Efficient Disparity Refinement Strategy for Stereo Vision Based on Edge-Aware Cost Processing

[1] Mr. Omar Murajia Ali [2] Mr. Ghaled M Daowd Bel kasem

https://orcid.org/0009-0000-7760-4441 1 Author

https://orcid.org/0009-0007-3568-523X 2 Author

[1] Department of Information Technology, Higher Institute of Science and Technology .Emsaad

[2] Department of Computer Science, The University of Tobruk

[1] **omar.murajia@histe.edu.ly** [2] **Khaled.belkasem@tu.edu.ly**

نهج فعّال لتنقية التباين في أنظمة الرؤية المجسّمة اعتمادًا على معالجة الكلفة الحساسة للحواف

أ. عمر مراجع علي [1] أ. خالد مطرود داوود [2]

[1] قسم تكنولوجيا المعلومات، المعهد العالي للعلوم والتقنية، أمساعد.

[2] قسم علوم الحاسوب، جامعة طبرق.

تاريخ الاستلام: 20-03-2026، تاريخ القبول: 05-04-2026، تاريخ النشر: 01-06-2026

Abstract:

Stereo matching is one of the most dynamic fields in computer vision. Cost aggregation is one of the popular methods for stereo matching due to its efficiency and effectiveness. This work aims to design an efficient aggregation cost computation algorithm and hardware based on the bilateral filter for stereo matching. We propose a new cost aggregation method based on the bilateral filter. Whereas the algorithms were generally divided into four steps: Matching cost, computation, Cost aggregation, Disparity computation, and post-processing. We have proposed a new method for calculating matching costs that combines the image gradient-based optimization cost with the census transform-based cost. In the cost aggregation stage, for each pixel, an adaptive model support window was generated across the base. Then, using the generated support window as a guide, a bilateral filter was used to aggregate the matching costs within the window. An approach was used to aggregate the matching costs within the window used to determine the ideal contrast for each pixel. Winner Take It All (WTA) for the aggregate cost. To further detect inaccurate matching results, a variance optimization framework based on multiple constraints was used to further detect inaccurate matching results. The hardware compliant algorithm model has been designed using MATLAB /Simulink software. This work also provides new insights into stereo matching algorithm design, and the proposed method is the most successful among all cost aggregation methods.

Keywords: Stereo Matching, Bilateral Filter, Cost Aggregation.

الملخص:

يُعدّ التوافق المجسم أحد أكثر المجالات ديناميكية في مجال رؤية الحاسوب. ويُعتبر تجميع التكاليف أحد الأساليب الشائعة للتوافق المجسم نظرًا لكفاءته وفعاليته. يهدف هذا العمل إلى تصميم خوارزمية فعّالة

لحساب تكلفة التجميع، بالإضافة إلى تصميم أجهزة تعتمد على المرشح الثنائي للتوافق المجسم. نقترح طريقة جديدة لتجميع التكاليف تعتمد على المرشح الثنائي. في حين أن الخوارزميات تُقسّم عادةً إلى أربع خطوات: تكلفة التوافق، والحساب، وتجميع التكاليف، وحساب التباين، والمعالجة اللاحقة، فقد اقترحنا طريقة جديدة لحساب تكاليف التوافق تجمع بين تكلفة التحسين القائمة على تدرج الصورة والتكلفة القائمة على تحويل التعداد. في مرحلة تجميع التكاليف، يتم إنشاء نافذة دعم نموذجية تكيفية لكل بكسل عبر القاعدة. ثم، باستخدام نافذة الدعم المُنشأة كدليل، يتم استخدام مرشح ثنائي لتجميع تكاليف التوافق داخل النافذة. تم استخدام نهج لتجميع تكاليف التوافق داخل النافذة المستخدمة لتحديد التباين الأمثل لكل بكسل. يتم تطبيق مبدأ "الفائز يأخذ كل شيء" (WTA) على التكلفة الإجمالية. لزيادة دقة الكشف عن نتائج المطابقة غير الدقيقة، تم استخدام إطار عمل لتحسين التباين قائم على قيود متعددة. صُمم نموذج الخوارزمية المتوافق مع الأجهزة باستخدام برنامج MATLAB/Simulink. يقدم هذا العمل رؤى جديدة في تصميم خوارزميات المطابقة المجسمة، وتُعد الطريقة المقترحة الأكثر نجاحًا بين جميع طرق تجميع التكاليف.

الكلمات المفتاحية :المطابقة المجسمة، المرشح الثنائي، تجميع التكاليف.

1. Introduction

STEREO vision adds a sense of depth, which is essential in a variety of applications. Stereo vision has several benefits, the most significant of which is the ability to derive a 3D definition from digital images. In this field, stereo matching is the most researched subject. It begins by taking images from both cameras as input and then searching for matching pixels [1]. As a result, 2D positions are converted to 3D depths, yielding a 3D estimate of the scene. In machine vision, stereo vision is linked to a biological term known as stereopsis, which refers to the depth that is perceived when an image is viewed by two eyes and ordinary binocular vision. Using stereoscopic vision, we can perceive the world and discover objects moving in our direction or away from us [2]. Nowadays, stereo vision has a wide range of applications, particularly in domains where distance and shape determination are needed, such as the extraction of a 3D object's location in robotics. The device can distinguish occluding image components used for object recognition by obtaining depth information. Another application employs stereo vision techniques to collect data from aerial surveys, allowing scientists to create contour maps and 3D heliographic maps [3]. Depending on the processing time, these purposes can be divided into two categories: static scene description and dynamic scene description [4]. With processing time, accuracy is more important in the static scene group. Image pairs are usually received via a special method and then reconstructed for cartography, auto crash scene reconstruction, crime scene reconstruction, 3D models for architecture, and other applications [5]. Of course, a certain degree of precision is needed. Impediment avoidance (e.g., pedestrians) and autonomous navigation are two possible features [6].

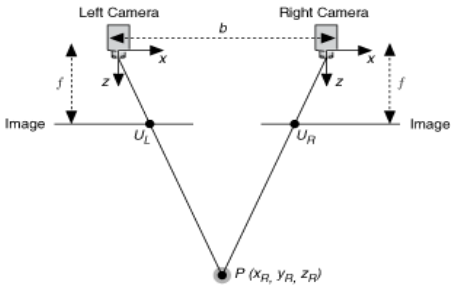

Fig. 1. Stereo Vision System.

Cost aggregation is the most important stage and overall performance of a stereo vision disparity map method, especially for local approaches [7]. Cost aggregation aims to eliminate matching uncertainty to the greatest extent possible. When calculating the matching cost for a single pixel, the information obtained is inadequate for precise matching, hence cost aggregation is required [8].

2. Previous Study

Stereo Matching

Many local stereo matching methods have been developed for creating high-quality disparity maps by establishing the weighting function w (p; q), which can implicitly measure the similarity of disparity values between pixels p and q. [9] suggested an adaptive (soft) weight strategy that takes advantage of color and spatial similarity measurements with related color images, and it may be thought of as a version of joint bilateral filtering. It is simple to build and delivers excellent accuracy, but it is quite complex due to the nonlinearity of the weighting function computation. Color segmentation-based cost aggregation was also provided with the assumption that pixels within the same segment had equal disparity values [10]. A shape-adaptive window, which consists of many horizontal line segments spanning several neighboring rows, was employed in cross-based techniques. Based on color similarity and an implicit connection requirement, the shape of the matching window N(p) is estimated, and a hard weighting value (1 or 0) is finally employed [11].

The cost aggregation complexity can be expressed as O(NBL), where N and B are the size of the input picture and the matching window N(p), respectively, and L is the search range, i.e., the number of discrete labels (e.g., disparity hypotheses) [12]. Many strategies have been developed in terms of the size of picture N and the matching window B to lessen the complexity of cost aggregation. In stereo matching, [13] introduced a new multiscale methodology for achieving reliable cost aggregation. On the coarse image and cost domain, they tried to minimize the complexity by utilizing smaller matching windows. [14]-[15] used an approximation of the bilateral filter to reduce the complexity of the adaptive support weight technique. Although the complexity is independent of the size of the matching window, the quality of a grey image utilized in the bilateral grid suffers.

Cost Aggregation

The most essential stage and overall performance of a stereo vision disparity map algorithm, especially for local techniques, is cost aggregation. The goal of cost aggregation is to reduce matching uncertainty as much as possible [16] The information gained for a single pixel when calculating the matching cost is insufficient for precise matching, hence cost aggregation is required.

In most cases, [17] is used to aggregate the matching costs at pixel and disparity when a square support window is centered on a pixel for standard ASW. The window size is a parameter that is set by the user. The function's value denotes the likelihood that a pixel will have a disparity value like the window's center pixel, which is a target pixel with a disparity value. If pixels and disparity values are equal, return a value of "1"; else, return a value of "0." [18] used a bilateral filter to create a trilateral filter based on the ASW technique. They also introduced a new weighted term to improve the object boundary robustness. As implemented by [19] in an ASW application, a larger weight will be assigned to a pixel if its intensity is more similar to that of the anchor pixel and it is placed at a smaller distance from the anchor pixel.

When compared to the prior approaches disclosed in their literature, our method produces a disparity map with well-preserved object boundaries and high accuracy. [20] conducted a thorough analysis of ASW locations. They ran the test on a GPU to see if the speed and processing efficiency were fast enough for real-time responses. According to their findings, the ASW method offers excellent results in terms of both computing efficiency and the quality of the disparity maps generated. A new methodology based on the ASW technique was developed by [21]. They calculated a weighting factor by combining it with the quantified gestalt law. A correlation weight, in general, shows the closeness, similarity, and continuity of both input pictures (i.e., left and right images).

Table 1. Time And Storage Complexity of the Proposed Feature-Vector-Based Cost Aggregation Algorithm

Time complexity	
Feature vector computation	Oa (2 × W × H)
Feature vector comparison	Ob (W × H × M) for C and Ob (W × H × (2w+ 1)) for P
Construction and update of ⌈	Oc (2w + 1)2 × W × M and Oc (3 × W × H×M), respectively
Construction and update of ∅	Oc (2w+1)2/2× W and Oc (3/2 × W × H × (2w + 1)), respectively
Cost aggregation for all pixels	Oc (3 × (2w + 1) × W × H × M)

3. Research Method

Proposed Algorithm

The proposed method consists of four steps. (1) Matching cost computation: We propose a new approach for calculating matching costs that combines improved image gradient-based cost and improved census transform-based cost. (2) Cost aggregation: first, for each pixel, a cross-based adaptive form support window is created. Then, using the generated support window as a guide, a bilateral filter is used to aggregate the matching costs within the window. (3) Disparity selection: a winner-take-all (WTA) According to the aggregated cost, an approach is employed to identify the ideal disparity for each pixel. (4) Multi-constraints-based disparity refinement framework. To further detect the incorrect matching results, a multi-constraints-based disparity refining framework is used to further detect the inaccurate matching results. This framework includes outlier identification with a left-right consistency testing procedure, occlusion/mismatch management, a weighted median filter, and subpixel enhancement.

Real-Time Stereo Vision Implementations

There are two types of dense matching algorithms: local and global. The implementation of global algorithms in hardware is neither appealing nor easy. Global methods are time and computationally intensive due to their iterative nature. This is also the reason why parallel

systems cannot be used to execute them. Global algorithms, on the other hand, necessitate unusual implementations rather than plain and straightforward ones.

Local methods, on the other hand, could benefit greatly from the use of such parallel and straightforward systems. Parallelism and simplicity are important features of dedicated hardware implementations that can cut down on required run times. Many works explain hardware-based local methods. Most of them have one thing in common: they use a basic algorithm and make heavy use of computation concurrency. During the hardware architecture creation process, custom choices are made to improve performance. Figure 2 shows a simplified block diagram of a hardware implementable stereo correspondence algorithm.

Field-programmable gate arrays (FPGAs) and application-specific integrated circuits (ASICs) are the two main types of hardware implementations. A review of recent literature indicates that FPGA implementations are superior. This is because the fabrication and testing of ASIC implementations take a long time and costs a lot of money. Furthermore, there is almost no space for future progress and changes. FPGA, on the other hand, allows for faster prototyping, is much less costly, and is easily adaptable to new requirements. In this way, FPGA combines the best features of both hardware and software solutions.

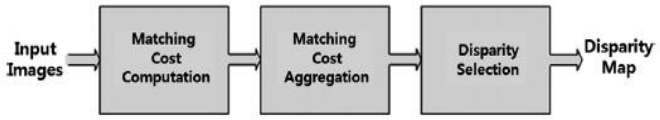

Fig. 2. Generalized block diagram of a hardware implementable stereo matching algorithm.

Stereo Matching Algorithm

Stereo matching algorithms aim to find an estimate of the depth inside a scene based on rectified stereo pairs of images and solve the correspondence problem. The stereo matching algorithm is fed two images of the same scene, a reference image and a target picture, each representing the picture from a different x-axis perspective. The purpose is to get the disparity, or relative depth information, for each pixel along the x-axis with precision. Objects corresponding to pixels of larger disparity are closer to the camera(s) than objects corresponding to pixels of lesser disparity, hence the results are frequently used to estimate the distance to an object in the picture.

Even though the algorithms are numerous and diverse, they share a common thread. The four steps of a stereo algorithm are usually as follows:

1. Matching cost computation
2. Cost aggregation
3. Disparity computation
4. Post-processing

To determine the level of matching between two pixels, all stereo matching algorithms require a cost criterion. The matching cost computation determines if the values of two pixels in a scene belong to the same spot in the scene. As a result, stereo matching cost computation can be characterized as a way of estimating each point's parallax between the left and right images. For all pixels under consideration, the matching cost is evaluated for each pixel. If the stereo pairings are accurately rectified, this matching can be done with a one-dimensional horizontal search. As a result, rectification is important in stereo matching. Because the search for correspondences can be limited to a single line rather than the entire image space, the needed time and search range are reduced.

The most critical stage in deciding the overall performance of a stereo vision disparity map algorithm, especially for local approaches, is cost aggregation. The goal of cost aggregation is to reduce matching uncertainty as much as possible. Because the information gained for a single pixel while calculating the matching cost is insufficient for exact matching, cost aggregation is required. By summing up the matching costs over a support region, local approaches aggregate the matching costs.

Cost Aggregation

The most essential stage and overall performance of a stereo vision disparity map algorithm, especially for local techniques, is cost aggregation. The goal of cost aggregation is to reduce matching uncertainty as much as possible. The information gained for a single pixel when calculating the matching cost is insufficient for precise matching, hence cost aggregation is required.

This stage is broken into two parts: 1. The local weight matrix is determined using the covariance matrix based on the local grey value. 2. The matching cost is aggregated using a bilateral filter.

In the adaptive kernel regression, $K_{H_{d^{steer}}}\ (x_d - x)$ is the data-adapted kernel. It is determined by the pixel spatial position and pixel intensity, thus effectively protecting the boundary. $H_{d^{steer}}$ called steering matrices. It can express by:

$$H_d^{steer} = h\mu_d C_d^{\frac{1}{2}} \tag{1}$$

Where C_d is covariance matrix based on differences in the local gray-values. It can be calculated from.

$$C_d = \gamma_d U_{\theta_d} \Lambda_d {U_{\theta_d}}^T \tag{2}$$

Where U_{θ_d} is the rotation matrix, Λ_d is the expansion matrix and γ_d determine the scale parameters.

The form of the locally adapted kernel is thus determined by the covariance matrix. Since it uses large windows in the smooth region and small windows in the depth discontinuities region, these locally adapted kernels can protect the image edges. “Using the covariance matrix to calculate the weighted kernel is an efficient way to boost the directed filter's effect.

Let $G = [G_R, G_G, G_B]^T$ be the guidance color image, the matching cost to be filtered is $Z(x_d)$, the image obtained after filtering is $' Z(x_d)$. It is assumed that in each disparity *d*, for the center point is k and the support window k_ω with radius is *r*, $' Z(x_d)$, can be expressed as:

$$z'(x_d) = a_k \bullet G(x_d) + b_k \tag{3}$$

After the cost aggregation, the aggregated cost $' Z(x_d)$ at the disparity *d* is obtained.

One of the core issues in constructing the cross-based adaptive shape support window is how to design proper rules to expand the pixel *p* to its neighbors. If we wish to build the adaptive support region of pixel *p*, we must first calculate the color difference Dc (*p*, *q*) and the spatial distance Ds (*p*, *q*) between pixel p and q, as shown in Figure 3. The pixel q is classified as textured region pixel or texture less area pixel based on the spatial distance Ds (*p*, *q*). The pixel q is designated as texture less region pixel if Ds (*p*, *q*) is greater than the spatial distance threshold d_{Lim}. Pixel q is otherwise labeled as textured region pixel. The color similarity thresholds should be higher in densely textured regions and decrease as the spatial distance increases. The color similarity thresholds in texture-less zones might be lower and should

decrease as the spatial distance increases. Based on these findings, we use the following two rules to compute the color similarity threshold for each pixel:

$$Rule\ 1\ \tau^{large}\left(Ds(p,q)\right) = -\frac{\tau 1}{L} \times Ds(p,q) + \tau 1, if\ Ds(p,q) \leq d_{Lim} \quad (4)$$

$$Rule\ 2\ \tau^{samll}\left(Ds(p,q)\right) = -\frac{\tau 2}{L} \times Ds(p,q) + \tau 2, otherwise \quad (5)$$

For densely textured regions, L1 is a small spatial distance constant and t1 is a large color similarity constant in the following criteria. For low texture regions, L2 is a relatively large spatial distance constant and t2 is a relatively small color similarity constant. The adaptively determined color similarity thresholds for densely texture and texture fewer regions are τ^{large} (Ds (p, q)) and τ^{samll} (Ds (p, q)). Rule 1 is a constrained condition that ensures that just a small portion of the richly textured regions are included. While Rule 2 is followed to ensure that the texture-less zones contain as many points from the same depth as possible.

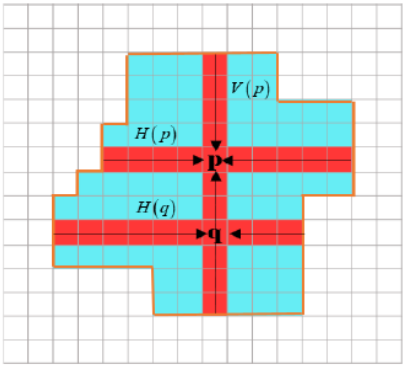

Fig. 3. Cross-based support region construction for cost aggregation.

4. Results And Discussion

Stereo Algorithms

The goal of the proposed method is to create an efficient cost-aggregation algorithm and hardware is designed based on a bilateral filter for stereo matching. Figure 4 shows the overall framework of the proposed algorithm. Our algorithm performs the following four steps: 1) constructing cost volume; 2) cost aggregation; 3) disparity selection; 4) disparity refinement.

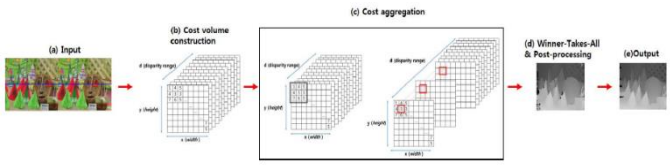

Fig. 4. The overall framework of our method.

The similarity measure of two pixels from the left and right images is referred to as cost. Many algorithms also perform some form of pre-and post-processing in addition to these phases. Figure 5 depicts two identical cameras, each with a focal length of f, positioned at CL

and CR, and vertically aligned at Y = 0. The baseline distance b separates the cameras along the X-axis. In the left and right images, the object point, which is positioned at $(0\ 0\ Z\ )^T$, is projected to the two places PL and PR, respectively.

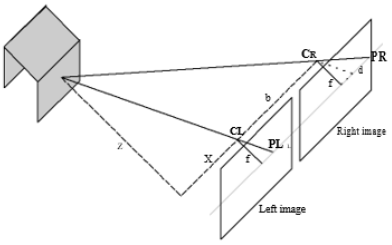

Fig. 5. A basic stereo rig.

Local, global, and semi-global stereo algorithms can also be classified into three types. Local algorithms act on individual pixels, attempting to identify the greatest disparity match based on a tiny area surrounding the target pixel. Global algorithms, on the other hand, seek to reduce global energy (cost) and, as a result, determine the correct disparity for each pixel. This type of 2D-optimizing issue is demonstrated to be NP-hard1. There are scan line1-based techniques that address a similar energy-minimization problem for each scan line independently to circumvent this.

All techniques assume that all surfaces in the images are Lambertian reflectors that are piecewise smooth, either implicitly or explicitly. This assumption states that a point's reflected intensity is constant in all viewing directions. Furthermore, it implies that a plane may characterize all surfaces. Although this assumption is incorrect, it has proven to be effective. A stereo algorithm's output is essentially a disparity map, in which the disparity values allocated to the pixels of the input image are connected to the distance from the camera. Figure 6 shows the famous Tsukuba stereo pair (top) and its corresponding ground-truth disparity map (bottom). High intensities represent pixels located close to the camera.

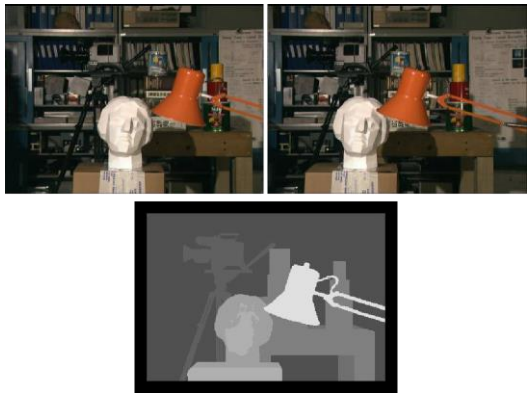

Fig. 6. Tsukuba stereo set and the corresponding ground truth disparity map.

Cost Aggregation Algorithm

Bilateral filter kernel weight is a quick and effective solution for judging p and q in the same object. The BF weight between two nearby pixels can be interpreted as the probability of the two pixels belong to the same object. The weight value is derived by the distance (the x term, also written as ws) and intensity similarity (the I term, also written as wr) between pixels p and q in the bilateral filter weight kernel function (Figure 7 and function (6)). Simply put, the greater

the weight q to p, the shorter the distance and the lower the intensity difference. So, we can just create a whole-image size bilateral filter kernel from the original color image, and then put it on the map of costs to aggregate them. Every pixel gets additional costs multiplied by weights with whole image pixels.

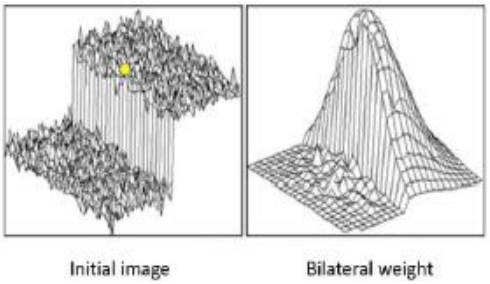

Fig. 7. Bilateral filter weight kernel schematic diagram.

$$W_{pq}^{bf}(I) = \frac{1}{K_i}\exp\left(-\frac{\|x_p - x_q\|^2}{\sigma_s^2}\right)\exp\left(-\frac{\|I_p - I_q\|^2}{\sigma_r^2}\right) = \frac{1}{K_i} w_s w_r \tag{6}$$

We define wn as the weight between nearby two pixels initially and then combine multiple wn to generate the ultimate weight (w (p, q)) between any two pixels in this new weight structure. In functions (7) and (8), a quick definition is provided (8). In 4-connect regions, wn is defined as the distance between each pixel and its neighbors in function (7). (That is, 4 pixels up, down, left, and right). (8) is an expansion of (7), containing all weights between all pixels in the entire map. Then we define W as the set of w (p, q) weight maps.

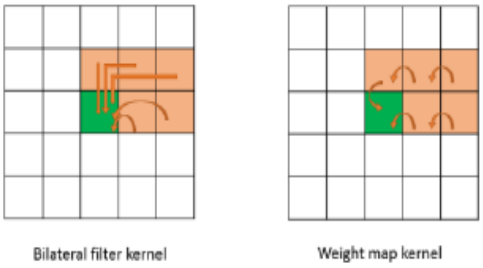

Fig. 8. Cost transferring by bilateral filter and proposed to weigh map.

$$w_n(i,j) = e^{\frac{-|I_L(i) - I_L(j)|}{\sigma}} \qquad j \in i's \text{ 4-connsected area} \tag{7}$$

$$w(p,q) = w_n(p,q_1)\cdot w_n(q_1,q_2)\cdot w_n(q_2,q_3)\cdots w_n(q_{n-1},q) \tag{8}$$

Figure 8 shows an example of a comparison. The green pixel would be the sum of the five orange costs. Every orange arrow has a starting pixel q and an end pixel p, regardless of whether it is a BF kernel or a WM kernel. Q multiplies w (p, q) and then adds to p in the aggregation process. In instances of BF, it must calculate 5 Wpq bf, which entails 5 ws and wr, for a total of 10 exponentials. It only needs to calculate 5 w (p, q) in the proposed WM scenario, totaling 5 exponentials. In other words, the WM treats wn values as wr and wn numbers as ws. That means WM can achieve a similar outcome to BF while using half the computational resources. Furthermore, WM can avoid giving too much weight to "not connected but close comparable pixel".

However, because all w (p, q) is determined by neighboring two pixels, pixels at gradually

shifting borders may get costs from various objects. We add the Canny edge to our weight map to solve this problem. The Canny edge is calculated by pixels in a 3*3 or larger kernel (in the proposed method, 5*5), resulting in superior object contour recognition. As seen in the function, all weights via edge pixels are set to 0. The measure "protects" objects from "invasion" by large objects. In other words, even though the weights at the edges are extremely little, the integral cost of the object is still incredible if the object is very large. The high cost may encroach on nearby objects, particularly in the color-changing zone. That is, the Canny edge can keep object contours complete.

$$w_c(p,q) = \begin{cases} 0 & p,q \in Canny\ edge \\ w(p,q) & otherwise \end{cases} \quad (9)$$

$$C_{p2}(d) = \sum_{q \in S} w_c(p,q) C_{p1}(d) \quad (10)$$

For the proposed algorithm, four sets of standard stereo image pairs were used for its evaluation, namely Tsukuba, Venus, Cones, and Teddy. The proposed method has been tested using the parameters, Tsukuba: window size 11 and Max. Disparity 15, Venus: window size 25 and Max.Disparity 19, Cones: window size 21 and Max. Disparity 59 and Teddy: window size 21 and Max.Disparity 59. The result shows in Figure 9.

Figure 9 shows the disparity map generated by the proposed algorithm. By visually comparing the disparity map obtained by the proposed algorithm with the ground-truth disparity map, the matching percentage rate is high, and the matching effect in the depth discontinuous region is better.

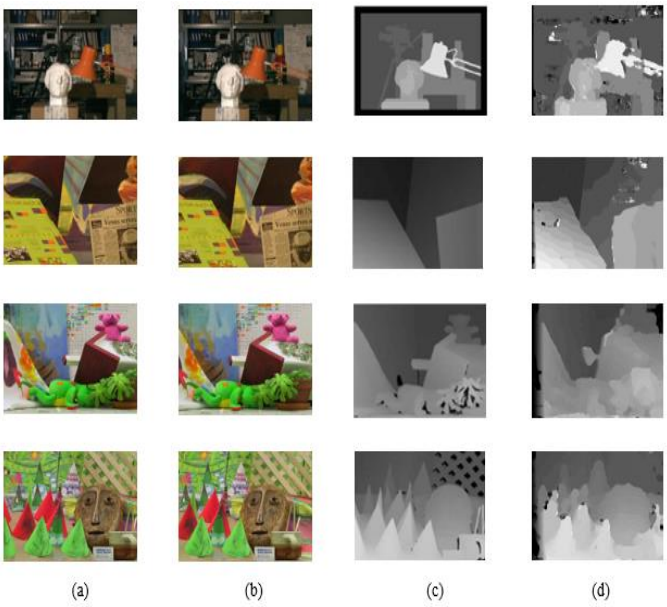

Fig. 9. The disparity maps of the proposed algorithm by using Tsukuba, Venus, Teddy, and Cones:(a) Left images, (b)Right images, (c) ground-truth images, (d) result for the proposed algorithm.

To find the correspondence between the right pixel and the left pixel, a matching cost needs to be computed for every right pixel candidate. The easiest matching costs assume that the gray levels at homologous pixels are equal. Common pixel-based matching costs consist of absolute differences, squared differences, sampled-insensitive or absolute or truncated variations of these. Some costs are insensitive to variations in the camera parameters.

Fig. 10. Visual comparison with conventional cost aggregation methods and proposed cost aggregation methods.

The data sets computed using conventional filter-based cost aggregation methods and the proposed cost aggregation method are presented in Figure 10, which shows that the proposed cost aggregation method provided more accurate results and error is well removed.

Cost Aggregation Hardware

Bilateral filtering is used in computer vision systems to filter images while preserving edges and has become ubiquitous in image processing applications. Those applications include denoising while preserving edges, texture and illumination separation for segmentation, and the aggregate cost.

Bilateral filtering is simple in concept: each pixel at the center of a neighborhood is replaced by the average of its neighbors. The average is computed using a weighted set of coefficients. The weights are determined by the spatial location in the neighborhood (as in a traditional Gaussian blur filter), and the intensity difference from the center value of the neighborhood.

These two weighting factors are independently controllable by the two standard deviation parameters of the bilateral filter. When the intensity standard deviation is large, the bilateral filter acts more like a Gaussian blur filter, because the intensity Gaussian is less peaked. Conversely, when the intensity standard deviation is smaller, edges in the intensity are preserved or enhanced.

This model provides a hardware-compatible algorithm using embedded MATLAB /Simulink software.

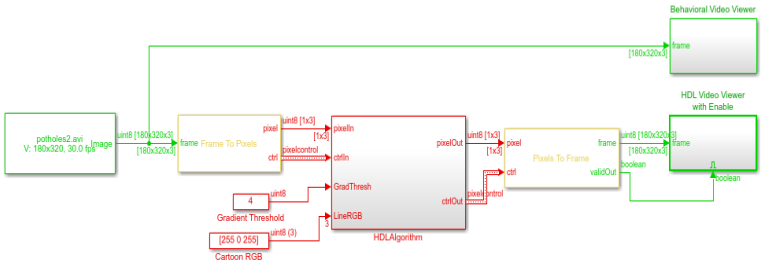

Fig. 11. Model a Hardware Compatible Algorithm – Simulink.

Step 1: Establish Parameter Values

To achieve a modest Gaussian blur of the input, choose a relatively large spatial standard deviation of 3. To give strong emphasis to the edges of the image, choose an intensity standard deviation of 0.75.

The intensity Gaussian is built from the image data in the neighborhood, so this plot represents the maximum possible values. Note the small vertical scale on the spatial Gaussian plot.

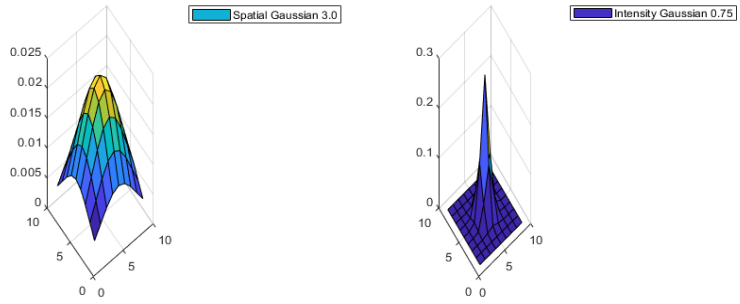

Fig. 12. Gaussian plot.

Step 2: Filter the Intensity Image

The model converts the incoming RGB image to intensity using the Color Space Converter block. Then the grayscale intensity image is sent to the Bilateral Filter block, which is configured for a 9-by-9 neighborhood and the parameters established previously.

The bilateral filter provides some Gaussian blur but will strongly emphasize larger edges in the image based on the 9-by-9 neighborhood size.

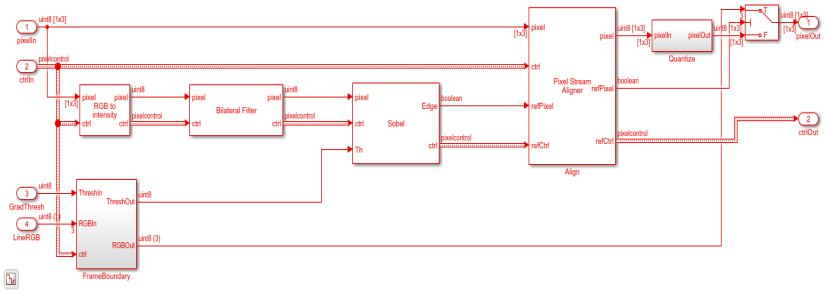

Fig.13. Bilateral filter Verilog HDL Algorithm – Simulink.

Step 3: Compute Gradient Magnitude

Next, the Sobel Edge Detector block computes the gradient magnitude. Since the image was pre-filtered using a bilateral filter with a large neighborhood, the smaller, less important edges in the image will not be emphasized during edge detection.

The threshold parameter for the Sobel Edge Detector block can come from a constant value on the block mask or from a port. The block in this model uses the port to allow the threshold to be set dynamically. This threshold value must be computed for your final system, but for now, you can just choose a good value by observing results.

Synchronize the Computed Edges

To overlay the threshold edges onto the original RGB image, you must realign the two streams. The processing delay of the bilateral filter and edge detector means that the threshold edge stream and the input RGB pixel stream are not aligned in time.

The Pixel Stream Aligner block brings them back together. The RGB pixel stream is connected to the upper pixel input port, and the binary threshold image pixel is connected to the reference input port.

The block delays the RGB pixel stream to match the threshold stream. The 9-by-9 bilateral filter has a delay of more than 4 lines, while the edge detector has a delay of a bit more than 1 line.

Color Quantization

Color quantization reduces the number of colors in an image to make processing it easier. Color quantization is primarily a clustering problem because you want to find a single representative color for a cluster of colors in the original image.

These algorithms require that you know beforehand all the colors in the original image. In pixel streaming video, the color discovery step introduces an undesirable frame delay. Color quantization is also generally best done in a perceptually uniform color space such as L*a*b. When you cluster colors in RGB space, there is no guarantee that the result will look representative to a human viewer.

The Quantize subsystem in this model uses a much simpler form of color quantization based on the most significant 4 bits of each 8-bit color component. RGB triples with 8-bit components can represent up to $2^{24} = 2^8.2^8.2^8$ colors but no single image can use all those colors. Similarly, when you reduce the number of bits per color to 4, the image can contain up to $2^{12} = 2^4.2^4.2^4$ colors. In practice, a 4-bit-per-color image typically contains only several hundred unique colors.

After shifting each color component to the right by 4 bits, the model shifts the result back to the left by 4 bits to maintain the 24-bit RGB format supported by the video viewer. In a Verilog HDL system, the next processing steps would pass on only the 4-bit color RGB triples.

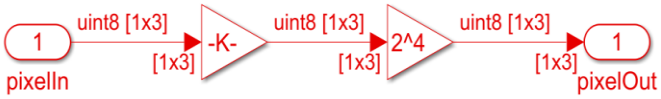

Fig. 14. Quantize – Simulink.

Overlay the Edges

A switch block overlays the edges on the original image by selecting either the RGB stream or an RGB parameter. The switch is flipped based on the edge-detected binary image.

Parameter Synchronization

In addition to the pixel and control signals, two parameters enter the Verilog HDL Algorithm subsystem: the gradient threshold and the line RGB triple for the overlay color.

The Frame Boundary subsystem provides run-time control of the threshold and the line color. However, to avoid an output frame with a mix of colors or thresholds, the subsystem registers the parameters only at the start of each frame.

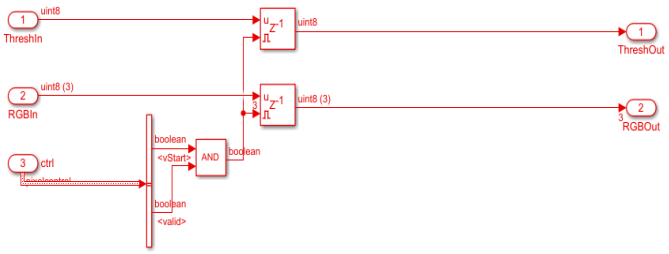

Fig. 15. Frame Boundary – Simulink

Figure 16 shows a method for aggregating costs using a proposed bilateral filter using MATLAB/Simulink software.

The resulting images from the simulation show bold lines around the detected features in the input video. This algorithm is suitable for FPGA implementation.

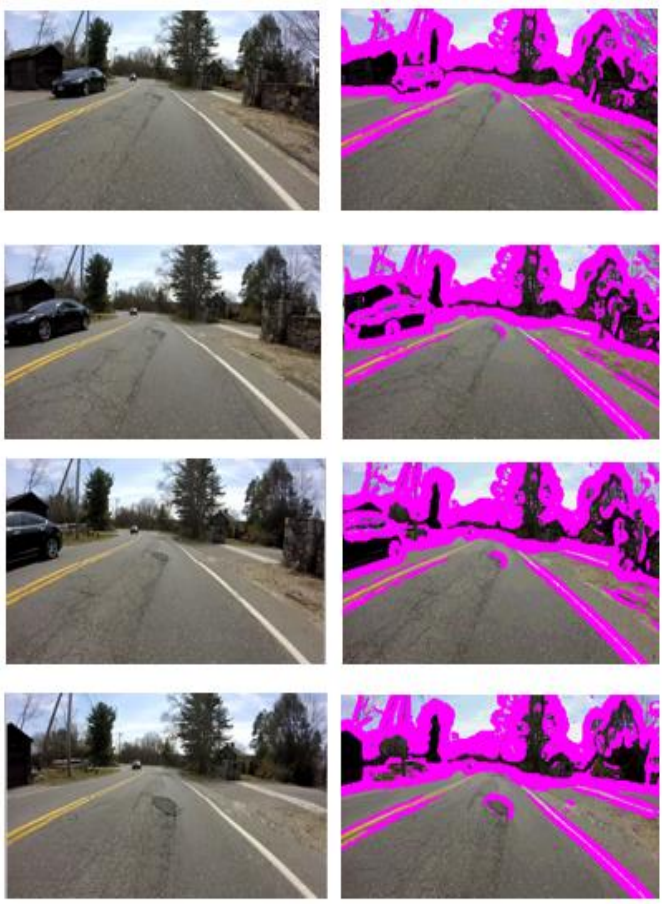

Fig. 16. Proposed cost aggregation method using Bilateral filter.

5. Conclusion

Overall Conclusion

The purpose of this work was to design an efficient aggregation cost computation algorithm and hardware based on a bilateral filter for stereo matching. To do this, an extensive literature study was initially performed. The study showed that although there are many well-performing methods for cost aggregation, none of them had the requirements and hardware based on a bilateral filter for stereo matching. The main difference is the configuration of the stereo rig. This project-focused proposed method consists of four steps. Matching cost computation, Cost aggregation, Disparity computation, and multi-constraints-based disparity refinement framework.

The proposed method exploited the bilateral filter to the aggregate cost volume. The hardware compliant algorithm model has been designed using MATLAB /Simulink software. The images generated from the simulation have shown bold lines around the detected features in the input video and this algorithm is suitable for FPGA implementation.

Future Work

The purpose of future work is to improve the performance of current research work. Many improvements can be made in this study. Therefore, the quality of stereo matching algorithms

can be developed and improved through cost-aggregation methods from the perspective of a histogram.

References:

1. Osswald, M., et al. (2017). "A spiking neural network model of 3D perception for event-based neuromorphic stereo vision systems." Scientific Reports 7(1): 1-12.
2. Banz, C., et al. (2019). Architectures for stereo vision. Handbook of Signal Processing Systems, Springer: 577-612.
3. Bebeşelea-Sterp, E., et al. (2017). "A comparative study of stereo vision algorithms." Int. J. Adv. Comput. Sci. Appl. 8: 1-17.
4. Jayakumar, D. R. (2018). "Real-Time Disparity Map Generation for Stereovision Based Obstacle Detection System Using SAD Algorithm." International Journal of Pure and Applied Mathematics 119(12): 14501-14507.
5. Jeon, H.-G., et al. (2018). "Depth from a light field image with learning-based matching costs." IEEE transactions on pattern analysis and machine intelligence 41(2): 297-310.
6. Kim, D. T., et al. (2020). "Designing a New Endoscope for Panoramic-View with Focus-Area 3D-Vision in Minimally Invasive Surgery." Journal of Medical and Biological Engineering 40(2): 204-219.
7. Koringa, P. A. and S. K. Mitra (2019). Class Similarity-Based Orthogonal Neighborhood Preserving Projections for Image Recognition. International Conference on Pattern Recognition and Machine Intelligence, Springer.
8. Li, G., et al. (2019). Matching algorithm and parallax extraction based on binocular stereo vision. Smart innovations in communication and computational sciences, Springer: 347-355.
9. Liu, H., et al. (2020). "Improved cost computation and adaptive shape guided filter for local stereo matching of low texture stereo images." Applied Sciences 10(5): 1869.
10. Liu, J., et al. (2021). "Segmentation of act white region in uterine cervical image based on deep learning." Technology and Health Care (Preprint): 1-14.
11. Logothetis, F., et al. (2020). "A CNN based approach for the near-field photometric stereo problem." arXiv preprint arXiv:2009.05792.
12. Ma, J., et al. (2021). "Image matching from handcrafted to deep features: A survey." International Journal of Computer Vision 129(1): 23-79.
13. Nakamura, M. and N. Fukushima (2017). Fast implementation of box filtering. Proc. International Workshop on Advanced Image Technology (IWAIT).
14. O'Byrne, M., et al. (2018). "A Stereo-Matching Technique for Recovering 3D Information from Underwater Inspection Imagery." Computer-Aided Civil and Infrastructure Engineering 33(3): 193-208.
15. Yu, H., et al. (2020). "A scalable region-based level set method using an adaptive bilateral filter for noisy image segmentation." Multimedia Tools and Applications 79(9): 5743-5765.
16. Yuan, W., et al. (2021). "Efficient local stereo matching algorithm based on fast gradient-domain guided image filtering." Signal Processing: Image Communication 95: 116280.
17. Zacharatou, E. T., et al. (2017). "GPU rasterization for real-time spatial aggregation over

arbitrary polygons." Proceedings of the VLDB Endowment 11(3): 352-365.
18. Zhang, F., et al. (2019). Ga-net: Guided aggregation net for end-to-end stereo matching. Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition.
19. Zhu, S. and L. Yan (2017). "Local stereo matching algorithm with efficient matching cost and adaptive guided image filter." The Visual Computer 33(9): 1087-1102.
20. Yang, W.-J., et al. (2019). An adaptive cost aggregation method based on bilateral filter and canny edge detector with segmented area for stereo matching. International Workshop on Advanced Image Technology (IWAIT) 2019, International Society for Optics and Photonics.
21. Ramírez-Hernández, L. R., et al. (2020). Stereoscopic vision systems in machine vision, models, and applications. Machine Vision and Navigation, Springer: 241-265.