A robust cost function for stereo matching of road scenes

Alina Miron, Samia Ainouz, Alexandrina Rogozan, Abdelaziz Bensrhair

A robust cost function for stereo matching of road scenes

Alina Miron[a,b,], Samia Ainouz[a], Alexandrina Rogozan[a], Abdelaziz Bensrhair[a]

[a] *INSA Rouen/LITIS laboratory - EA4108, 76801, Saint-Etienne du Rouvray, France.*
[b] *Babes-Bolyai University, Cluj-Napoca, Romania*

Abstract

In this paper different matching cost functions used for stereo matching are evaluated in the context of intelligent vehicles applications. Classical costs are considered, like: sum of squared differences, normalized cross correlation or census transform that were already evaluated in previous studies, together with some recent functions that try to enhance the discriminative power of **C**ensus **T**ransform (CT). These are evaluated with two different stereo matching algorithms: a global method based on graph cuts and a fast local one based on cross aggregation regions. Furthermore we propose a new cost function that combines the CT and alternatively a variant of CT called **C**ross-**C**omparison **C**ensus (CCC), with the mean sum of relative pixel intensity differences (DIFFCensus). Among all the tested cost functions, under the same constraints, the proposed DIFFCensus produces the lower error rate on the KITTI road scenes dataset[1] with both global and local stereo matching algorithms.

Keywords: stereo vision, census transform, cross comparison census, graph cuts, matching cost comparison

1. Introduction

Stereo matching has been an intensely studied topic in research due to its crucial applications that vary from 3D reconstruction to image-based rendering or object hypothesis generation.

Our field of application is intelligent vehicles, in particular the detection of road obstacles like pedestrians. The objective is to reduce the hypothesis space using the information provided by the disparity map. Classic techniques like sliding window produce an extensive search space while ground subtraction based techniques can not be applied to dynamic scenes. Robust disparity map is consequently essential in order to have good hypothesis over the location of pedestrians.

Most of the stereo matching algorithms rely on four important steps: Cost computation; Cost aggregation; Disparity computation/optimisation and Disparity refinement [19]. Each step is important for the quality of the disparity map, with the cost computation step being crucial as it stands at the basis of the stereo matching algorithms. A given cost function can be minimised using different methods within the step of *disparity computation/optimisation*. There exists many techniques for energy minimisation that vary from local methods that find the minimum of the cost function using a winner takes it all strategy like in [24] and [14], to global techniques like graph cuts [11], dynamic programming [2], or belief propagation [3], [9]. Authors in [21] and [10] compared different optimisation algorithms based on energy functions and showed that the lowest energy is produced by the graph cuts.

Choosing a cost function has to take into account the radiometric distortions, since in real traffic situations these are very pronounced. Some of the causes are sun flares, reflections or just camera sensor differences. In this context, our contribution is twofold.

- First, we compare different cost functions in order to be able to choose the most adapted one for our field of application. For this, we combine different cost functions with two stereo matching methods: a global technique based on Graph Cuts [11] and a local stereo matching algorithm based on cross zones aggregation with local voting [24].
- Secondly, we propose a new cost function that is robust to radiometric distortions.

2. Related works

Choosing the right cost function is paramount for having a good disparity map. As presented in [8], the costs can be divided into parametric functions, where the cost incorporates the magnitude of pixel intensity, and non-parametric ones. Common parametric costs include those based on absolute differences and square differences, along with the window-based approaches: sum of absolute differences (SAD) and sum of squared differences (SSD) [1],

Email addresses: alina.miron@insa-rouen.fr (Alina Miron), samia.ainouz@insa-rouen.fr (Samia Ainouz), alexandrina.rogozan@insa-rouen.fr (Alexandrina Rogozan), abdelaziz.bensrhair@insa-rouen.fr (Abdelaziz Bensrhair)

[1] http://www.cvlibs.net/datasets/kitti

normalized cross-correlation (NCC), zero-mean based costs (like ZSAD, ZSSD and ZNCC), or costs computed on the first (gradient) or second (laplacian of gaussian) image derivatives. Non-parametric costs include the popular Census and Rank methods [23].

There exists several studies where comparison of cost functions is performed, the most extended ones being made in [7] and [8]. In comparison with [7], where six cost functions where tested, in [8], authors compared fifteen different stereo matching costs in relation with images affected by radiometric differences. These costs are compared using three different stereo matching algorithms: one based on global energy optimisation (Graph Cuts), one using semi-global matching [5] and a local window-based algorithm. They conclude that the cost based on CT gives the best overall performance. In comparison with [8] that use both simulated and real radiometric changes in a laboratory environment, in this paper the experiments are performed on real road images from the KITTI dataset [4] which presents significant radiometric differences. Besides the cost functions that provided the best results in [8], we also test some recent functions based on CT that gave good results on the Middlebury dataset[1]. Moreover we propose a new cost function $C_{DiffCensus}$ that remains robust to radiometric changes. These costs will be presented in the following two sections.

3. State of the art matching costs

In this section we define each matching cost function used in the experiments. Along with our new proposed function, eight different cost functions will be compared: squared intensity differences (C_{SD}), zero-mean normalized cross-correlation (C_{ZNCC}) [6], [8], census tranform C_{CT} [23], cross comparison census C_{CCC} [16], a function combining the sum of absolute differences with gradient (C_{klaus}) [9], a function combining absolute differences with census transform ($C_{ADCensus}$)[14] and one that combines absolute differences computed both on visible and gradient with census tranform computed on gradient (C_{cstent}) [20].

The functions C_{SD}, C_{ZNCC} and C_{CT} were already compared in [8] on the Middlebury dataset composed of images with simulated or real radiometric distortions. We have chosen these functions as reference.

In the following, the functions presented are grouped into costs based on differences of intensities and costs based on CT.

3.1. Intensity differences based costs

$\boldsymbol{C_{AD}}$, $\boldsymbol{C_{SD}}$ & $\boldsymbol{C_{SAD}}$. One of the most popular cost matching function is the *squared intensity differences (SD)* (see equation 2) like used in [11] or *absolute intensity differences (AD)* (see equation 1) like used in [14],[9].

Let p be a pixel in the left image with coordinates (x, y) and d the disparity value for which the cost of p is computed. Let $I_l(x, y)_i$ be the intensity value of pixel p in the left image on color channel i, while $I_r(x, y-d)_i$ is the intensity value of the pixel given by coordinates $(x, y-d)$ in the right image. We consider n the number of color channels ($n = 1$ for gray scale images and $n = 3$ for color images).

$$C_{AD}(x,y,d) = \frac{1}{n}\sum_{i=\overline{1,n}} |I_l(x,y)_i - I_r(x,y-d)_i| \qquad (1)$$

$$C_{SD}(x,y,d) = \frac{1}{n}\sum_{i=\overline{1,n}} (I_l(x,y)_i - I_r(x,y-d)_i)^2; \qquad (2)$$

If we consider $N(x, y)$ to be the neighbourhood of the pixel with coordinates (x, y), then the cost AD on this neighbourhood is defined like in equation 3.

$$C_{SAD}(x,y,d) = \sum_{(a,b)\in N(x,y)} C_{AD}(a,b,d) \qquad (3)$$

$\boldsymbol{C_{ZNCC}}$. Zero-mean normalized cross correlation (eq. 4 is a parametric window based matching function that provided one of the best results in the study [21] in presence of radiometric distortions.

$$C_{ZNCC}(x,y,d) = 1 - ZNCC(x,y,d) \qquad (4)$$

where

$$ZNCC(x,y,d) = \frac{\sum\limits_{(a,b)\in N_{(x,y)}} ZV(I_l,a,b)ZV(I_r,a,b-d)}{\sqrt{\sum\limits_{(a,b)\in N_{(x,y)}} (ZV(I_l,a,b))^2 \sum\limits_{(a,b)\in N_{(x,y)}} (ZV(I_r,a,b-d))^2}} \qquad (5)$$

and

$$ZV(I,x,y) = I(x,y) - \overline{I}_{N(x,y)}(x,y), \qquad (6)$$

where $\overline{I}_{N(x,y)}$ is the mean value computed in the neighbourhood $N(x, y)$.

$\boldsymbol{C_{klaus}}$. There exists several variations based on the costs previously described[2]. One of the top three algorithms on the Middlebury dataset [9] proposes the combination of C_{SAD} with a gradient based measure C_{GRAD} (see equation 7). Both costs are computed in a neighbourhood $N(x, y)$ of 3×3 pixels and are weighted by w, which is computed by a grid search.

[1] http://vision.middlebury.edu/stereo/

[2] If the authors did not name the proposed cost functions we are going to use the first name on the article to name the cost

$$C_{klaus}(x,y,d) = (1-w)*C_{SAD}(x,y,d)+w*C_{GRAD}(x,y,d) \tag{7}$$

where

$$C_{GRAD}(x,y,d) = \sum_{(a,b)\in N(x,y)} |\Delta_x I_l(a,b) - \Delta_x I_r(i,j-d)| + \sum_{(a,b)\in N(x,y)} |\Delta_y I_l(a,b) - \Delta_y I_r(i,j-d)|, \tag{8}$$

where Δ_x and Δ_y are the horizontal and vertical gradients of the image.

3.2. CT based cost functions

$\boldsymbol{C_{CT}}$**.** As demonstrated in [8], the **C**ensus **T**ransform (CT) [23] is one of the most robust cost function to radiometric changes. CT will basically replace all the intensity of pixels with a bitstring obtained by comparing the intensity of each pixel with the intensities of pixels in its vicinity. The *CT* cost is given by the Hamming distance (D_H) between two bit strings (equation9).

$$C_{CT}(x,y,d) = D_H(CT(x,y), CT(x,y-d)), \tag{9}$$

where *CT* is the bit string build like in eq. 10.

$$CT(u,v) = \otimes_{\substack{i=\overline{1,n}\\ j=\overline{1,m}}} (\xi(I(u,v), I(u+i, v+j))), \tag{10}$$

where $n \times m$ is the census support window, $\otimes$ denotes a bitwise concatenation, and ξ function is defined in equation 11.

$$\xi(p_1,p_2) = \begin{cases} 1 & p_1 \le p_2 \\ 0 & p_1 > p_2 \end{cases} \tag{11}$$

$\boldsymbol{C_{CCC}}$**.** **C**ross **C**omparison **C**ensus (CCC)[16] (eq. 13) is a variant of CT. While standard CT (eq. 10) explores only the comparisons of the central pixel with its vicinity, the bit string for CCC is obtained by comparing each pixel in the considered window with those in its immediate vicinity in a clockwise direction. The cost of CCC is also given by the Hamming distance between two bit strings (equation 12).

$$C_{CCC}(x,y,d) = D_H(CCC(x,y), CCC(x,y-d)) \tag{12}$$

where

$$CCC(u,v) = \otimes_{\substack{i=\overline{0:step:n}\\ j=\overline{0:step:m}}} (\xi(I(i,j), N_{CCC}(i,j,step)) \tag{13}$$

where the neighbourhood N_{CCC} is given by eq. 14.

$$N_{CCC}(i,j,step) = \{(i, j+step); (i+step, j+step); (i+step, j); (i+step, j-step)\} \tag{14}$$

step is an empirically chosen value in order to *skip* some pixels in the support window, $(j + step) < m$ and $(i + step) < n$ and $(j - step) >= 0$

As shown in [16], CCC can be computed in a very fast way. In the first step we compute for each pixel in the image a four-bit string value obtained by comparison with the four pixels in its immediate vicinity. The final CCC for a pixel is obtained by the concatenation of these bit substrings of the relevant pixels in the window of the considered pixel. If CT with a window of 7 × 9 pixels takes around 3 seconds[3] to be computed on a image of 1241 × 376 pixels, CCC with the same window size needs only 0.6 seconds.

$\boldsymbol{C_{ADCensus}}$ **&** $\boldsymbol{C_{cstent}}$**.**

CT based functions became popular due to the good results obtained on the Middlebury dataset.

In [14] a combination between the CT and AD is used (eq. 15), that on Middlebury dataset reduces the error in non-occluded areas in average with 1.3%.

$$C_{ADcensus}(x,y,d) = \rho(C_{CT}(x,y,d), \lambda_{census}) + \rho(C_{AD}(x,y,d), \lambda_{AD}) \tag{15}$$

where λ_{census} and λ_{AD} control the influence of each cost.

Another combination of a CT and AD (eq. 16) where both are computed on the gradient images is proposed in [20]. It was shown that this function can reduce the erroneous pixels on Middlebury dataset with up to 2.5%.

$$C_{cstent}(x,y,d) = \rho(C_{\Delta census}(x,y,d), \lambda_{census}) + \rho(C_{AD}(x,y,d), \lambda_{AD}) + \rho(C_{\Delta AD}(x,y,d), \lambda_{\Delta AD}), \tag{16}$$

where $\Delta census$ and ΔAD are the CT cost, respectively the AD cost, computed on gradient images; λ_{census}, λ_{AD} and $\lambda_{\Delta AD}$ are parameters controlling the influence of each cost.

4. The proposed matching cost: $\boldsymbol{C_{DIFFCensus}}$

We propose a new function that combines the CT [23], or its variant CCC[16], with the mean sum of relative differences of intensities inside a window (eq. 17). We consider CCC separately from CT due to its fast computation time. In comparison with functions like $C_{ADCensus}$ or C_{cstent} that use the pixel intensities values, the $C_{DIFFCensus}$ does not rely on the value of the pixel intensity but on the difference

[3] Single threaded a machine with 2.4 GHz Intel Core 2 Duo

of intensity between a considered pixel and its neighbourhood. This keeps the function as a non-parametric one while incorporating extra information.

$$C_{DIFFCensus}(x, y, d) = \rho(C_{census}(x, y, d), \lambda_{census}) + \rho(C_{DIFF}(x, y, d), \lambda_{DIFF}) \tag{17}$$

where C_{census} can be either C_{CT}, which will give C_{DIFFCT}, or C_{CCC}, which will give $C_{DIFFCCC}$; C_{DIFF} defined in eq. 18.

$$C_{DIFF}(x, y, d) = |\overline{DIFF}(x, y) - \overline{DIFF}(x, y - d)| \tag{18}$$

where $n \times m$ is the same support window that is used to compute the CT.

$$\overline{DIFF}(u, v) = \frac{DIFF(u, v)}{CensusSize} \tag{19}$$

where *CensusSize* is the size of the bit string given by the support window $n \times m$ and *step* which is chosen like in eq. 14.

$$DIFF(u, v) = \sum_{\substack{i=\overline{1:step:n} \\ j=\overline{1:step:m}}} (|I(u, v) - I(u + i, v + j)|), \tag{20}$$

Choosing the appropriate cost function depends on the stereo matching algorithm. In order to test the proposed function we are going to compare it with the other eight matching costs functions with two different stereo matching algorithms: a global one based on graph cuts and a local one based on cross zone aggregation, that will be presented in the next section.

5. Stereo matching algorithms

5.1. Graph cuts

As described in [11], a graph cut is a partition of a graph with two distinguished terminals called source (s) and sink (t) into two sets V^s and V^t, such that $s \in V^s$ and $t \in V^t$. The cost of the cut is represented by the sum of the edges' weights between the two partitions. Finding the minimum cut, and implicitly the minimum cost, can be resolved by computing a maximum flow between terminals. In practice the global energy minimisation technique using graph cuts has been shown to be effective with the condition of having an appropriate cost function.

For the cost comparison, the energy function is used as described in [11]. The purpose is to find a disparity function f that minimizes a global energy $E(f)$ as seen in equation 21. The occlusion term E_{occ} imposes a penalty for occluded pixels, while E_{smooth} is the smoothness term which forces neighbouring pixels in the same image to have similar disparities. The data term $E_{data}(f)$ measures the cost of matching the function f.

$$E(f) = E_{data}(f) + E_{occ}(f) + E_{smooth}(f) \tag{21}$$

The data term used in [11] is defined as the cost of squared intensity differences (C_{SD}). For the following experiments we will only alter the data term, while keeping E_{smooth} and E_{occ} as defined in [11].

5.2. Cross-Zones Aggregation & Histogram Voting

For the local technique of energy minimisation we chose to test a cross-based aggregation as described in [24]. The algorithm consists in finding for each pixel a cross support zone. In the first step, a cross is constructed for each pixel. Given a pixel p, its directional arms (left, right, up or down) are found by applying the following rules:

- $D_c(p, p_a) < \tau$. The color difference (D_c) between the pixel p and an arm pixel p_a should be less than a given threshold τ. The color difference is defined as $D_c(p, p_a) = max_{i=\overline{1,nc}}|I_i(p) - I_i(p_a)|$, where $I_i(p)$ is the color intensity of the pixel p at channel i, and nc are the number of color channels considered.

- $D_s(p, p_a) < L$, where D_s represents the euclidean distance between the pixels p and p_a and L is the maximum length threshold.

Each pixel in the image has a cost given by the considered cost functions. The cost values in the support region are summed up efficiently using integral images. To select the disparity, the minimum cost value is selected using a Winner-Take-All strategy. Then a local high-confidence voting scheme for each pixel is used as described in [13].

6. Experiments

6.1. Datasets

There exist several challenging databases for testing the stereo matching algorithms, as presented in table 1. The HCI/Bosch Challenge [15] contains some difficult situations for all the stereo matching algorithms like: reflections, flying snow, rain blur, rain flares or sun flares, thus giving an insight of where the algorithms might fail. Unfortunately it does not come with a ground truth thus making difficult the evaluation of the different cost functions. Datasets like Van Syntetic stereo[22] and EISATS[17] have the advantage of having ground truth for all the pixels, but they are composed of synthetic images. One of the best known datasets for algorithm comparison is Middlebury[19], but the images are taken inside in controlled light conditions. Other datasets containing real road images are Make3D Stereo [18] and Ladicky[12] but provide ground truth for a limited number of pixels. KITTI [4] dataset provides real road images with ground truth for around 50% of the pixels, thus making a good dataset for cost functions comparison.

Dataset	Number of Images	Ground Truth	Scene	Image Type
KITTI [4]	389	YES (for 50% of px)	Road	Real
Middlebury[19]	38	YES (for 100% of px)	Laboratory	Real
EISATS[17]	498	YES (for 100% of px)	Road	Synthetic
Make3D Stereo [18]	257	YES (for 0.5% of px)	Road	Real
Ladicky[12]	70	YES - manual labels	Road	Real
HCI/Bosch Challenge[15]	451	NO	Road	Real
Van Syntetic stereo[22]	325	YES (for 100% of px)	Road	Synthetic

Table 1: Datasets comparison for stereo matching evaluation

In what follows all the numerical experiments are performed using KITTI stereo images. KITTI dataset is divided into 194 images in the training set for which the ground truth images is provided, and 195 images in the testing set for which an evaluation server should be used in order to have the results. Since only one submission in 72 hours is allowed on the evaluation server, and having an important number of situations to be tested, we have used only the 194 images from training set for all the experiments. All the cost functions in this paper are evaluated by the average percentage of erroneous pixels in all zones, occlusions included, and computed at 3 pixels error threshold.

6.2. *Insight into stereo matching cost function and radiometric distortions*

Radiometrical similar pixels refers to those pixels that correspond to the same scene point and have similar or in an ideal case the same values in different images [7]. Radiometrical differences or distortions are therefore the situations where corresponding pixels have different values. This is caused by differences of camera parameters (aperture, sensor) which can cause different image noises or vignetting, by the surface properties like non-Lambertian surfaces or by the fact that the images are acquired at different times (like is the case of some satellite imaging).

In this subsection we intend to compare the cost functions without any kind of aggregation in order to measure how discriminative the considered functions are. In [7] a comparison of cost functions in the context of radiometric distortions is also made, using six different cost functions in combination with three stereo methods. In [8] the experiments are extended to include more functions. Like in [8] we are interested to measure the discriminative power of different functions by analysing strictly per-pixel cost. For this a comparison between the magnitude of radiometric distortions found in the datasets Middlebury and KITTI is performed. Moreover the discriminative power of several functions is studied on the KITTI dataset.

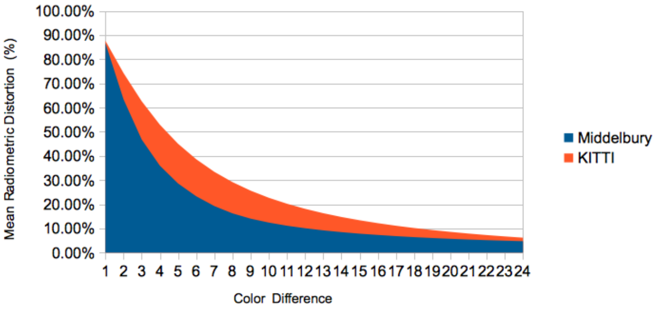


Figure 1: Percentage of radiometric distortions over the absolute color differences between corresponding pixels.

6.2.1. *Radiometric distortions statistics.*

In figure 1 is presented the mean percentage of radiometric distortions for the datasets Middlebury and KITTI, over the absolute difference between corresponding pixels. As the authors of [7] stated, the Middlebury dataset is taken inside a laboratory in controlled light conditions. Even so, for example at a color absolute difference of *five*, on the Middlebury dataset the average percentage of radiometric distortions is around 28%. On the other hand on KITTI dataset, where the images were collected outside, the average percentage of radiometric distortions at the same difference of color is larger than 45%.

6.2.2. *Discriminative power of cost functions*

For a second test we wanted to quantify how pertinent the information given by each cost function is in relation to all the possible disparities. This is the equivalent of computing the error rate of stereo matching using only these functions without any cost aggregation technique. Because some of the cost functions are defined in a neighbourhood, thus having an advantage in report with the others, we also compute the error given by each function when using a fixed aggregation window. The results for an error threshold of three pixels are presented in table 2.

For the cost functions we compare C_{AD}, C_{SD}, C_{CT} with a support window of 7×9 pixels (bit string of 63 elements),

Function	C_{AD}	C_{SD}	C_{CT}	C_{ADCT}	C_{CCC}	C_{ADCCC}	C_{cstent}	C_{klaus}	C_{ZNCC}	$C_{DIFFCCC}$	C_{DIFFCT}
Error NoAggr	85.8%	86.22%	71.9%	74.5%	62.3%	71.6%	68.05%	57.52%	**39.97%**	58.96%	66.51%
Error Window Aggr	42.20%	43.56%	26.92%	23.49%	26.51%	23.49%	27.29%	31.28%	28.68	22.36%	**21.60%**

Table 2: Error percentage of stereo matching no aggregation.

C_{CCC} with a support window of 7 × 9 pixels and a step of 2 (bit string of 55 elements), C_{ADCT} and C_{ADCCC}, C_{cstent}, C_{klaus}, C_{ZNCC}, $C_{DIFFCCC}$ and C_{DIFFCT}. For the results obtained with an aggregation window we have used one of 9 × 7 pixels. With no aggregation and winner takes it all strategy, the most discriminative function is the cost given by the $ZNCC$ with an error of 39.97%, followed by C_{klaus} with 57.52%. From the census based functions, $C_{DIFFCCC}$ provides the best results with an error of 58.96% followed by C_{CCC} with 62.3%. The combination of AD with either CT or CCC, overall increases the error rate at 71.9% and 71.6% respectively. Therefore from a discriminative point of view, C_{ZNCC}, C_{klaus} and C_{CCC} are the most competitive.

For the results obtained using a window aggregation and winner takes it all strategy, the proposed function based on mean sum of relative differences provides the best results: C_{DIFFCT} with 21.60%, followed by $C_{DIFFCCC}$ with 22.36%. These are followed by the functions based on *ADCensus*: C_{ADCT} and C_{ADCCC} both with 23.49%.

6.3. Matching cost comparison

We have optimised each cost function by performing a grid search for the parameters using the first three images from the KITTI training dataset. The optimised parameters were use throughout the experiments (see table 3).

Function	**Parameters**
C_{klaus}	$w = 0.2$
$C_{ADcensus}$	$\lambda_{census} = 90$; $\lambda_{AD} = 90$
$\boldsymbol{C_{DiffCT}}$	$\lambda_{census} = 55$; $\lambda_{Diff} = 95$
$\boldsymbol{C_{DiffCCC}}$	$\lambda_{census} = 55$; $\lambda_{Diff} = 95$
C_{cstent}	$\lambda_{census} = 80$; $\lambda_{AD} = 35$; $\lambda_{\Delta AD} = 80$

Table 3: Optimised parameters obtained with grid search.

6.3.1. Results Graph cuts stereo matching

The graph cuts minimisation algorithm was used as described in [11] and section 5.1. Graph cuts minimisation is an iterative process, with the error decreasing when increasing the number of iterations. One iteration takes around six minutes[4] to complete for an image of size 1241 × 376 pixels.

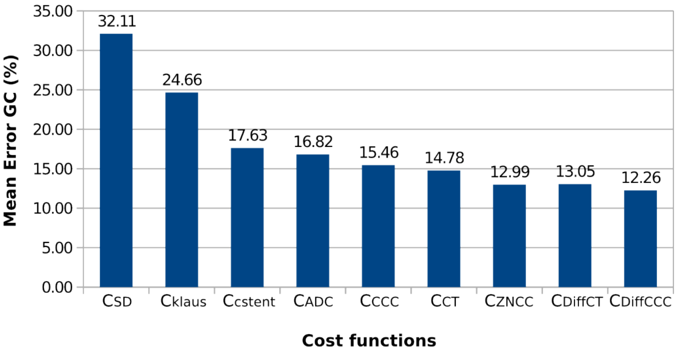


Figure 2: Mean error for each cost function using **graph cuts** stereo matching.

We have started the experiments using *six* iterations but we did not observed any significant improvement over using just one iteration, while the running time was considerably increased. Therefore all the experiments presented in this section were carried out with one iteration.

In order to show the importance of the data term for the energy function, we have tested the nine cost functions presented in section 3: C_{AD}, C_{Census}, $C_{CCCensus}$, C_{klaus}, $C_{ADcensus}$, C_{cstent}, C_{ZNCC}, $C_{DIFFCCC}$ and C_{DIFFCT}. This functions were used without an aggregation window with the except of C_{klaus} where a neighbourhood of 3 × 3 pixels is required by the algorithm and C_{ZNCC} where, for the same reasons, a neighbourhood of 9 × 7 pixels was used.

In figure 2 there are presented the mean error rate on all the 194 images from the training KITTI dataset. The error with C_{SD} is quite large, while with the other cost functions the error decreases significantly. The best overall performance is given by the proposed $C_{DIFFCCC}$ function with an error of 12.26%, followed by C_{DIFFCT} with 12.97% and very closely by C_{ZNCC} with 12.98%. In terms of computing time the C_{ZNCC} is the slowest function taking in average ten times longer to compute in comparison with the other functions.

6.3.2. Local energy optimisation based on cross zone aggregation

Without a real time constraint, the global energy optimisation technique can give very accurate disparity maps. In comparison, local techniques could achieve real time running with some trade-off concerning the quality of the disparity map. We have chosen to compare with the global

[4] on a computer with Dual Core 2.4 GHz single threaded

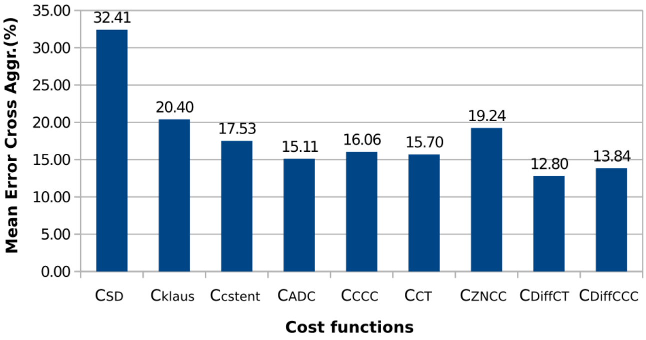


Figure 3: Mean error for each cost function using **local cross aggregation** stereo matching.

energy optimisation based on graph cuts a local optimisation based on cross zone aggregation and local high confidence voting [24] due to the promising results obtained on the Middlebury[19] dataset.

The same cost functions tested with the graph cuts were evaluated with the local energy optimisation. The color threshold for cross zone construction used is $\tau = 20$, as chosen in [24]. For the maximum arm length two different thresholds were used due to a predilection in the considered dataset of objects to have the same disparity in horizontal: vertical arm $L_{vertical} = 10$; horizontal arm $L_{horiz} = 17$. The results obtained on the KITTI dataset are presented in figure 3.

The overall results are better than those obtained with the graph cuts method (tested in a reasonable running time situation). When comparing the functions, the best results are obtained by our proposed functions based on sum of differences: $C_{DIFFCCC}$ and C_{DIFFCT}. C_{DIFFCT}, with a 13% error rate, gives better results than the $C_{DIFFCCC}$, with a 14.07% error rate, but the latter has a smaller running time of around 40%. The $DIFF$ based functions are followed as results by the $C_{ADCensus}$ and standard C_{CT} based cost functions.

6.4. Discussion

Even though the tested cost functions show different discriminative power, as seen in subsection 6.2 where C_{CCC} has proven to be the most discriminative, a cost aggregation or cost minimisation algorithm can change the ranking. For each minimisation method must be chosen a specific cost function. In figure 4 a visualisation of the output disparity map for each function in combination with the two stereo algorithms is shown. Columns one and three show the results obtained using the local stereo matching based on cross zone aggregation, while columns two and four the results obtained with graph cuts. The output results for two images is presented. While for the first image, results in columns one and two, a satisfactory disparity map is obtained with both of the stereo matching algorithm, the second image presented is more difficult due to large regions without texture.

For the graph cuts algorithm the proposed $C_{DIFFCCC}$ function provided the best results with very smooth disparity results in the road region but still erroneousness pixels could be found in textureless areas.

The local stereo matching algorithm gives comparable results with those of graph cuts at a much lower time cost. In this situation the best results are given by our proposed function C_{DIFFCT}. The disparity map is not as smooth as in the case of the graph cuts algorithm because we did not used any method of post-filtering. The main problems of the local minimisation technique based on cross-aggregation lies in big regions of similar color. The assumption when using an aggregation area is that in the considered region all the pixels have the same disparity. In practice large areas of same or similar color will not have the same disparity (for example road region and slanted walls).

7. Conclusion & Future Work

In this paper we have compared nine cost functions using two different stereo matching algorithms: a global method based on graph cuts and a local method based on cross zone aggregation with high confidence voting. Also we have proposed a new cost function based on mean sum of relative differences of pixel intensities. The functions are chosen to be robust to radiometric distortions since in real traffic situations these are very pronounced.

Experiments show that the results of local methods are comparable with those of global methods. In addition local methods have a high computing speed. Consequently, in the context of real time constraint of the intelligent vehicle application, our choice as a stereo matching algorithm is for the local method in combination with a cost function based on $DIFF$, ($C_{DIFFCCC}$).

As future work it would be ideal to test the functions on color road stereo images, since color is known to be more discriminative and therefore some of the functions could improve by the usage of it.

References

[1] Stan Birchfield and Carlo Tomasi. A pixel dissimilarity measure that is insensitive to image sampling. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20(4):401–406, 1998.

[2] Michael Bleyer and Margrit Gelautz. Simple but effective tree structures for dynamic programming-based stereo matching. In *International Conference on Computer Vision Theory and Applications (VISAPP)*, pages 415–422, 2008.

[3] Pedro F Felzenszwalb and Daniel P Huttenlocher. Efficient belief propagation for early vision. *International journal of computer vision*, 70(1):41–54, 2006.

[4] Andreas Geiger, Philip Lenz, and Raquel Urtasun. Are we ready for autonomous driving? the kitti vision benchmark suite. In *Computer Vision and Pattern Recognition (CVPR)*, Providence, USA, June 2012.

[5] Heiko Hirschmuller. Stereo processing by semiglobal matching and mutual information. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 30(2):328–341, 2008.

[6] Heiko Hirschmüller, Peter R Innocent, and Jon Garibaldi. Real-time correlation-based stereo vision with reduced border errors. *International Journal of Computer Vision*, 47(1-3):229–246, 2002.

[7] Heiko Hirschmuller and Daniel Scharstein. Evaluation of cost functions for stereo matching. In *IEEE Computer Vision and Pattern Recognition (CVPR)*, pages 1–8, 2007.

[8] Heiko Hirschmuller and Daniel Scharstein. Evaluation of stereo matching costs on images with radiometric differences. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 31(9):1582–1599, 2009.

[9] Andreas Klaus, Mario Sormann, and Konrad Karner. Segment-based stereo matching using belief propagation and a self-adapting dissimilarity measure. In *IEEE International Conference on Pattern Recognition*, volume 3, pages 15–18, 2006.

[10] Vladimir Kolmogorov and Carsten Rother. Comparison of energy minimization algorithms for highly connected graphs. In *ECCV*, pages 1–15. Springer, 2006.

[11] Vladimir Kolmogorov and Ramin Zabih. Computing visual correspondence with occlusions using graph cuts. In *IEEE International Conference on Computer Vision*, volume 2, pages 508–515, 2001.

[12] Lubor Ladickỳ, Paul Sturgess, Chris Russell, Sunando Sengupta, Yalin Bastanlar, William Clocksin, and Philip HS Torr. Joint optimization for object class segmentation and dense stereo reconstruction. *International Journal of Computer Vision*, pages 1–12, 2012.

[13] Jiangbo Lu, Gauthier Lafruit, and Francky Catthoor. Anisotropic local high-confidence voting for accurate stereo correspondence. In *Proc. SPIE-IS&T Electronic Imaging*, volume 6812, pages 605822–1, 2008.

[14] Xing Mei, Xun Sun, Mingcai Zhou, Shaohui Jiao, Haitao Wang, and Xiaopeng Zhang. On building an accurate stereo matching system on graphics hardware. In *IEEE International Conference on Computer Vision Workshops (ICCV Workshops)*, pages 467–474, 2011.

[15] S. Meister, B. Jähne, and D. Kondermann. Outdoor stereo camera system for the generation of real-world benchmark data sets. *Optical Engineering*, 51(02):021107, 2012.

[16] A Miron, S Ainouz, A Rogozan, and A Bensrhair. Cross-comparison census for colour stereo matching applied to intelligent vehicle. *Electronics Letters*, 48(24):1530–1532, 2012.

[17] Sandino Morales and Reinhard Klette. Ground truth evaluation of stereo algorithms for real world applications. In *Computer Vision–ACCV 2010 Workshops*, pages 152–162. Springer, 2011.

[18] Ashutosh Saxena, Jamie Schulte, and Andrew Y Ng. Depth estimation using monocular and stereo cues. IJCAI, 2007.

[19] Daniel Scharstein and Richard Szeliski. A taxonomy and evaluation of dense two-frame stereo correspondence algorithms. *International journal of computer vision*, 47(1):7–42, 2002.

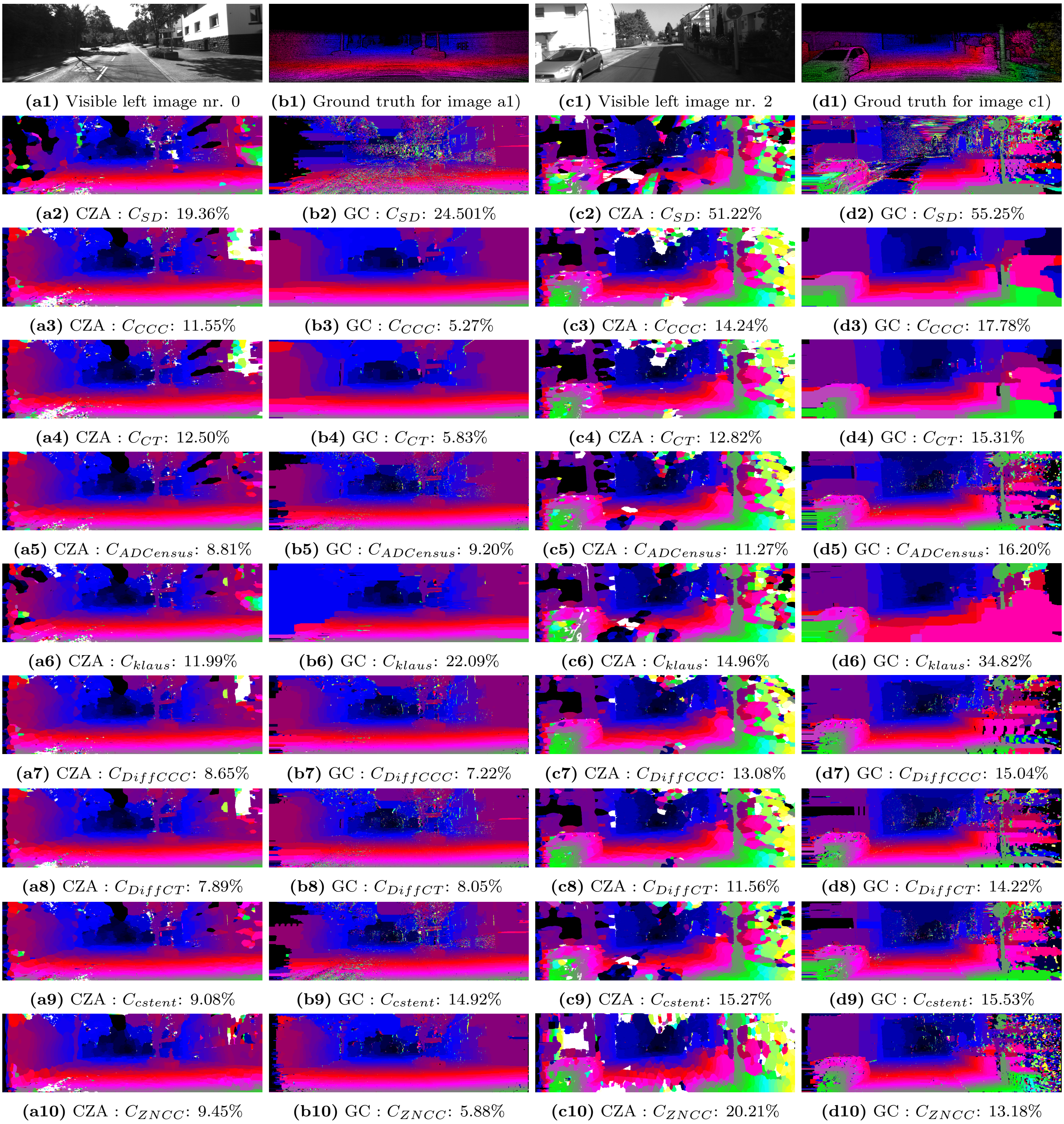
[20] C Stentoumis, L Grammatikopoulos, I Kalisperakis, E Petsa, and G Karras. A local adaptive approach for dense stereo matching in architectural scene reconstruction. 2013.

[21] Richard Szeliski, Ramin Zabih, Daniel Scharstein, Olga Veksler, Vladimir Kolmogorov, Aseem Agarwala, Marshall Tappen, and Carsten Rother. A comparative study of energy minimization methods for markov random fields with smoothness-based priors. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 30(6):1068–1080, 2008.

[22] Wannes Van Der Mark and Dariu M Gavrila. Real-time dense stereo for intelligent vehicles. *IEEE Transactions on Intelligent Transportation Systems*, 7(1):38–50, 2006.

[23] Ramin Zabih and John Woodfill. Non-parametric local transforms for computing visual correspondence. *ECCV*, pages 151–158, 1994.

[24] Ke Zhang, Jiangbo Lu, and Gauthier Lafruit. Cross-based local stereo matching using orthogonal integral images. *IEEE Transactions on Circuits and Systems for Video Technology*, 19(7):1073–1079, 2009.



(a1) Visible left image nr. 0 **(b1)** Ground truth for image a1) **(c1)** Visible left image nr. 2 **(d1)** Groud truth for image c1)

(a2) CZA : C_{SD}: 19.36% **(b2)** GC : C_{SD}: 24.501% **(c2)** CZA : C_{SD}: 51.22% **(d2)** GC : C_{SD}: 55.25%

(a3) CZA : C_{CCC}: 11.55% **(b3)** GC : C_{CCC}: 5.27% **(c3)** CZA : C_{CCC}: 14.24% **(d3)** GC : C_{CCC}: 17.78%

(a4) CZA : C_{CT}: 12.50% **(b4)** GC : C_{CT}: 5.83% **(c4)** CZA : C_{CT}: 12.82% **(d4)** GC : C_{CT}: 15.31%

(a5) CZA : $C_{ADCensus}$: 8.81% **(b5)** GC : $C_{ADCensus}$: 9.20% **(c5)** CZA : $C_{ADCensus}$: 11.27% **(d5)** GC : $C_{ADCensus}$: 16.20%

(a6) CZA : C_{klaus}: 11.99% **(b6)** GC : C_{klaus}: 22.09% **(c6)** CZA : C_{klaus}: 14.96% **(d6)** GC : C_{klaus}: 34.82%

(a7) CZA : $C_{DiffCCC}$: 8.65% **(b7)** GC : $C_{DiffCCC}$: 7.22% **(c7)** CZA : $C_{DiffCCC}$: 13.08% **(d7)** GC : $C_{DiffCCC}$: 15.04%

(a8) CZA : C_{DiffCT}: 7.89% **(b8)** GC : C_{DiffCT}: 8.05% **(c8)** CZA : C_{DiffCT}: 11.56% **(d8)** GC : C_{DiffCT}: 14.22%

(a9) CZA : C_{cstent}: 9.08% **(b9)** GC : C_{cstent}: 14.92% **(c9)** CZA : C_{cstent}: 15.27% **(d9)** GC : C_{cstent}: 15.53%

(a10) CZA : C_{ZNCC}: 9.45% **(b10)** GC : C_{ZNCC}: 5.88% **(c10)** CZA : C_{ZNCC}: 20.21% **(d10)** GC : C_{ZNCC}: 13.18%

Figure 4: Comparison between cost functions. On first row there are presented two left visible images (**a1** and **c1**) from the KITTI dataset with the corresponding ground truth disparity images (**b1** and **d1**) . On the following lines are the output disparity maps corresponding to different functions: on the first (a2-a10) and third column (b2-b10) the output obtained with the cross zone aggregation (CZA) algorithm, while on columns two (b2-b10) and fourth (d2-d10) the output of the graph cuts algorithm. Images a2-a10 and b2-b10 correspond to the disparity map computed for image a1 while the images c2-c10 and d2-d10 correspond to the disparity map computed for image c1.