Stereo Matching Using improved SGM cost propagation method

Chia-Pin Tseng, BoYang Wang

***Abstract*— Stereo vision is used in robotic vision and autonomous vehicle navigation. The Semi-Global Matching (SGM) algorithm is one of the most widely used stereo vision algorithms. The objective of this paper was to introduce a branch cost propagation concept into SGM algorithm. The branch mechanism gives each path ability to actively search and collect feature information. The active information boost the meaningful signal energy and help to overcome the noise. We also implement the algorithm on Xilinx Spartan-6 FPGA This implementation utilized 6907 slices registers, 11748 LUTs, and the maximum pixel clock frequency reached to 80.32 MHz. Compare to SGM algorithm our branch SGM used 10% more hardware resources, and reduced 10%~30% average error under noisy input disparity pairs.**

Keywords: stereo vision, Semi-Global Matching

INTRODUCTION

Stereo Vision plays a key role in human cognizing and observing the world. People are expected to simulate human vision functions on applications. With the development of computer technology, digital image processing, computer graphics, and artificial intelligence, human stereo vision simulation will be improved and applied. This technology can be utilized in many different fields, such as automatic tracking and recognition of moving targets, autonomous vehicle navigating system, and robot vision system. Human binocular stereo vision system is an intelligent but complex system. The purpose of computer vision research is to cognitive three-dimensional information by analyzing two-dimensional image information from the machine. This is not just the ability of obtaining the geometric information, such as shape, location, and attitude, but computer can also do analysis, storage, and recognition.

To implement a real-time stereo vision system, there are two mainstream computation platforms, GPU and FPGA. GPU uses much understandable C based language by CUDA or OpenCL with complete hardware access API, such as memory, camera, and monitor. In addition, GPU based stereo vision system can be easily accessed by the software implementation in this paper. However, considering the speed and power consumption, GPU based computation is not as good as FPGA. Instead of this, for FPGA based computation, developer can utilize register level memory to avoid memory interfacing delay. The real-time ability for stereo matching system is essential especially for those applications on the driver-less vehicles and unmanned aerial vehicle (UAV), in which any frame counts.

Local methods, which are based on correlation can have very efficient implementations that are suitable for real time applications. However, these methods assume constant disparities within a correlation window, which is incorrect at discontinuities and leads to blurred object boundaries. Semi-Global Matching supports pixel-wise matching for maintaining sharp object boundaries. In addition, by the smoothness constraint mechanism, it avoided the aperture problem and provided ability to tolerate various sizes of objects in this simple approach. The matching result from Semi-Global Matching (SGM) method performs much better than local methods and is almost as accurate as global methods. However, SGM is much faster than global methods.

SGM also provides hardware friendly structure, parallel processing and sequential pixel accessing. Although, SGM needs high memory bandwidth to access intermediate results, FPGA is able to use register and block RAM to overcome the problem. Complete SGM method would use 8 to 16 directional scan line to get the best result. SGM method aggregates several direction of scan-line optimization matching cost to produce final disparity value. Each scan line information uses dynamic programming strategy to minimize the cost. The scan-line optimization uses two penalty data P1 and P2 to constrain the smoothness and it's usually used to overcome the noise caused discontinuities.

This paper proposes a novel method to improve SGM method on resisting input noise. Our branch SGM gives scan-line ability to actively search and locate feature region in 2-dimensional spatial domain. Also, we built a scalable and highly parallel stereo matching core using Spartan6 platform to examine the difference between SGM and our method in terms of hardware utilization and maximum frequency/frame rate and average error from Middlebury stereo evaluation.

RELATED WORK

Semi-Global Matching algorithm is based on multiple one-dimensional dynamic programming in image matching, and simulate global matching optimization of two-dimensional image. This method has low calculation complexity, good performance in hardware design, and robust environmental tolerance. Last but not the least, semi-global matching algorithm has similar accuracy as global matching. [1][2]

C. Banz and S. Hesselbarth [3] utilized non-parametric rank transform and semi global matching algorithm to realize the disparity calculation and disparity map display. The hardware equipment was Xilinx Virtex-5 FPGA, and the disparity map was 640 *480 resolutions at a frame rate of 30 frames per second for real-time condition. The maximum disparity range was up to 128 under the clock frequency at least 12 MHz. It could meet the demands of real world applications for disparity resolution, frame rate, and the use of resource. In many applications, especially for automotive sectors, Semi-global matching was in good use. [4] In addition, adding rank transform to semi-global matching has been presented to be insensitive to noise and had small effect on exposure problem and lightning problem [5]. The advantages were obvious for SGM that it not only benefitted to local neighborhood, but also showed an optimization across the whole image. Implementing SGM on SIMD-CPUs and GPUs achieved frame rates of 1.4 fps and 13 fps for QVGA images, respectively [6]. It could reach to 30 fps for VGA images (640*480 resolution) for a specialized VLIW-ASIP with architectural adaptations [7].

S. Hermann [8] proposed an improved algorithm called fSGM which embedded the scan-line dynamic programming method into a hierarchical scheme. It could handle large pixel displacements with an accurate method.

R. Rzeszutek and D. Tian's [9] estimation system was based on the semi-global matching algorithm. They used a different cost-aggregation strategy to minimize the number of paths that needed to be integrated. Their improvement was able to handle misaligned images to any disparity estimation system, not only for SGM one. They also presented a disparity estimation approach that was able to acquire disparity maps for stereo images with non-rectified or misaligned cameras. They did this by proposing a modification in the matching stage.

Semi-Global Matching

Outline

Semi-global matching provides very good accuracy and low computation complexity. In this design, we utilized 4 directional SGM method to implement the design. SGM can detect different object size without extra setting or compromising. Also, comparing to adaptive local matching, SGM is a fairly static algorithm, which means the whole process has less random factor and is suitable for parallel computing especially on low-cost FPGA system. SGM utilizes dynamic programming and smoothness constraint to achieve pixel-wise matching.

Pixel-wise cost Calculation

The pixel-wise matching cost calculation is usually based on a simple similarity measurement.

$$C(p,d)=\left|I_1(p)-I_2(p-d)\right| \tag{1}$$

The absolute difference (AD) is a very easy and practical way to measure the difference between two pixel intensities. In real world implementations, however, it is not always a good way to find the pixel-wise similarity. Different camera gives different pixel intensities. It is hard to find the exact same object in left and right images.

Census Transform

Adding census transform to SGM can produce good result on generating disparity map. It is based on relative intensity rather than absolute pixel intensity. Within a window size, every pixel compares to the center pixel. If the surrounding pixel intensity is higher, it will become 0, if it is lower, it becomes 1. Then we use the hamming distance to measure the matching cost. Since we have left and right images, for each pixel comparison, we compared two census windows. For each pair of corresponding census vectors, make exclusive or operation and add them together. The adding operation is using an adder tree.

Core Equation

Semi-Global Matching applies one-dimensional matching strategy to optimize two-dimensional matching. It follows scan line based dynamic programming structure. The core equation for SGM can be seen as follows:

$$\begin{aligned} L_r(p,d) = C(p,d) + \min(& \\ & L_r(p-1,d), \\ & L_r(p-1,d-1)+P1, \\ & L_r(p-1,d+1)+P1, \\ & \min_k(\,L_r(p-1,k))+P2\,) - \min_k(\,L_r(p-1,k)) \end{aligned} \tag{2}$$

$$\begin{aligned} S(\mathrm{p},\mathrm{d}) &= \sum_r L_r(\mathrm{p},\mathrm{d}) \\ disparity(\mathrm{p}) &= \arg\min_d(S(\mathrm{p},\mathrm{d})) \end{aligned} \tag{3}$$

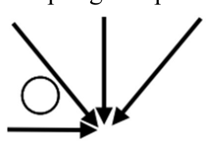
In this equation, for each pixel p and disparity d, the path cost L is computed by the sum of the matching cost C and the minimum path cost of the previous pixel along the direction r, considering two penalties P1 and P2. In this design, we have four searching directions, so we have four path r. Those four directions are 45°, 90°, 135°, and 180°, respectively. Their path cost information from the previous pixels.

The minimum path cost is computed by the minimum over four values. The first value is the path cost of the previous pixel of the same disparity. The second and third value are the path cost of the previous pixel with the next lower and higher disparity, small penalty P1 is taking into consideration. The last value is the minimum path cost of the previous pixel over all disparities with the additional higher penalty P2. We also need to subtract the minimum path cost to have a low path cost L to save the bit width. This subtraction will never make the whole equation negative.

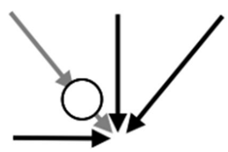
Next, sum the cost L in all directions r. Then, search the disparity d with the minimum summed cost for each image pixel p.

Some issues about SGM

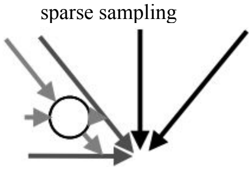
SGM method uses several 1D beams to approach global matching method. Each beam can only propagate cost information to it's own direction. Also each direction cannot share the cost information. This property performs very simple computation but it also causes two main drawbacks, sparse sampling and private information.



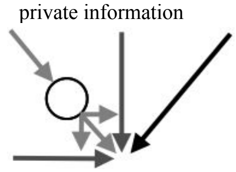
sparse sampling



private information



additional path passing feature information



Path share the feature information

Sparse sampling

The sparse sampling problem is pretty straightforward. SGM uses several 1D scanline to simulate 2D global matching. If we consider each scanline is actually to sample an area uses radial beams. Cost aggregation will collect those sampling information and calculate the matching cost based on sample information. The problem is, when scanline goes further from p, the sample density would be even lower (see Fig 1). The easiest way is to use more scanline to overcome the sparse problem, but this way would need more computation and memory space. It's not an ideal way to do in real-time application system. Another issue is the sparse sampling might only cover a very small portion of the feature information, however, scanline still need to accept noise information of each pixel. That causes weak signal and strong noise.

Private information

The private information is caused by the SGM path characteristic. The equation(1) shows the cost information Lr can only propagate to following L_r variable. The whole information can only propagate on a straight path with specific direction. The problem is, even if there is a very important feature nearby the scanline, the information would still not be carried. Because each scanline is totally independent to each other, important information cannot be propagated as far as it can.

The disparity value continuity of SGM relies on smoothness constrain and spatial domain continuity. The spatial domain continuity could be corrupted by noise easily, because each scanline runs on it's path and different path has totally independent noise on it's path. And because of private information, a path with important feature information could be ignored by other majority paths that don't sense the exist of the feature (see Fig 2).

BRANCH SEMI-GLOBAL MATCHING

Approaches to solving the issues

To solve sparse sampling and private information issues we are trying to find out the approach for each other and bind both solution and provide a unified method. Before we discuss more information about approaches we want to modify the original equation first to accept out modification plugin.

$$\begin{aligned} L'_r(p,d) = C(p,d) + \min(& \\ & L_r(p-1,d), \\ & L_r(p-1,d-1) + P1, \\ & L_r(p-1,d+1) + P1, \\ & \min_k(L_r(p-1,k)) + P2) - \min_k(L_r(p-1,k)) \end{aligned} \tag{m}$$

$$L_r(p,d) = \alpha L'_r(p,d) + (1-\alpha) Lb_r(p,d) \quad \textit{where} \;\; 0 \le \alpha \le 1 \tag{bri}$$

We modified the output of the equation above from L_r to L'_r. It means we create an temporary variable L'_r. The middle variable allows us to attach extra information and output matching cost L_r. The extra process is performed in the equation (bri) and we need to find out Lb_r to fix the issues that we just mentioned, sparse sampling and private information. Lb_r will contain two main information, missed feature and adjacent path information.

$$Lb_r(p,d) = (LS_r(p,d) + LP_r(p,d))/2 \tag{lbr}$$

LS_r:	path cost with missed feature information
LP_r:	path cost with adjacent path information

Firstly, to solve sparse sampling issue, one way is to create/find additional path and join the information into matching cost of current direction. To avoid adding too much noise on the cost information and reduce the usage of resources, the additional paths must across at least a feature location, and the covered feature could be as close to point p as possible. Also, if there are multiple feature position, near feature information should have higher weight than far features, so that near feature could obtain more importance in order to decide disparity value. Fortunately, according to equation (bri) the latest Lb_r would have highest weight and when every time (bri) was performed the old Lb_r would reduce weight again by α coefficient.

Another problem is how could we decide which point has a important feature and how to create a path to cover the feature information?

A good feature usually means a region with rich texture, and rich texture means a strong edge and corner. Any slight disparity mismatch would produce lots of difference or cost in *C(p,d)*. The pixel-wise cost information *C(p,d)* adds into L'_r by equation (m). Since L'_r carries matching cost information, a path with high L'_r is a good indication that the path has covered an important feature before.

$$LS_r(p,d) = \max_r(L'_r(p,d)) \tag{lsr}$$

Equation (lsr) is to select the largest matching cost from all the different direction r. LS_r will be added into the main path cost L_r by (bri).

Secondly, private information could be solved by blending adjacent (r+1 and r-1 path) information.

$$LP_r(p,d) = (L'_{r+1}(p,d) + L'_r(p,d) + L'_{r-1}(p,d))/3 \tag{lpr}$$

With equation (lsr) and (lpr), Lb_r has high cost information and adjacent path information. However, equation (lsr) considered to many directions, and equation (lpr) blends adjacent information blindly without selection. Next step is to refine and combine (lsr) and (lpr) into only one simple equation. And it turns out (lsr) actually provides information sharing ability and feature information selection all we need to do is to reduce the range of selection.

$$\begin{aligned} &LS_r(p,d) = \\ &LP_r(p,d) = \\ &Lb_r(p,d) = \max(L'_{r-1}(p,d), L'_r(p,d), L'_{r+1}(p,d)) \end{aligned} \tag{lsr2}$$

We reduce the (lsr) selection scope and let (lpr) have ability to select feature information. It turns out LS_r, LP_r and Lb_r becomes totally the same thing as equation(lsr2).

$$\begin{aligned} L_r(p,d) \ &= \alpha L'_r(p,d) + \\ &(1-\alpha)\max(L'_{r-1}(p,d),\ L'_r(p,d), L'_{r+1}(p,d)) \\ where \quad &0 \le \alpha \le 1 \end{aligned} \tag{bri2}$$

The bridge equation is formed as (bei2). The equation provides a simple solution for sparse sampling and private information issues.

Branch SGM

Our branch SGM provides an approach to solve SGM drawbacks and make the scanline have ability to actively search and collect feature information. The extra feature information gives more signal strength to the matching cost to overcome the noise.

Our complete branch SGM equation as following

$$\begin{aligned} L'_r(p,d) = C(p,d) + \min(& \\ & L_r(p-1,d), \\ & L_r(p-1,d-1) + P1, \\ & L_r(p-1,d+1) + P1, \\ & \min_k(\ L_r(p-1,k)) + P2 \\ &) - \min_k(\ L_r(p-1,k)) \end{aligned} \tag{m2}$$

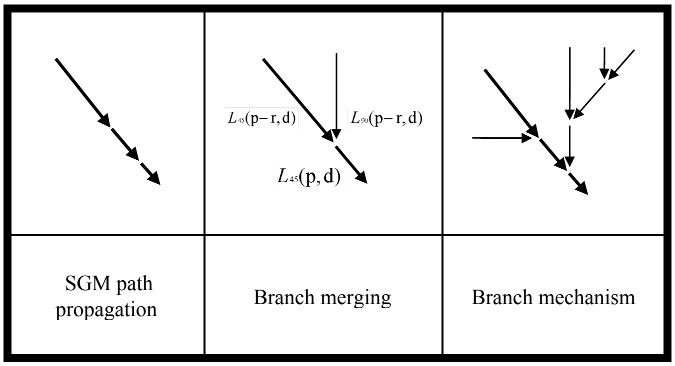
$$\begin{aligned} &S(p,d) = \sum_r L'_r(p,d) \\ &disparity = \arg\min_d S(p,d) \end{aligned} \tag{agg}$$

$$\begin{aligned} L_r(p,d) \ &= \alpha L'_r(p,d) + \\ &(1-\alpha)\max(L'_{r-1}(p,d),\ L'_r(p,d), L'_{r+1}(p,d)) \\ where \quad &0 \le \alpha \le 1 \end{aligned} \tag{bri3}$$

The (m2) and (agg) are basically the same as SGM equation. The only different is we create an middle variable $L'_r(p,d)$. The Eq2 is set to be a bridge between $L_r(p,d)$ and $L'_r(p,d)$. The main idea of (bri3) is to let each path notice the highest path cost information which besides itself. High matching cost usually means an important feature presents on the path. The branch mechanism will broadcast this information.

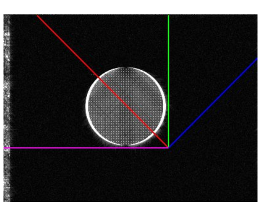
If you look into the mechanism and trace back where the information comes from, the behavior would actually like a main path stretches lots of branches. Each branch will try to cover the important feature and collect the feature data.

For instance, our main direction is r=45°. Equation (bri3) will check paths cost from three direction $L'_0(p,d)$, $L'_{45}(p,d)$ and $L'_{90}(p,d)$ then find out which one has the largest cost. Let's say after the comparison we find out $L'_{90}(p,d)$ is the biggest one, then $L_{45}(p,d)$ is just a simple combination of $L'_{45}(p,d)$ and $(1-\ )L'_{90}(p,d)$. Since $L'_{45}(p,d)$ and $L'_{90}(p,d)$ is also the combination of other paths, $L_{45}(p,d)$ is not only the combination of two direction paths but the combination of several subbranches.

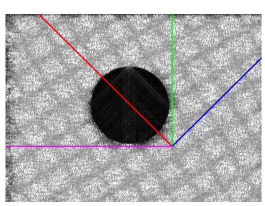


The parameter in Branch SGM

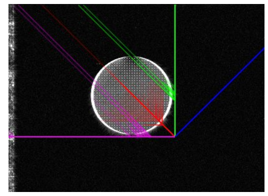
There is a parameter *(1-)* to control how much portion of the path cost with feature information should merge into the current path. If you change the point of view, and trace back to the origin of current path cost, you can actually consider *(1-)* as a branch energy that split out from stem. When the value gets smaller, branches will get more energy from stem to perform feature searching and cost collecting. When equal to 1, the branches get 0 energy and the behavior will be exactly the same as SGM method. To visualize the branch behavior there is a simple disparity pair as a example (Fig XX1 and Fig XX2). The disparity pair consist of two objects background and foreground. The disparity value of background is db, and the disparity value of foreground is df. In the following tests I uses 4 directional scanline to perform the branch SGM method.



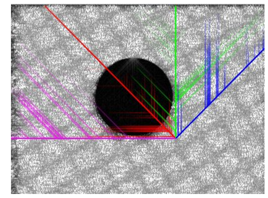
α=1,d=db



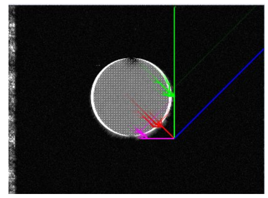
α=1,d=df



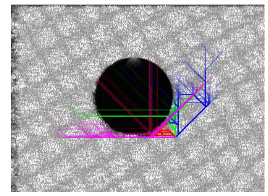
α=0.95,d=db



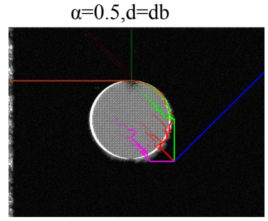
α=0.95,d=df



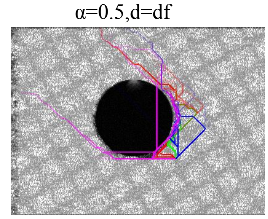
α=0.5,d=db



α=0.5,d=df



α=0.2,d=db

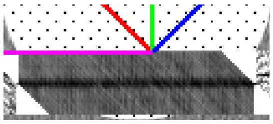


α=0.2,d=df

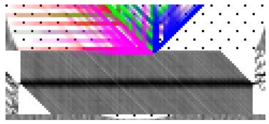
The Fig(8 and 9) shows the disparity pair subtraction under different disparity values. When disparity value equal to db, background is perfectly matched so there is almost no difference been produced but the foreground part has lots of pixelwise costs. When disparity value equal to df, foreground has very little error, but background is full of error. When α equals to 1, the branch SGM loses all branch characteristics and back to the SGM method. As you can see, the Fig 8 and Fig 9 uses 4 straight paths to do the semi-global matching. When the α value set to be 0.95, branches gets small portion of energy (0.05) to reach out from original path and collect feature data. If you take a look Fig 10 carefully, each branch cover the white dots, which are pretty important feature location.

When ɑ is 0.5 (Fig 12 and Fig 13), branches have half energy from the stem. Although, branches get more energy, main stem loses those energy as well. You can see the travel distance doesn't too far. When ɑ becomes even lower (Fig 14 and Fig 15), branches takes almost all the energy from stem. In this state, branch has more freedom to perform cost collecting and searching. In Fig 14, the branches stick on the high cost region especially on the arc of the circle. And some paths craw over the arc and leave form the top of the arc. In the Fig 15, the branches try to evade the black circle since there is almost no cost in the circle. Branches actually by pass the circle and collect cost data as much as it can.

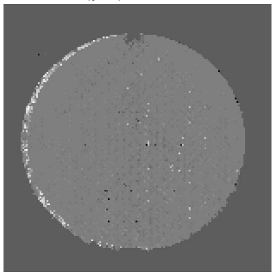
The following result uses P1=12, P2=53 and Gaussian noise variance 0.001 for both left and right image. The figure (16 and 17) shows part of cost aggregation branches and S(p,d) plot. The result shows branch SGM method provides clearer S(p,d) plot and make low matching cost tunnel even darker.



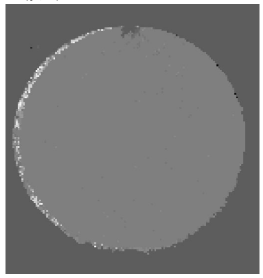
The S(p,d) of SGM



The S(p,d) of branch SGM α=0.8



The noisy input causes SGM output error



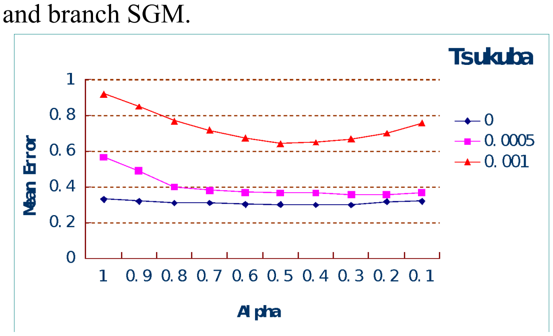
The branch SGM provides low noise and sharp output edge.

EXPERIMENTAL RESULT

In this section, we analyze the disparity map average error under different noise level and value, also we compare the hardware requirements between SGM and branch SGM.

Algorithm performance comparison

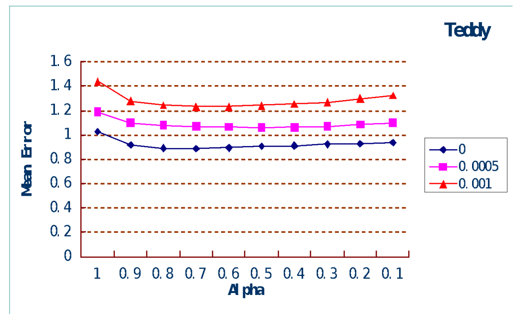
The algorithm performance comparison part we use four standard disparity pair sets Tsukuba, Teddy, Cone, and Venus to find out the performance improvement between SGM and branch SGM.



The graph about Tsukuba test pair

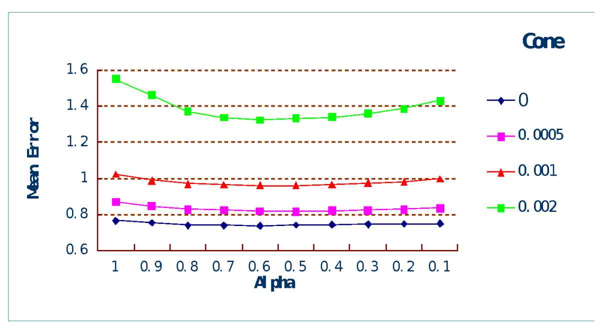
The Tsukuba disparity pair test uses P1=12 and P2=20. This parameter set could produce well enough average error without damaging the edge sharpness. The disparity value is from 0 to 19.

When there is no noise in the test disparity pair, the average error reduces 6%. And notice the improvement will be even better along with higher Gaussian noise variance. When variance equals to 0.0005, the average error reduces 33%. This noise level is pretty common for real video image pair noise. And when noise gets even worse, branch SGM algorithm can also keep good performance and noise reduction ability. When variance equals to 0.001, the average error reduces 21%.



The graph about Teddy test pair

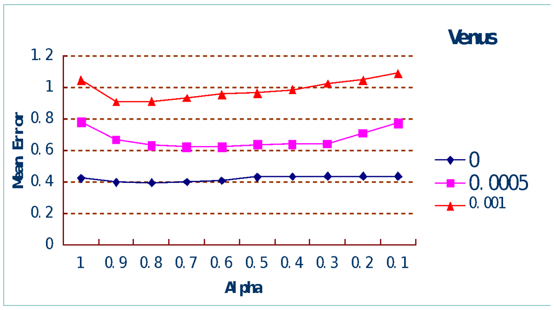
This set uses Teddy disparity pair. P1=9 and P2=36. The disparity value is from 0 to 59. Like Tsukuba test set, our branch algorithm has better noise reduction ability than SGM method. In this test set, our branch SGM has pretty stable improvement. The average error reduces 21% over three noise conditions. The reason for this is because the resolution is big enough and there are a lot of pixel intensity changes. In this case, because the feature is so strong so the additional noise is not able to affect the result too much. However, some area with weaker feature, like paint or the roof of the house, noise will dominate the SGM calculation. Branch SGM can still handle those areas and provide good disparity quality.



The graph about Cone test pair

This set uses Cone disparity pair. P1=9 and P2=36. The disparity value is from 0 to 59. In this set, we have very little improvement at the variance between 0 to 0.001. The average error reduces 3.4% over three noise conditions. Because the Cone test set has lot of features, cone, mask, wood, fabric. Those features are very strong, and light noise is not able to affect those strong information. When the noise variance goes up to 0.002, the noise start to affect the

SGM result. Lot of bad pixel makes it look like salt-and-paper noise. Our result reduces 13% average errors and remove a lot of bad pixels compare to the SGM result.

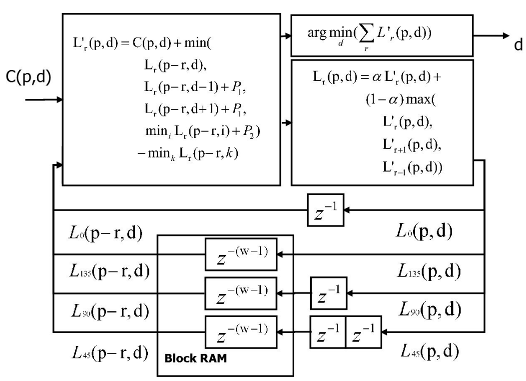


The graph about Venus test pair

The last one is Venus. In this test set we uses P1=9 and P2=36. The disparity value is from 0 to 19. Branch SGM did improvement under noisy situation, around 13% error reduction.

Hardware requirement comparison

The hardware implementation is performed on Atlys™ Spartan-6 FPGA Development Board. The hardware requirements are focus on register slices, Logic elements, block RAM usage and maximum operation frequency. The video is VGA resolution(640 x 480) and the disparity range is 32.



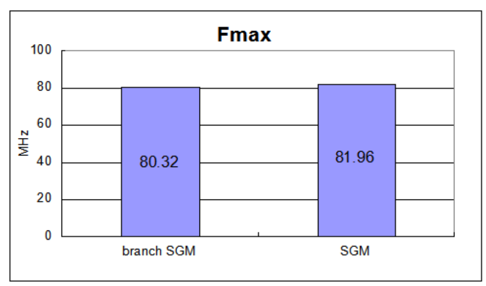
Hardware architecture of branch SGM core

The hardware implementation of branch SGM is almost the same as SGM implementation. Just have an additional block for branch searching and blending. The additional block uses 6% more slices and 10% more LUTs. The branch SGM and SGM uses totally the same amount of block ram.

Hardware requirement Comparison Table

	register	LUTs	Block RAM (18Kb)
SGM	6519	10625	130
Branch SGM	6907	11748	130

The maximum operation frequency was shown as Fig 25. The Fmax of branch SGM is 2% less (80.32MHz) than SGM method.



The Fmax comparison between branch SGM and SGM

Conclusion

In this paper, we proposed a novel method to improve SGM algorithm. The method we called branch SGM. Branch SGM added an additional equation to select and blend neighboring path information. Branch SGM gives ability for each path to actively locate feature and collect feature information. The branch mechanism is able to let each path gather more meaningful information (signal) than SGM does. Since the noise level still remain the same, higher signal energy leads higher SNR.

As result, branch SGM presents higher noise reduction ability, in the same time the algorithm still very friendly for hardware implementation. In our test sets, branch SGM reduced 7% to 25% average error in four test sets(Tsukuba, Teddy, Corn, Venus). In the test result, because our method provides higher SNR, the improvement will actually goes higher alone with noise level.

In the hardware implementation, we used Spartan-6 FPGA and VMOD-CAM to realize real-time branch SGM based stereo vision engine. The disparity image was in VGA resolution and the disparity range was up to 32 under real-time conditions.

The SGM implementation used 6519 slices registers, 10625 LUTs, and the maximum pixel clock frequency reached to 83 MHz. In the other hand, branch SGM implementation used 6907 slices registers, 11748LUTs, and the maximum pixel clock frequency reached to 83 MHz. Our branch SGM uses 6% more slices and 10% more LUTs.

Although, branch SGM provides a new perspective of dynamic feature searching, the equation might not be the most efficient way to do feature searching. In the future, we will apply entropy field to guide branches toward the nearest minimum entropy location to obtain the maximum possible feature information.

References

H. Hirschmuller, “Accurate and Efficient Stereo Processing by Semi-Global Matching and Mutual Information,” in IEEE Conference of Computer Vision and Pattern Recognition, 2005, vol. 2, 807-814.

H. Hirschmuller, “Stereo Processing by Semi-Global Matching and Mutual Information,” in IEEE Transactions of Pattern Analysis and Machine Intelligence, 2008, vol. 30, 328-341.

C. Banz; S. Hesselbarth; H. Flatt; H. Blume; P. Pirsch, “Real-Time Stereo Vision System using Semi-Global Matching Disparity Estimation: Architecture and FPGA- Implementation”, in Institute of Microelectronic Systems, Leibniz Universit¨at Hannover, Hannover (Germany).

S. Gehrig; F. Eberli; T. Meyer, “A real-time low-power stereo vision engine using semi-global matching,” in International Conference of Computer Vision Systems. Springer, 2009, pp. 134–143.

H. Hirschm¨uller and D. Scharstein, “Evaluation of Cost Functions for Stereo Matching,” in IEEE Computer Vision and Pattern Recognition Conference, pp. 1–8, 2007..

J. Gibson and O. Marques, “Stereo Depth with a Unified Architecture GPU,” in IEEE Computer Vision and Pattern Recognition Conference, 2008, pp. 1–6.

G. Pay´a-Vay´a; J. Mart´ın-Langerwerf; C. Banz, F. Giesemann; P. Pirsch,; H. Blume, “VLIW Architecture Optimization for an Efficient Computation of Stereoscopic Video Applications,” in IEEE Green Circuits and Systems International Conference, 2010

S. Hermann and R. Klette, “Hierarchical Scan Line Dynamic Programming for Optical Flow using Semi-Global Matching”, in The .enpeda.. Project, Department of Computer Science, The University of Auckland, NewZealand

R. Rzeszutek1; D. Tian2; A. Vetro2, “Disparity Estimation of Misaligned Images in a Scanline Optimization Framework” 1 in Department of Electrical and Computer Engineering, Ryerson University, Toronto, Ontario, Canada; 2 in Mitsubishi Electric Research Laboratory, Cambridge, Massachusetts, USA