*2}^{y+Rad*2} |D(i,j) - D(x,y)| > T_D)$$
$$Condition : |I(i,j) - I(x,y)| \leq i_D \tag{11}$$

This work assumes a disparity similar to another if the absolute difference is $\leq T_D$ according to:

$$T_D = \max(ceil(0.03 * disparity\ range), 1) \tag{12}$$

T_D equals 2 for Cones and Teddy and 1 for Venus and Tsukuba. Also, i_D is set to 15 and was found by trial and error.

Moreover, the mean difference between the disparity value and the surrounding disparities is computed which is the fourth feature.

$$f_4 = \frac{1}{n} \sum_{i=x-Rad*2}^{x+Rad*2} \sum_{j=y-Rad*2}^{y+Rad*2} |D(i,j) - D(x,y)| \tag{13}$$

4) FEATURE 5: DISPARITY VALUE

This feature is basically the disparity value $f_5 = D(x,y)$. This was used because it was noticeable that the lower disparities were more often true than the higher ones, especially in the SAD case. Also, this included no computational cost.

5) FEATURE 6: CANDIDATE ACCURATE DISPARITY

The main idea here is to check if the four disparity values at the corners of the neighborhood kernel are close to the current disparity. ''Close'' means that the absolute difference between each pair is $\leq T_D$. If all the differences are $\leq T_D$, the disparity is kept, while otherwise removed.

These disparity values are often true as shown in Fig. 7. This is termed Disparity Map A (D_A).

D_A is used in several of the features since it is easy to compute and has proven to enhance the accuracy. The first one of these features is feature six which is a logical value that indicates whether the value exists or not.

$$f_6 = 1\ if\ D_A(x,y)\ exists;\ 0\ \ otherwise \tag{14}$$

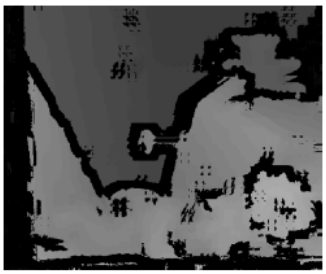


FIGURE 7. Disparity map A.

6) FEATURE 7: HISTOGRAM DISPARITY VALUE DIFFERENCE

Many of the features here combine the intensity and the disparity value and this feature is one of these. This is based on D_A in addition to the intensity of the pixels. The idea is to determine a histogram that indicates the average of each gray scale intensity value *(gi)*. This is straightforward because *gi* varies between 0 and 255 for 8 bit quantization and between 0 and 63 for 6 bits per pixel. So if one checks the intensity value of each known disparity pixel, the average disparity is found.

$$H_{DA}(gi) = \frac{1}{N = \#\text{ of pixels with} gi} \sum_{i=1}^{N} D_A(I = gi) \tag{15}$$

And after obtaining the histogram (H_{DA}), the intensity of the pixel to be classified is checked to find the equivalent disparity in the histogram. This disparity is then subtracted from the candidate disparity which is feature seven.

$$f_7 = \left| H_{DA}(I(x,y)) - D(x,y) \right| \tag{16}$$

7) FEATURES 8 AND 9: REGIONAL HISTOGRAM DISPARITY VALUE DIFFERENCES

These two features are rather similar and that is why they are grouped together. Like the previous feature, these rely on (D), (D_A) and (I). The ideas of the two features are the same, but each one is depicted from a different map, the first is from (D_A) and the second is from (D). So, feature eight is found by dividing (D_A) into four equal parts like a two by two array and assigning for each part a histogram similar to that in feature seven. Thus, there would be four histograms indicating the average disparity for each intensity value ($H1_{DA}$), ($H2_{DA}$), ($H3_{DA}$), and ($H4_{DA}$).

Now, and since any pixel would lie in either part, its feature is extracted from the histogram of the part it belongs to, as in feature seven. This means that the difference between the average and the candidate disparity is computed.

$$f_8 = \left| Hi_{DA}(I(x,y)) - D(x,y) \right| \tag{17}$$

The same applies for feature nine which is similar to feature eight but depicted from the regular disparity map (D) which leads to have ($H1_D$), ($H2_D$), ($H3_D$), and ($H4_D$).

$$f_9 = |Hi_D(I(x,y)) - D(x,y)| \tag{18}$$

8) FEATURES 10 AND 11: LOCAL HISTOGRAM DISPARITY VALUE DIFFERENCES

In a similar manner like features eight and nine, features ten and eleven are from (D_A) and (D) respectively. Also, these are obtained by dividing the maps into equal parts but instead of four, there are nine parts like a three by three array to lead to $(h1_{DA}), \ldots, (h9_{DA})$ and $(h1_D), \ldots, (h9_D)$. This means that there are nine histograms and each pixel is assigned to the histogram in the part it belongs to. Then, the absolute difference between the disparity and the corresponding histogram value is computed as in the previous features.

$$f_{10} = \left|hi_{DA}(I(x,y)) - D(x,y)\right| \tag{19}$$

$$f_{11} = \left|hi_{D}(I(x,y)) - D(x,y)\right| \tag{20}$$

D. COMPLETION STAGE

As already mentioned when discussing the method that relates classification and stereo matching, some disparities are left unknown, and so they must be found. This means that the completion stage decides the overall accuracy and speed of the system. It is worth emphasizing that any stereo matching algorithm that performs LRC requires a similar stage.

Before explaining the proposed method, it is important to note that there are many references as in [51] that initially rely on finding the maps for a set of points. In this work, the initially found points constitute a large percentage and were always more than 70% of all the pixels. Also, if the initial method was accurate, more than 90% of the pixels would be known. This means that the unknown pixels are rapidly found.

Definitely, there are several works that have a completion or filling stage and many of these rely on choosing nearby disparities and weighted median filtering as in [52]–[54]. In this work, a new heuristic approach is proposed to achieve acceptable accuracy with fast performance. This proposed stage can be considered as a minor contribution of this work. This is done by combining two techniques.

Before describing the techniques, it is important to note that the lower accuracy methods had very erroneous data before 60% of the disparity range. So, these pixels were set to unknown. This was not performed with [9] and [22] since they already handled occlusion using LRC.

The first technique is comprised of few steps and can be thought as a simple segmentation approach. At first, the gray image (I) is quantized into *10* values *(255/25.5)* or more, or the RGB image into *10* values in each scale *(255/25.5)* to yield new image *(Iq)*. Then, each connected empty disparity region is treated as a single region. An empty region is a region where the disparities are currently unknown since they were found erroneous.

The region dimensions are limited by *min((# of rows in Image)/2, (# of columns in Image)/2)*. So, if a dimension bypasses the limit, the region is divided into several parts.

Then, we check each empty region. In the region, we consider the *Iq* values with unknown disparities and assign them with the median disparity of similar known *Iq* values in the vicinity of the region. The vicinity of the region is every known disparity pixel that neighbors the region as in Fig. 8. The obtained map is termed (D_1).

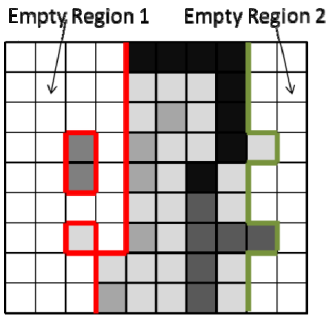


FIGURE 8. Example of empty regions where each (gray level) pixel with a colored border being in the vicinity of the empty (white) region.

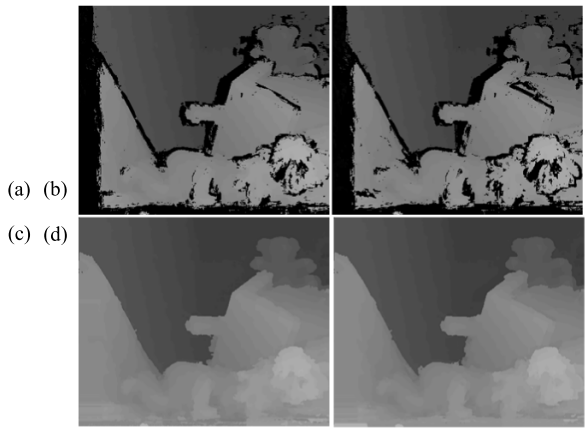


FIGURE 9. (a) Obtained disparity map through AdaboostM1, (b) Obtained disparity map through RUSBoost, (c) and (d) obtained maps after the completion stage.

The second technique is to use the basic approach in [52]–[54], termed *(Cb)*, where each unknown disparity is filled with the nearest spatially known disparity along the same row to yield the map (D_2).

These two techniques are then combined meaning that if the two disparity values have an absolute difference $\leq T_D$ as in feature 3, they are averaged, while otherwise the value is set to unknown again.

This validation approach leaves a set of unknown pixels which are now filled by the first process unless no similar *Iq* known value exists. These unknown *Iq* values are filled with *(Cb)* to yield the map (D_3).

An idea that enhanced the results is to special treat the pixels that lie before 60% of the disparity range. These disparities are set to the maximum value between that of (D_3) and (D_2) leading to the final map.

Fig. 9 shows an example of the obtained maps after the filling stage for the Teddy NCC obtained map through both AdaboostM1 and RUSBoost.

It is important to mention that after the completion stage, median filtering is performed since it reduces the noise and improves the results.

A very vital notion here is that the completion stage could not have been any useful if the classification stage did not well identify the erroneous pixels. Additional ideas are mentioned in the discussions in section VI.

V. RESULTS

In this section, the main results are presented. These are shown for the Middlebury evaluation set [1]. Also and since we are using a classification based approach, it is important to create a training dataset which is also based on images selected from [1]. The images used for training included Bull, Aloe, Art and Books except for [9] and [22] since their computed disparity maps are not readily available. The true disparity maps of these images are available, so the correct classes can be easily checked.

In particular, 25% of the pixels of each image were randomly chosen for training which makes a total of 167350 pixels. 25% means that there are 4 parts. This was performed for 10 times in each case which means a total of 40 runs were made. The results tabulated here are the average of the different runs with the figures being of a random case. Definitely, each method had a separate training stage since each one had different types of error.

The main errors in the training stages varied between 3 to 13% for all the classification techniques. The RUSBoost method generally had the higher error rate although it was better in identifying the minority class or the erroneous pixels. A notable idea is that the minority ratio for RUSBoost was set to *"1"* in all methods except in [9] and [22] where it was set to *"0.55"* since the default ratio led to worse results.

In regards to the error calculation and the comparison of the disparity with the true one, two measures are used. The first measure is that used in [1]. This basically finds the percentage of pixels whose disparity differs from the true disparity by a certain difference threshold according to equation (21):

$$\%error = \frac{\#pixels\ with(|disp - truedisp| > thresh)}{\#all\ pixels} * 100 \quad (21)$$

The threshold usually varies between 0.5 and 2 according to [1] and in this work we set it equal to T_D.

The other error measure is the mean absolute error:

$$MAE = \frac{1}{\#all\ pixels} \sum_{all\ pixels} |disp - truedisp| \quad (22)$$

This is used because it provides an indication of the average error per pixel and is not governed by a threshold value.

Fig. 10 shows the NCC obtained disparity maps through the proposed approach with AdaboostM1 before and after the completion stage. Moreover, all the figures that are shown afterwards were obtained through AdaboostM1. It is important to mention that all the proposed features excluding 9 and 11 were considered for classification using this approach.

Fig. 11 shows the SAD obtained disparity maps before and after using the proposed enhancement. Also, and like NCC, all the features were chosen excluding 9 and 11.

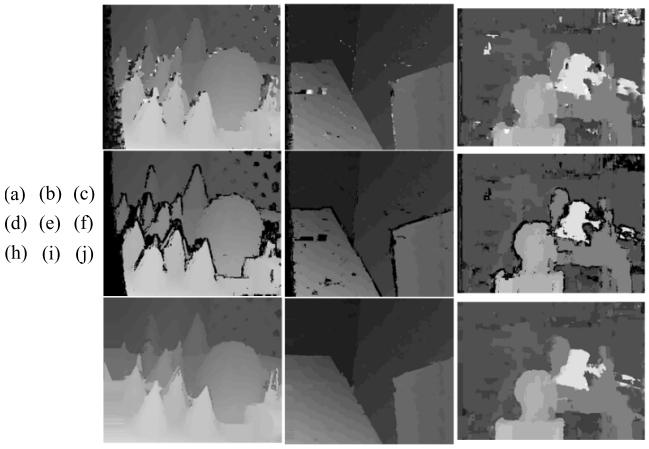


FIGURE 10. (a) , (b) and (c) Original NCC disparity maps, (d), (e) and (f) disparity maps after removing erroneously classified pixels, (h), (i) and (j) output disparity maps.

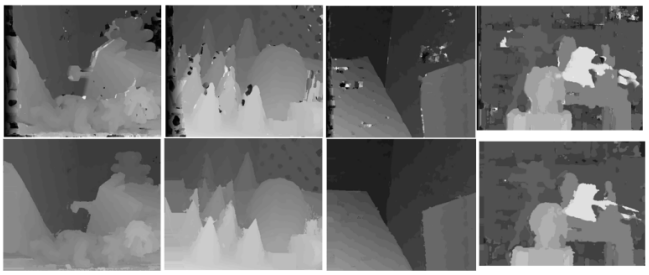


FIGURE 11. Disparity maps using SAD (top row) and through SAD + proposed approach (bottom row).

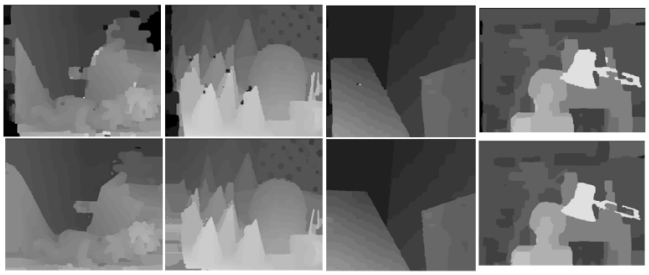


FIGURE 12. Disparity maps using [16] (top row) and through [16] + proposed approach (bottom row).

Fig. 12 shows the obtained disparity maps through [16] and after enhancing the results with the proposed approach. For this method, the selected features were the same as NCC.

Fig. 13 shows the results obtained by performing the method of [17] before and after the proposed approach. The same set of features as before was chosen with this approach.

Also, Fig. 14 shows the disparity maps obtained through [9] before and after using the proposed enhancement. The selected features were features 2 to 9.

Fig. 15 shows the disparity maps obtained through [22] before and after performing the proposed implementation. For this method, the set of selected features was composed of features 1, 3, 5, 6, 7, 9 and 11.

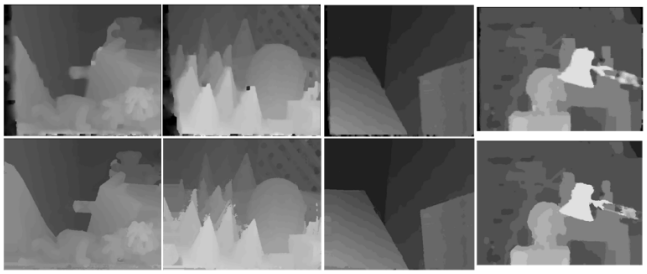


FIGURE 13. Disparity maps using [17] (top row) and through [17] + proposed method(bottom row).

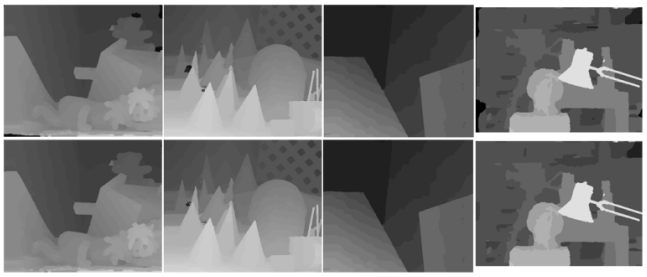


FIGURE 14. Disparity maps using [9] (top row) and through [9] + proposed approach (bottom row).

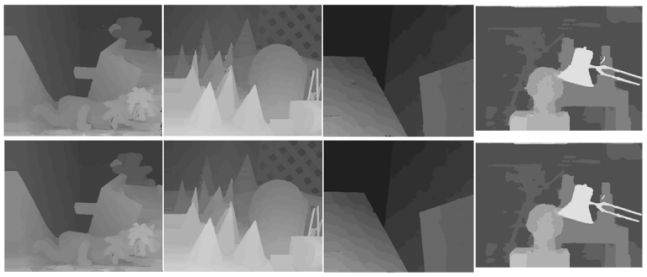


FIGURE 15. Disparity maps using [22] (top row) and through [22] + proposed approach (bottom row).

Table 1 shows the results evaluated according to equation (20) with a threshold value equal to 2 or 1 according to the disparity range of the image. Table 2 shows the MAE results. The tables show the original error and the found error after applying the proposed approach for the different used methods. The initial processing results are presented when possible. It is worth remarking that preprocessing was not performed to the methods of [9] and [22] and is not needed for NCC that is illumination invariant.

Besides, this work argues that it is possible to perform occlusion handling through classification which should be validated. This is achieved by testing the different methods with LRC in comparison with classification. The tested methods are the NCC, SAD, method of [16] and method of [17]. The works of [9] and [22] are not tested since their code and right disparity maps are not available. For each method, we present the results achieved after performing filling through both our proposed completion stage (*Cp*) and the basic regular method (*Cb*).

In order to ensure a fair comparison, all the images underwent initial processing. In addition, the disparities prior to 60% of the disparity range were set to unknown in all the cases since it generally improved the results. Table 3 shows the comparison between the methods. In Table 3, {1} means the proposed approach using AdaboostM1 with (*Cp*) and {2} means the proposed approach using AdaboostM1 with (*Cb*) and median filtering. Also, {3} means LRC followed by (*Cp)* and {4} means LRC followed by (*Cb*) and median filtering.

The results validate that classification performs better than LRC in most cases, albeit not all, for the tested methods. LRC performs better on the Teddy and Cones images using the methods of [16] and [17]. Also, the results prove that *(Cp)* performs better or at least similar to *(Cb)*.

It is important to note that we validated the approach using other images. For example, we report the results of using SAD with AdaboostM1 on images that include Map, Bull and Barn from [1]. To obtain the maps, the classifier is trained using the Middlebury evaluation images and maps. So, Table 4 presents the error results for these images with and without the proposed work.

VI. DISCUSSION

This work addresses stereo matching through classification. This requires further analysis and discussion of the different ideas especially regarding features, speed and accuracy.

A. DISCUSSION ABOUT FEATURES

Features were selected based on speed and accuracy. Some features seemed redundant but were used since they were easy to compute. Some candidate features improved the accuracy but were dismissed due to extra computations. Various tests were performed with and without the used features. The tests included random combinations of different features and testing the possible set excluding each contending feature. These tests led to the described set of features for each initial method.

Definitely, each feature had a different effect on the accuracy of the proposed work. In particular, features 2, 3, and 6 seemed the most influential, with other features having varying effects according to the initial method.

Also, there are similarities between the features and this is mainly because they are depicted from the intensity and disparity data. Besides, some features are indeed alike as in features 8 and 9 and features 10 and 11. In such cases, it is possible to use one of each, but they are mentioned here for possible future use.

It is important to note that in order to ensure faster results; each of the histograms used 64 bins instead of 256 bins. This did not diminish the accuracy and helped obtain higher speeds. Furthermore, few of the histograms had some empty bins, so, these were filled with the nearest known values.

In addition, features 1 and 2 can be further improved by dividing each feature by the mean of the

TABLE 1. Results on the middlebury test images measured by percentage error threshold difference.

Error / Method \ Image	*Teddy*			*Cones*			*Venus*			*Tsukuba*		
	Original	Initial	Proposed	Original	Initial	Proposed	Original	Initial	Proposed	Original	Initial	Proposed
NCC+AdaBoostM1	21.3	21.3	*11.6*	19.7	19.7	*12.5*	6.9	6.9	*2.7*	11.8	11.8	*5.9*
NCC+RUSBoost	21.3	21.3	*10.8*	19.7	19.7	*12.7*	6.9	6.9	*2.3*	11.8	11.8	*5.5*
NCC+NN	21.3	21.3	*11.3*	19.7	19.7	*11.8*	6.9	6.9	*2.6*	11.8	11.8	*4.9*
SAD+AdaBoostM1	26.8	23.2	*11.5*	20.2	19.5	*12.5*	8.4	8.4	*2.6*	10.1	10.1	*5.2*
SAD+RUSBoost	26.8	23.2	*10.9*	20.2	19.5	*12.2*	8.4	8.4	*2.4*	10.1	10.1	*5.4*
SAD+NN	26.8	23.2	*11.9*	20.2	19.5	*11.9*	8.4	8.4	*2.7*	10.1	10.1	*4.9*
[16]+AdaBoostM1	22.6	20.9	*11.9*	19.4	14.3	*11.7*	2.8	2.8	*1.2*	5	5	*2.9*
[16]+RUSBoost	22.6	20.9	*11.3*	19.4	14.3	*11.8*	2.8	2.8	*0.9*	5	5	*3.8*
[16]+NN	22.6	20.9	*11.6*	19.4	14.3	*11.2*	2.8	2.8	*1.5*	5	5	*2.7*
[17]+AdaBoostM1	19.3	18.8	*11.5*	16.8	16.0	*11.9*	4.7	4.7	*1.9*	5.2	5.2	*4.5*
[17]+ RUSBoost	19.3	18.8	*11.2*	16.8	16.0	*11.2*	4.7	4.7	*1.9*	5.2	5.2	*4.3*
[17]+ NN	19.3	18.8	*11.8*	16.8	16.0	*10.9*	4.7	4.7	*1.9*	5.2	5.2	*3.9*
[9]+AdaBoostM1	7.0	NA	*6.6*	6.1	NA	*6.1*	0.5	NA	*0.4*	2.3	NA	*1.9*
[9]+RUSBoost	7.0	NA	*6.5*	6.1	NA	*6.0*	0.5	NA	*0.4*	2.3	NA	*1.9*
[9]+NN	7.0	NA	*6.6*	6.1	NA	*6.0*	0.5	NA	*0.4*	2.3	NA	*2.0*
[22]+AdaBoostM1	7.9	NA	*7.6*	7.2	NA	*6.9*	0.7	NA	*0.5*	1.8	NA	*1.6*
[22]+RUSBoost	7.9	NA	*7.6*	7.2	NA	*7.0*	0.7	NA	*0.5*	1.8	NA	*1.6*
[22]+NN	7.9	NA	*7.6*	7.2	NA	*6.9*	0.7	NA	*0.5*	1.8	NA	*1.5*

corresponding image. This reduces the color variances amongst varying images.

Also, there could be some ambiguities as in accounting for a true disparity which is usually in sub-pixel accuracy. For this, T_D was used and if the disparity error was $\leq T_D$, it was accepted as true. The same idea was used for the confident disparity map (D_A), so if the difference was $\leq T_D$, the disparities were considered similar.

Moreover, there are many possibilities for the features. The first one is including the SAD or the NCC distances. It is further possible to choose different sized neighborhoods. Also, one can use the median instead of the mean for several of the features. Another simple choice is to divide the map into different sized parts and so on. Also, one can generate other confident maps as in (D_B) and (D_C)by accounting for various combinations from the neighboring pixels. Furthermore, different features can be combined through normalization and multiplication which could improve accuracy.

Additionally, it is possible to use the edge filtered map of the stereo image or use the Canny [55] Edge map for the stereo images and maps, but this is generally time demanding.

Besides, Local Binary Patterns (LBP) and Histogram Oriented Gradients (HOG) were tested for both the image and the map, but they provided little improvement. This is probably due to redundancy and similarity to other features.

It is important to stress that including more features would require additional processing and slow the system.

Obviously, there are numerous possibilities and countless features to try, which can be carried in future work.

B. DISCUSSION ABOUT COMPLETION STAGE

The filling stage is a key part of the presented system and in finalizing the disparity map. This section presents an additional usage of the suggested approach in addition to possible enhancements.

This can be used to combine two similar performing algorithms as in combining the methods of [9] and [22]. Another example would be to combine the proposed RUSboost and Adaboost obtained maps.

The combination starts by validating each pixel in the two maps where dissimilar pixels are set to unknown. Then, the unknown pixels are filled according to the proposed completion stage. However, and to further improve the results it is better to special treat the pixels whose column values are less than the disparity range. So, in the case of combining [9] and [22], an additional step would be to select the disparities of [9]. Fig. 16 shows the resulting images for the Middlebury evaluation set.

Table 5 shows the error results of the obtained maps. These are better than both of the methods.

TABLE 2. Results on the middlebury test images measured by MAE.

MAE / Method \ Image	*Teddy*			*Cones*			*Venus*			*Tsukuba*		
	Original	Initial	Proposed	Original	Initial	Proposed	Original	Initial	Proposed	Original	Initial	Proposed
NCC+AdaBoostM1	3.36	3.36	*1.13*	3.18	3.18	*1.22*	0.74	0.74	*0.38*	0.72	0.72	*0.39*
NCC+RUSBoost	3.36	3.36	*1.06*	3.18	3.18	*1.19*	0.74	0.74	*0.36*	0.72	0.72	*0.41*
NCC+NN	3.36	3.36	*1.05*	3.18	3.18	*1.17*	0.74	0.74	*0.37*	0.72	0.72	*0.38*
SAD+AdaBoostM1	4.05	3.48	*1.09*	3.18	3.07	*1.19*	0.88	0.88	*0.36*	0.66	0.66	*0.41*
SAD+RUSBoost	4.05	3.48	*1.02*	3.18	3.07	*1.18*	0.88	0.88	*0.35*	0.66	0.66	*0.42*
SAD+NN	4.05	3.48	*1.14*	3.18	3.07	*1.13*	0.88	0.88	*0.35*	0.66	0.66	*0.42*
[16]+AdaBoostM1	2.87	2.59	*0.99*	2.92	2.28	*1.08*	0.57	0.57	*0.3*	0.44	0.44	*0.37*
[16]+RUSBoost	2.87	2.59	*0.94*	2.92	2.28	*1.14*	0.57	0.57	*0.32*	0.44	0.44	*0.41*
[16]+NN	2.87	2.59	*0.97*	2.92	2.28	*1.09*	0.57	0.57	*0.32*	0.44	0.44	*0.36*
[17]+AdaBoostM1	3.3	2.65	*1.14*	2.5	2.37	*1.05*	0.49	0.49	*0.33*	0.37	0.37	*0.32*
[17]+RUSBoost	3.3	2.65	*1.06*	2.5	2.37	*1.04*	0.49	0.49	*0.33*	0.37	0.37	*0.33*
[17]+NN	3.3	2.65	*1.15*	2.5	2.37	*0.99*	0.49	0.49	*0.33*	0.37	0.37	*0.29*
[9]+AdaBoostM1	0.92	NA	*0.82*	0.73	NA	*0.71*	0.29	NA	*0.28*	0.22	NA	*0.21*
[9]+RUSBoost	0.92	NA	*0.79*	0.73	NA	*0.7*	0.29	NA	*0.28*	0.22	NA	*0.21*
[9]+NN	0.92	NA	*0.79*	0.73	NA	*0.71*	0.29	NA	*0.28*	0.22	NA	*0.22*
[22]+AdaBoostM1	0.84	NA	*0.78*	0.78	NA	*0.77*	0.31	NA	*0.3*	0.2	NA	*0.2*
[22]+RUSBoost	0.84	NA	*0.77*	0.78	NA	*0.77*	0.31	NA	*0.31*	0.2	NA	*0.2*
[22]+NN	0.84	NA	*0.79*	0.78	NA	*0.76*	0.31	NA	*0.3*	0.2	NA	*0.19*

TABLE 3. Occlusion handling results on the middlebury test images measured by percentage error threshold difference.

Error / Image \ Method	*NCC {1}*	*NCC {2}*	*NCC {3}*	*NCC {4}*	*SAD {1}*	*SAD {2}*	*SAD {3}*	*SAD {4}*	*[16] {1}*	*[16] {2}*	*[16] {3}*	*[16] {4}*	*[17] {1}*	*[17] {2}*	*[17] {3}*	*[17] {4}*
Teddy	11.6	11.9	12.3	12.7	11.5	13.9	14	16	11.9	13.6	11.7	13.5	11.5	13.4	11.3	11.7
Cones	12.5	12.6	14.5	14.5	12.5	12.6	12.6	13	11.7	12.4	10.1	11.9	11.9	13.2	11.5	12.4
Venus	2.7	2.9	4.3	4.7	2.6	3.1	5.3	5.6	1.2	1.3	1.4	1.5	1.9	2.6	3.4	3.4
Tsukuba	5.9	6.4	9.3	9.7	5.2	5.8	8.0	8.2	2.9	3.5	3.1	3.9	4.5	5.2	4.1	5.5

TABLE 4. Error results of other images from [1].

Image	*Map*	*Bull*	*Barn 1*	*Barn 2*
SAD % Error	7.1	5.2	6.9	9.3
SAD MAE	1.48	0.47	0.65	0.89
SAD+AdaBoostM1 % Error	2.1	1.6	2.8	2.9
SAD+AdaBoostM1 MAE	0.66	0.28	0.35	0.39

It is worth noting that it is possible to use weighted median filtering to further enhance the accuracy of the completion stage and stereo matching. This was not considered here due to the additional computations.

C. DISCUSSION ABOUT RESULTS

The best obtained results varied amongst the classification approaches according to the tested image and the chosen algorithm. Also, high enhancement was not always achieved as in the Tsukuba and Venus images which were less enhanced using [9] and [22].

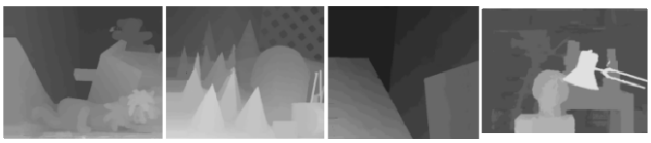


FIGURE 16. Obtained disparity maps using [9] + [22] with the filling stage.

Moreover, it is notable that the results were more improved in the lower accuracy methods as opposed to the higher ones.

Besides, the results in Table 3 indicate that classification does not surpass LRC unless used with a suitable approach, like NCC and SAD in this work.

This varying enhancement amongst the methods is due to the error being more systematic in the lower methods

TABLE 5. Error results on the middlebury set of [9]+[22] with the filling stage.

Image	*Teddy*	*Cones*	*Venus*	*Tsukuba*
% Error	6.4	5.8	0.4	1.5
MAE	0.79	0.67	0.27	0.17

with the existence of bigger room for improvement. Another reason is that the features concentrated on the lower methods and additional features are required to enhance the others. Besides, a vital idea is that the accurate methods already handled occlusion using LRC which reduced the main source of error.

Also, and as an argument in favor of this work, many of the best results in literature are obtained through time demanding approaches performed on the high resolution colored images. This is different from using the low resolution gray scale ones as in this work. A typical example of this is [54] which used segmentation and matching to achieve high accuracy at the expense of extra computations.

Moreover, the found results achieve better accuracy and speeds than those in [39] and [40]. Besides, the enhanced SAD shows a clear improvement over using [16] and [17].

D. TIME ANALYSIS

Stereo matching is analyzed according to accuracy and speed as already noted. The combination of a fast and relatively accurate system is a property that is achieved here. Thus, it is important to analyze the speed of the system. The analysis is explained according to a parallel implementation on a CUDA device. In particular, we used an Nvidia Geforce GTX 460 GPU and obtained the kernel execution time results using the Visual Profiler by Nvidia.

We used the texture memory of the CUDA device to store the image and its map. Also, we used the shared memory in all the features, since this is essential for reducing the accesses to global or texture memory to obtain real time speeds.

In regards to the initial processing, this is relatively fast since the histograms are easy to compute using atomic operations in CUDA [57].

Concerning the local methods such as SAD and SSD, these can be efficiently computed on the GPU [31]. For a radius ($Rad = 5$), the proposed algorithm in [31] consumes about 4 ms on a Geforce GTX 460 for the Teddy image. Other methods such as BT and NCC require additional time but are generally acceptable according to the size of the neighborhood kernel.

Regarding classification, training is time consuming since it tries to determine the parameters of the classification method, with each approach using a different training operation. But, this is only performed once and prior to the implementation of the stereo matching system. Also, the training does not require a GPU implementation.

Concerning the testing or implementation stage of classification, this is required for every pixel which is computationally acceptable as explained next.

The vital idea here is that the proposed classification is independent of the disparity range since it is a post processing stage. This means that classification consumes the same time on same sized images.

Concerning the features, and although there are eleven, they require little time. Features 1 and 2 are direct and can be computed using separable convolution [58]. The work in [58] reported speeds of 1.4 (Giga pixels per second) GPPS for large arrays. Also, Feature 5 is already available.

For feature 6, the map (D_A) requires checking four or five values per pixel which is straightforward. Concerning features 7, 8, 9, 10 and 11, the histograms can be computed with a speed of 5.5 GPPS [57] for histograms of 256 bins and 10 GPPS for histograms of 64 bins for large arrays.

The main time consuming features are 3 and 4 since they are performed on bigger neighborhoods as in a (21×21) kernel as opposed to an (11×11) neighborhood.

The required time can be checked by finding the number of operations done per pixel. So, if we study the exhaustive algorithm case of the SAD computation, each pixel would require $dr \times (Rad \times 2 + 1)^2$ set of operations. dr is the disparity range and Rad is the Radius of the kernel. The operations include additions and conditions. This is compared with the hardest to compute features, features 3 and 4. These require about $2 \times ((Rad \times 2) \times 2 + 1)^2$ different operations where the $2\times$ is for the map and the image, and the first $\times 2$ is for the larger radius. So, if ($Rad = 5$ & $dr = 60$), the SAD algorithm would require almost eight times the computation of features 3 and 4 which are found together. In fact, the computation of these two features is comparable to a SAD disparity map computation performed at a larger radius $Rad \times 2$ but at a small disparity range $dr \leq 3$.

After finding the features, classification is performed. This is fast especially according to boosting which requires checking a set of conditions according to the tree classifier and determining the class of each pixel. It is worth mentioning that the total time required for the features and AdaboostM1 is about 40% of the SAD map computational time in the Teddy and Cones images for an (11×11) kernel.

After the classification, the completion of the unknown pixels mainly requires checking the neighboring pixels and the possible known disparities. The time consumed here is about 60% of the SAD map required time since this is performed only for a set of pixels that are grouped together. The completion stage mainly relies on labeling connected components which is fast on both the CPU and the GPU [57].

After the filling stage, median filtering is used. This is also fast since it is performed on a small neighborhood ($Rad = 1$).

It is important to note that the usage of the basic completion stage (Cb) directly means less total time, albeit at the expense of reduced accuracy. (Cb) consumes about 0.2 ms.

Table 6 shows the portion of each computational part in the SAD and AdaboostM1 approach for the Teddy image.

It is important to remark that the completion stage requires the most time in the SAD case since it was the most erroneous, so using other matching methods should be faster.

TABLE 6. Portion in overall time of each component for the teddy MAP computation using SAD + AdaboostM1.

Part	Time (Milliseconds)	Portion
Initial Processing	1.3	13.5
SAD Stereo	3.9	40.6
Feature 1	0.1	1
Feature 2	0.2	2.1
Features 3 and 4	0.4	4.2
Feature 5	0	0
Feature 6	0.1	1
Feature 7, 8, 9, 10, 11	0.5	5.2
Classification (AdaboostM1)	0.3	3.1
Completion (Cp)	2.4	25
Median Filtering (MF)	0.4	4.2
Total	9.6	100%

TABLE 7. Execution time (ms) comparison with other stereo matching methods.

Method	Teddy	Cones	Venus	Tsukuba	SAD Ratio
SAD (11x11)	3.9	3.9	2.1	0.9	1
SAD(11x11) +LRC+Cb	9.7	9.6	5.8	3.4	2.6
SAD(11x11) +LRC+Cp	13.9	12.5	7.8	4.9	3.6
Proposed (SAD+Boost)	9.6	9.6	6.7	5.2	2.9
[16]	217	217	104	37	53
[17]	79	79	39	31	21

Table 7 shows the execution times obtained by using SAD with the proposed stages in comparison with [16] and [17] without the proposed enhancement. In all the cases, initial processing and median filtering are included.

It is necessary to note that the literature usually compares between algorithms on different GPUs as in [28]–[30], which can be hard to follow. So, we opt to account for the time of the SAD approach on an (11×11) window as a benchmark for comparison. The reported time for this SAD does not include preprocessing and median filtering. This is done since the GPU algorithms in literature require more time than the SAD approach even with LRC. Through this, it would become easier to compare related stereo matching research. So, Table 7 shows the ratio of the execution time of each method in comparison with the SAD (11×11) approach.

It is worth mentioning that the approaches in [28]–[30] consume more than 40 ms on the Teddy image. So, the proposed approach consumes little time in comparison with such works. But, it is not totally fair to compare with these approaches since they are performed on bigger kernels and they are more accurate due to using an ASW based approach.

Also, such works use LRC, which is usually computationally demanding except in some works that handle occlusion using a winner takes all approach where the disparities are selected from a cost volume.

E. ADVANTAGES AND LIMITATIONS

The main advantage of using classification for stereo matching in the unique suggested manner is the speed that the system improves accuracy. This is achieved without the need for LRC and regardless of the number of possible disparities. Still there are arguable drawbacks.

First, the enhancement for lower accuracy images is more as already discussed with the possibility of having no improvement if erroneous pixels were not correctly found.

Besides, the data and storage requirements of the system might seem high due to having eleven features per pixel. These features need to be computed for every pixel which could require a relatively high amount of data to store if not well treated. In any case, the memory storage is acceptable since the worst case scenario does not exceed *"13"* times the image size, but this is discouraged since it would require additional global memory accesses if used with CUDA.

Besides, it is clear that the proposed approach is totally dependent on the initial matching method in addition to the classification technique and the features used.

Furthermore, the system is totally reliant on the quality of the training data and its relation to the testing data. Thus, if the training data is too dependent on a certain scene it is expected to perform well on similar scenes and less on different ones.

Also, the completion stage needs to be accurate and the overall accuracy of the system directly depends on this stage, so it is not just a matter of classification.

Still, some of the limitations may be claimed as advantages The approach can be used with any initial matching method which makes it generic. Also, the system delivers according to what you give it, so the dependence on training is convenient when stereo matching is to be performed in a known environment Another very significant advantage is that the ideas presented here constitute a new technique for handling occlusion without computing the opposite map.

Moreover, there are several gains from the presented approach. The first one is that the system's dependence on features means it can be improved with others. Also, one can use segmentation since similar disparities are usually obtained in same segments. In addition, using color information should enhance the accuracy, although more computations would be required. Also, one can use other classifiers such as other boosting methods, SVM, Bagging, etc. . .

Furthermore, there are other possibilities for the classification problem since one can check the correctness of a set of disparities as opposed to checking only one as done here. This can be useful since it is highly likely that the correct disparity would be in the selected set. Of course, the more there are candidate disparities, the more computations are required. All these ideas prove that there are many prospects for future improvements.

VII. CONCLUSION

This work targeted stereo matching in an original approach that relies on classification to reduce errors and handle occlusion.

The images were first processed using histogram information for reducing illumination variances. Afterwards, different standard and new methods of stereo matching were considered to obtain the initial disparity map. The map and the image were then used as the main data for obtaining several features that are used in a classification scheme. This scheme succeeds in remedying different errors of the initially used algorithm. The work further discussed the different possible prospects noting that the timing added to the implemented system is minimal and justifies the increase in performance.

The main contribution is that the proposed system is generic since any algorithm can be used for the initial matching. This also renders the approach as a novel verification stage that can be used for solving the occlusion problem. A rather minor contribution is the suggested completion stage which can be used in any stereo matching method after occlusion handling. There are several merits of this work. First, handling occlusion in the proposed manner saves the computation of the opposite map if not desirable. Second, the features used are easy to compute and can be improved. Third, the classification is performed per pixel in a straightforward manner. Fourth, most of the parts are parallelizable and fast yielding real time performance through parallel architectures like CUDA. Fifth, the accuracy is acceptable since low resolution gray scale images were used. And finally, all these ideas provide room for future work.

REFERENCES

[1] D. Scharstein and R. Szeliski, "High-accuracy stereo depth maps using structured light," in *Proc. IEEE Comput. Soc. Conf. CVPR*, vol. 1. Jun. 2003, pp. I-195–I-202.
[2] (2014, Feb. 14). *'Stereo_camera' in Wikipedia: The Free Encyclopedia* [Online]. Available: http://en.wikipedia.org/wiki/Stereo_camera
[3] Z. Lei, G. Xianwei, Z. Cheng, and D. Xiuze, "Research on the technology of extracting 3D face feature points on basis of binocular vision," in *Proc. 2nd Int. CISP*, Oct. 2009, pp. 1–4.
[4] B. Cyganek and J. P. Siebert, *An Introduction to 3D Computer Vision Techniques and Algorithms*. New York, NY, USA: Wiley, 2011.
[5] S. Birchfield and C. Tomasi, "A pixel dissimilarity measure that is insensitive to image sampling," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 20, no. 4, pp. 401–406, Apr. 1998.
[6] T. Kanade and M. Okutomi, "A stereo matching algorithm with an adaptive window: Theory and experiment," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 16, no. 9, pp. 920–932, Sep. 1994.
[7] K.-J. Yoon and I.-S. Kweon, "Locally adaptive support-weight approach for visual correspondence search," in *Proc. IEEE Comput. Soc. Conf. CVPR*, vol. 2. Jun. 2005, pp. 924–931.
[8] M. Gong, R. Yang, L. Wang, and M. Gong, "A performance study on different cost aggregation approaches used in real time stereo matching," *Int. J. Comput. Vis.*, vol.75, no. 2, pp. 283–296, 2007.
[9] D. Min, J. Lu, and M. N. Do, "Joint histogram-based cost aggregation for stereo matching," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 35, no. 10, pp. 2539–2545, Oct. 2013.
[10] V. Vineet and P. J. Narayanan, "CUDA cuts: Fast graph cuts on the GPU," in *Proc. IEEE Comput. Soc. Conf. CVPRW* Jun. 2008, pp. 1–8.
[11] M.-H. Ju and H.-B. Kang, "A new method for stereo matching using pixel cooperative optimization," in *Proc. 16th IEEE ICIP*, Nov. 2009, pp. 2105–2108.
[12] Z.-F. Wang and Z.-G. Zheng, "A region based stereo matching algorithm using cooperative optimization," in *Proc. IEEE Conf. CVPR*, Jun. 2008, pp. 1–8.
[13] J. Y. Goulermas, P. Liatsis, and T. Fernando, "A constrained nonlinear energy minimization framework for the regularization of the stereo correspondence problem," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 15, no. 4, pp. 550–565, Apr. 2005.
[14] E. Park and K. Wohn, "Stereo and motion correspondences using nonlinear optimization method," *Comput. Vis. Image Understand.*, vol. 101, no. 3, pp. 194–203, 2006.
[15] W. Kim, J. Park, and K. Lee, "Stereo matching using population-based MCMC," *Int. J. Comput. Vis.*, vol. 83, no. 2, pp. 195–209, 2009.
[16] P. F. Felzenszwalb and D. P. Huttenlocher, "Efficient belief propagation for early vision," *Int. J. Comput. Vis.*, vol. 70, no. 1, pp. 41–54, 2006.
[17] Q. Yang, L. Wang, and N. Ahuja, "A constant-space belief propagation algorithm for stereo matching," in *Proc. IEEE Conf. CVPR*, Jun. 2010, pp. 1458–1465.
[18] J. Sun, Y. Li, S. B. Kang, and H.-Y. Shum, "Symmetric stereo matching for occlusion handling," in *Proc. IEEE Comput. Soc. Conf. CVPR*, vol. 2. Jun. 2005, pp. 399–406.
[19] J. Zhu, L. Wang, R. Yang, and J. Davis, "Fusion of time-of-flight depth and stereo for high accuracy depth maps," in *Proc. IEEE Conf. CVPR*, Jun. 2008, pp. 1–8.
[20] Q. Yang, L. Wang, R. Yang, H. Stewenius, and D. Nister, "Stereo matching with color-weighted correlation, hierarchical belief propagation, and occlusion handling," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 31, no. 3, pp. 492–504, Mar. 2009.
[21] I. Ernst and H. Hirschmüller, "Mutual information based semi-global stereo matching on the GPU," in *Proc. 4th Int. Symp. Adv. Vis. Comput.*, 2008, pp. 228–239.
[22] K.-J. Yoon and I. S. Kweon, "Stereo matching with the distinctive similarity measure," in *Proc. IEEE ICCV*, Oct. 2007, pp. 1–7.
[23] J. Gibson and O. Marques, "Stereo depth with a unified architecture GPU," in *Proc. IEEE Comput. Soc. Conf. CVPRW*, Jun. 2008, pp. 1–6.
[24] A. F. Bobick and S. S. Intille, "Large occlusion stereo," *Int. J. Comput. Vis.*, vol. 33, no. 3, pp. 181–200, 1999.
[25] C. L. Zitnick and T. Kanade, "A cooperative algorithm for stereo matching and occlusion detection," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 22, no. 7, pp. 675–684, Jul. 2000.
[26] M. Baydoun and M. A. Al-Alaoui, "Enhancing stereo matching with varying illumination through histogram information and normalized cross correlation," in *Proc. 20th IWSSIP*, Jul. 2013, pp. 5–9.
[27] J. Lu, S. Rogmans, G. Lafruit, and F. Catthoor, "Stream-centric stereo matching and view synthesis: A high-speed approach on GPUs," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 19, no. 11, pp. 1598–1611, Nov. 2009.
[28] K. Zhang, J. Lu, Q. Yang, G. Lafruit, R. Lauwereins, and L. Van Gool, "Real-time and accurate stereo: A scalable approach with bitwise fast voting on CUDA," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 21, no. 7, pp. 867–878, Jul. 2011.
[29] W. Yu, T. Chen, F. Franchetti, and J. C. Hoe, "High performance stereo vision designed for massively data parallel platforms," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 20, no. 11, pp. 1509–1519, Nov. 2010.
[30] J. Kowalczuk, E. T. Psota, and L. C. Pérez, "Real-time stereo matching on CUDA using an iterative refinement method for adaptive support-weight correspondences," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 23, no. 1, pp. 94–104, Jan. 2013.
[31] J. Stam, "Stereo imaging with Cuda," Tech. Rep. OpenVIDIA, version 0.2, Jan. 2008.
[32] K. Zhu, M. Butenuth, and P. d'Angelo, "Comparison of dense stereo using CUDA," in *Trends and Topics in Computer*. Berlin, Germany: Springer-Verlag, 2012, pp. 398–410.
[33] R. E. Schapire and Y. Freund, *Boosting: Foundations and Algorithms*. Cambridge, MA, USA: MIT Press, 2012.
[34] Y. Freund and R. E. Schapire, "A decision-theoretic generalization of on-line learning and an application to boosting," *J. Comput. Syst. Sci.*, vol. 55, no. 1, pp. 119–139, 1997.
[35] C. Seiffert, T. M. Khoshgoftaar, J. Van Hulse, and A. Napolitano, "RUSBoost: A hybrid approach to alleviating class imbalance," *IEEE Trans. Syst., Man, Cybern. A, Syst., Humans*, vol. 40, no. 1, pp. 185–197, Jan. 2010.
[36] A. Demiriz, K. P. Bennett, and J. Shawe-Taylor, "Linear programming boosting via column generation," *Mach. Learn.*, vol. 46, nos. 1–3, pp. 225–254, 2002.
[37] J. Friedman, T. Hastie, and R. Tibshirani, "Additive logistic regression: A statistical view of boosting," *Ann. Statist.*, vol. 28, no. 2, pp. 337–407, 2000.
[38] W. Wang and S. Goto, "Stereo matching with pixel classification and reliable disparity propagation," in *Proc. ISCAS*, May 2012, pp. 1891–1894.

[39] K. Zhang, J. Lu, and G. Lafruit, "Cross-based local stereo matching using orthogonal integral images," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 19, no. 7, pp. 1073–1079, Jul. 2009.
[40] W. Hoff and N. Ahuja, "Surfaces from stereo: Integrating feature matching, disparity estimation, and contour detection," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 11, no. 2, pp. 121–136, Feb. 1989.
[41] G. Pajares, J. M. de la Cruz, and J. A. López-Orozco, "Improving stereovision matching through supervised learning," *Pattern Anal. Appl.*, vol. 1, no. 2, pp. 105–120, 1998.
[42] G. Pajares and J. M. de la Cruz, "Stereovision matching through support vector machines," *Pattern Recognit. Lett.*, vol. 24, no. 15, pp. 2575–2583, Nov. 2003.
[43] T.-H. Sun, "Stereo matching using synchronous Hopfield neural network," *J. Chin. Inst. Ind. Eng.*, vol. 26, no. 4, pp. 276–288, 2009.
[44] M. S. Lew, T. S. Huang, and K. Wong, "Learning and feature selection in stereo matching," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 16, no. 9, pp. 869–881, Sep. 1994.
[45] H. Saito and M. Mori, "Application of genetic algorithms to stereo matching of images," *Pattern Recognit. Lett.*, vol. 16, no. 8, pp. 815–821, Aug. 1995.
[46] M. Chessa, V. Bianchi, M. Zampetti, S. P. Sabatini, and F. Solari, "Real-time simulation of large-scale neural architectures for visual features computation based on GPU," *Netw., Comput. Neural Syst.*, vol. 23, no. 4, pp. 272–291, 2012.
[47] L. Nalpantidis and A. Gasteratos, "Biologically and psychophysically inspired adaptive support weights algorithm for stereo correspondence," *Robot. Auton. Syst.*, vol. 58, no. 5, pp. 457–464, 2010.
[48] H. Hirschmuller and D. Scharstein, "Evaluation of cost functions for stereo matching," in *Proc. IEEE Conf. CVPR*, Jun. 2007, pp. 1–8.
[49] G. Bradski, "The OpenCV library" *Doctor Dobbs J.*, vol. 25, pp. 120–126, 2000.
[50] M. A. Al-Alaoui, "Direct approach to image edge detection using differentiators," in *Proc. 17th IEEE Int. Conf. Electron., Circuits and Syst.*, Athens, Greece, Dec. 2010, pp. 154–157.
[51] Y. S. Heo, K. M. Lee, and S. U. Lee, "Simultaneous color consistency and depth map estimation for radiometrically varying stereo images," in *Proc. IEEE 12th Conf. Comput. Vis.*, Sep./Oct. 2009, pp. 1771–1778.
[52] A. Hosni, C. Rhemann, M. Bleyer, C. Rother, and M. Gelautz, "Fast cost-volume filtering for visual correspondence and beyond," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 35, no. 2, pp. 504–511, Feb. 2013.
[53] A. Hosni, M. Bleyer, C. Rhemann, M. Gelautz, and C. Rother, "Real-time local stereo matching using guided image filtering," in *Proc. IEEE Int. Conf. ICME*, Jul. 2011, pp. 1–6.
[54] W.-H. Lee, Y. Kim, and J. B. Ra, "Efficient stereo matching based on a new confidence metric," in *Proc. 20th EUSIPCO*, 2012, pp. 1139–1143.
[55] J. Canny, "A computational approach to edge detection," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. PAMI-8, no. 6, pp. 679–698, Nov. 1986.
[56] M. Bleyer, C. Rother, P. Kohli, D. Scharstein, and S. Sinha, "Object stereo—Joint stereo matching and object segmentation," in *Proc. IEEE Conf. CVPR*, Jun. 2011, pp. 3081–3088.
[57] V. Podlozhnyuk, "Histogram calculation in CUDA," NVIDIA Corporation, Santa Clara, CA, USA, White Paper, version 1.1.1, Nov. 2007.
[58] V. Podlozhnyuk, "Image convolution with CUDA," NVIDIA Corporation, Santa Clara, CA, USA, White Paper, version 1.0, Jun. 2007.
[59] O. Kalentev, A. Rai, S. Kemnitz, and R. Schneider, "Connected component labeling on a 2D grid using CUDA," *J. Parallel Distrib. Comput.*, vol. 71, no. 4, pp. 615–620, 2011.

MOHAMMED BAYDOUN received the B.E. and M.E. degrees in mechanical engineering from Lebanese University, Beirut, Lebanon, and the American University of Beirut, Beirut, in 2005 and 2007, respectively, and the Ph.D. degree in electrical and computer engineering from the American University of Beirut in 2014, after working in the field of oil and gas. His research interests include digital image processing, digital signal processing, machine intelligence, and other related domains.

MOHAMAD ADNAN AL-ALAOUI received the B.S. degree in mathematics from Eastern Michigan University, Ypsilanti, MI, USA, in 1963, the B.S.E.E. degree from Wayne State University, Detroit, MI, USA, in 1965, and the M.S.E.E. and Ph.D. degrees in electrical engineering from the Georgia Institute of Technology (Georgia Tech), Atlanta, GA, USA, in 1968 and 1974, respectively. He served as an Assistant Project Engineer with the Bendix Radio Division, Department of Avionics, from 1966 to 1967. He was a Teaching Assistant with the School of Electrical Engineering and the School of Mathematics at Georgia Tech. He also joined the Department of Electrical Engineering at the Royal Scientific Society, Amman, Jordan. He served as an Assistant Professor with the Department of Electrical Engineering, American University of Beirut, from 1977 to 1980. He served as a Visiting Assistant and an Associate Professor with the Department of Electrical Engineering and Computer Science, University of Connecticut, Storrs, CT, USA. He served as an Associate Professor of Electrical Engineering with the Hartford Graduate Center, Hartford, CT, USA. He was the Chairman of the Department of Automatic Control at the Higher Institute for Applied Science and Technology, Damascus, Syria, from 1985 to 1988. In 1988, he rejoined the American University of Beirut, where he is currently a Professor of Electrical and Computer Engineering. He served as the Chair of the Department of Electrical and Computer Engineering from 2001 to 2004. His research interests are communications, computer vision, image and signal processing, pattern recognition and classification, neural networks, and machine intelligence.