Honda Research Institute EU
Honda Research Institute Europe GmbH
https://www.honda-ri.de/

A Meta-technique for Increasing Density of Local Stereo Methods through Iterative Interpolation and Warping

Aniket Murarka, Nils Einecke

2014

Preprint:

This is an accepted article published in Conference on Computer and Robot Vision. The final authenticated version is available online at: https://doi.org/[DOI not available]

A Meta-technique for Increasing Density of Local Stereo Methods through Iterative Interpolation and Warping

Aniket Murarka
Honda Research Institute USA, Inc.
425 National Avenue, Suite 100
Mountain View, CA 94043
Email: amurarka@honda-ri.com

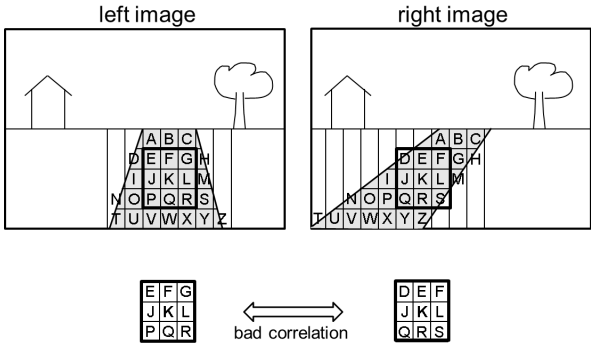
Nils Einecke
Honda Research Institute Europe GmbH
Carl-Legien-Strasse 30
D-63073 Offenbach/Main, Germany
Email: nils.einecke@honda-ri.de

***Abstract*—Despite much progress in global methods for computing depth from pairs of stereo images, local block matching methods are still immensely popular largely due to low computational cost and ease of implementation. However, such methods usually fail to produce valid depths in several image regions due to various reasons such as violations of a fronto-parallel assumption and lack of texture. In this paper, we present a simple and fast meta-technique for increasing the percentage of valid depths (depth map density) for local methods while keeping the percentage of pixels with erroneous depths, low. In the method, the original disparity map computed by a local stereo method is iteratively improved through a process of depth interpolation and image warping based on the interpolated depth. Image warping gives a mechanism for testing the validity of the interpolated depths allowing for incorrect depths to be discarded. Our results on the KITTI stereo data set demonstrate that, on average, we can increase density by 7-13% after a single iteration, for a 15-29% increase in computation and only a slight change in the outlier percentage, depending on the cost function used for matching.**

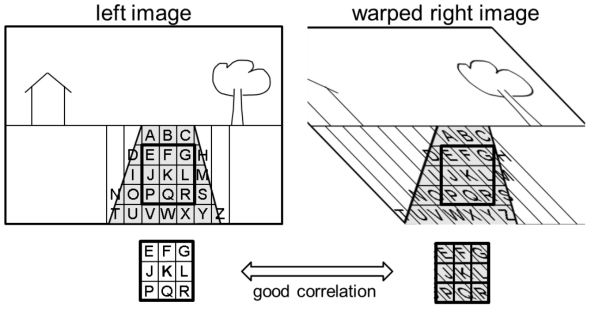
I. Introduction

There has been significant progress in stereo methods, particularly global methods that tend to produce accurate depth maps [1]. However, local block-matching stereo methods still remain in wide use due to their low computational cost and simpler implementation. Additionally, such methods are usually easily parallelizable allowing for very fast implementation on modern Graphics Processing Units (GPUs), e.g., [2] report a frame rate of 75Hz for a local block-matching method based on the census transform [3]. As a result, local methods scale well with increasing image size which can be very important for certain applications, such as autonomous driving where objects need to be detected early. Unfortunately, local stereo methods produce less dense depth maps compared to global methods. Towards this end, in this paper we present a meta-technique that increases the density of local block-matching methods (typically by 10-25%).

The inspiration for our method arises from the following observations. A well known problem of block-matching stereo methods is that they implicitly favor fronto-parallel surfaces by employing a single homogeneous disparity within a matching block [4], [1]. Fig. 1a depicts how this leads to correlation problems in the case of slanted surfaces causing the depth map to become very sparse in regions that violate the fronto-parallel assumption and also lack texture (e.g., Fig. 11b).



(a) The problem with the implicit constant disparity assumption of standard block-matching stereo: For highly slanted surfaces, e.g. a street, the disparity values within a patch can vary drastically which leads to strongly dissimilar corresponding patches. This dissimilarity will lead to a failure of the correspondence search due to a bad correlation value and leads to regions with bad/missing depths in the disparity map.



(b) Avoiding the problem described above: If we warp the right image by its correct depth, then we can ensure that correct intensities are matched. Since, the correct depth is obviously unknown, in this paper we use hypothesized depth values for regions with missing depths.

Fig. 1: (a) Problem caused by violation of the fronto-parallel assumption and (b) a possible fix (figure adapted from [4]).

Fig. 1b shows one way of avoiding mismatches between the left and right images as proposed by [4]. If we know

978-1-4799-4337-1/14 $31.00 © 2014 IEEE
DOI 10.1109/CRV.2014.59

the (partial) depth of the scene, we can warp the right image by the known depth to make it look like the left image and thereby use a single homogeneous disparity within a matching block. [4] used this to good effect to improve their depth estimates of street scenes by using a simple prior street surface depth model. However, in many cases, prior knowledge of the scene maybe be unknown or too complex to easily utilize. The method we present in this paper avoids the use of prior knowledge and can be applied to any scene in order to increase the density of the depth map estimated using local block-matching methods.

The basic concept is to utilize *hypothesized* depth estimates instead of prior knowledge. An initial depth map is computed using the local stereo method and the hypothesized depth map is obtained by filling in regions with missing depth using interpolation. The hypothesized depth map is then used to warp the right image and the warped image is used to re-compute a new depth map using the same local stereo method (with modifications to handle negative disparities - see Section III-D). The process is carried out iteratively until a fixed number of iterations or until the depth map stops changing. By warping we are able to reduce perspective effects for many of the regions with missing depths and thus improve the quality of block-matching depth estimates. Additionally, we add checks in the iterative process to ensure that incorrect depth estimates are rejected to reduce the effect of accumulated errors.

We have implemented our technique on top of five different local stereo methods (each based on a different cost function) and extensively evaluated its performance using the KITTI Stereo Benchmark [5]. We see density gains anywhere in the range of 3% to 26% depending on a variety of parameters. Typically though, we show that a gain of about 10% can be expected after a single iteration for most cost functions, at a fraction (usually 20%) of the original computational cost and a slight change in the outlier percent - details are presented in Section IV. The next section presents related work, followed by a description of the algorithm.

II. Related Work

There have been a few approaches that have used image warping to improve stereoscopic depth estimation. In the 1990's, [6], [7] used image warping to reduce the computational time of stereo processing to a reasonable value. By using a planar model of the street, the view of one camera was warped into the view of the other camera thereby reducing the disparity of street pixels to almost zero. Every pixel for which no correspondence was found in a small search range was assumed to be an obstacle. In [4] disparity maps from warped and unwarped stereo images were combined. The warped images were created by utilizing prior depth knowledge about the scene. In particular, for street scenes, one stereo image was warped according to a planar model describing the relation between the camera and the street surface. By doing so perspective artifacts of the warped surface were compensated leading to improved disparity even when the model did not perfectly match. Unfortunately, such model-based approaches

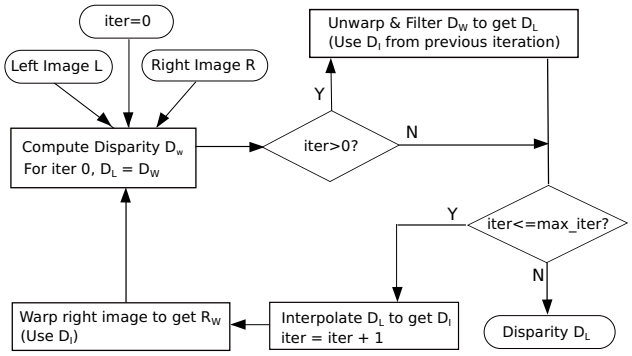


Fig. 2: An overview of our algorithm - see text for details.

can only be applied when prior knowledge about the scene is available. Furthermore, the improvement is restricted to the area where the model is suitable, e.g. the warping according to a planar street model will not improve the depth accuracy for pedestrians or cars in the street scene.

Some approaches, such as those in [8], [9], integrate image warping directly into the correspondence search for very accurate depth estimations. The advantage of the direct integration is a much higher accuracy as compared to methods that fit planes after the stereo computation. The main drawback of this approach is the high computational cost as compared to standard block-matching because the image warping has to be done multiple times for each image patch.

Iterative stereo processing has also been widely used in other work. Mainly, this technique is (inherently) used in global methods like Belief Propagation [10] or Graph-Cuts [11], [12] where starting from an initial (often random) disparity map, the disparities are improved by means of some iterative processing. In Belief Propagation, messages of disparity cost are passed around in an iterative fashion to improve the disparity values. On the other hand in Graph-Cut methods, so called α-β-swaps or α-expansions are used to iteratively change the disparity maps a tiny bit but keeping only improved changes for the next iteration. The major problem of Graph-Cut and Belief Propagation based methods is the high computational cost which often leads to run-times an order of magnitude higher than that of local methods. Another technique is to iteratively improve disparities starting from a disparity map computed by means of any stereo method. A typical approach here is to use anisotropic diffusion [13]. For example [14] achieved good results by setting the diffusion parameters depending on the color similarity of pixels. The problem of diffusion processes is to define a good stopping criterion. If too many iterations are done then the resulting disparity map can be worse than the starting disparity map.

III. Iterative Interpolation and Warping based Stereo Algorithm

Fig. 2 gives an overview of our iterative algorithm for increasing the density of a local block-matching stereo method. Given a pair of left and right stereo images as input, the first step is to compute a depth map, D_L, for the left image using the original local method. Next, regions in the depth map with

missing depths are filled in using linear interpolation [1]. The interpolated depth map, D_I is then used to warp the right image to reduce perspective effects and make block matching more effective as explained in Section I. The original stereo method is now re-applied to the left image and the warped right image to compute a new warped depth map, D_W (for this computation, the original method has to be modified to search for both negative and positive disparities for reasons explained below). The warped depth map D_W is then unwarped to get disparities relative to the original right image and is also filtered to remove incorrect disparities to get a new D_L. Filtering is partly done by comparing the new depth map to the interpolated depth map, D_I from the last iteration and keeping matching depths. The new D_L can again be interpolated, used to warp the right image and compute a new depth map. In short, the process continues until the maximum number of iterations is reached or the change in the depth map is below a threshold (in practice we limit to 8 iterations). Fig. 3 gives an example result from the method.

In the rest of this section we highlight important details in each step. We assume the input images are rectified into a parallel configuration.

A. Computing Disparity using Original Images

First, an initial disparity map, D_L, for the left image is computed from the unwarped left and right stereo images (L, R) using a local block-matching stereo method. We have tested five different local methods as detailed in the Experimental Results section.

We also use standard post processing steps [1] to remove bad disparity values, in particular, we apply a Left-right check and remove small disparity regions. This is fairly important otherwise the resulting depth maps are quite noisy.

B. Depth Map Fill-In: Linear Interpolation

After post-processing, the depth map has many regions with missing depths (i.e., holes, see Fig. 3b). Such a depth map cannot be used directly for warping as it would only warp parts of the image for which depth is already known and would not provide any benefit. We need a process to *hypothesize* depths for the missing regions - this is done in two ways.

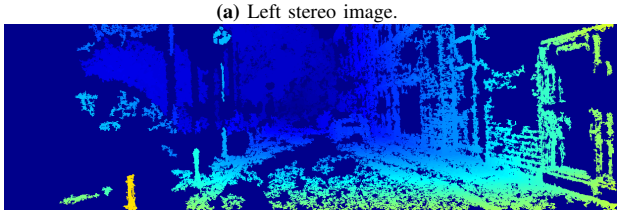
In the first method, called *Horizontal Interpolation*, we just fill-up each row horizontally by linearly interpolating between the first left and right pixel having a valid disparity entry, as long as the difference in disparities is within a threshold (we use a large threshold of 40 pixels). The remaining pixels are simply filled with the smallest neighboring disparity.

In the second method, called *Multi-directional Interpolation*, we change the direction of interpolation for every iteration. In this case we equally distribute the interpolation direction according to the iteration number around 180 degrees. For example, in case of two iterations we interpolate first horizontally and then vertically. In case of four iterations we first interpolate horizontally, $0°$, then vertically, $90°$, then at $45°$ and finally at $135°$. The impact of these two different fill-up variants is investigated in the next section. The interpolated depth map is denoted D_I.

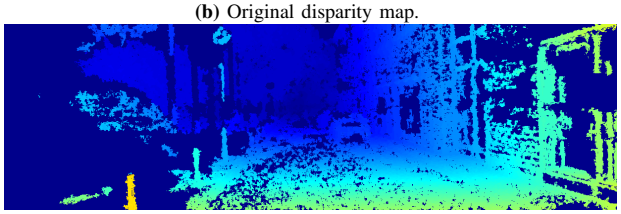
[1]Other methods can be used to "fill-in" and are a subject of future work.



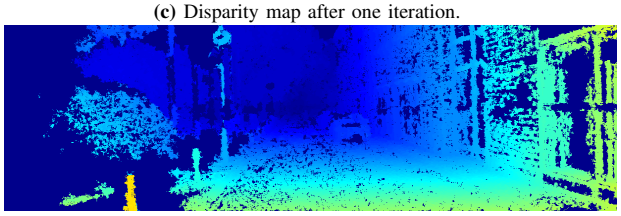
(a) Left stereo image.



(b) Original disparity map.



(c) Disparity map after one iteration.



(d) Disparity map after eight iterations.

Fig. 3: An example result of our proposed method. The sub-figures (a) and (b) show the left stereo image and the original block-matching disparity map respectively, computed using the Rank Transform [3] cost function. (c) shows the disparity map after one iteration and demonstrates significant improvement in depth estimates for the road and some improvements for the building on the right. (d) shows the disparity map after eight iterations showing significant improvements for both the road and building.

C. Warping Right Image

D_I is then used to warp the right stereo image (R) to get a new stereo image R_W. In particular, we compute the intensity at pixel position (x, y) in R_W from R as follows,

$$R_W(x, y) = R(x - D_I(x, y), y) \tag{1}$$

Since, in parallel stereo camera images corresponding pixels lie on the some image line the y-position is not altered by the warping. Also, in general the disparity values are not integers, specially after interpolation, and so in practice the intensities for R_W are obtained by linear interpolation of neighboring intensity values in R.

D. Computing Disparity using Warped Images

After warping, appearance differences between the new stereo image pair (L, R_W) are reduced. This means that

computing a new disparity map D_W from the new stereo pair is less prone to perspective effects because the new stereo images are much similar to one another. In the ideal case (i.e., D_I has no errors), the new stereo images would look exactly the same. However, D_I will not be perfect so the warped stereo image pair will still show some appearance differences. This in turn means that D_W is likely not perfect and is the reason we propose to repeat the warping and depth estimation for several iterations.

When finding correspondences between the left image and the warped right image we need to search negative disparities as well. This is because the disparity estimates in D_I might be higher than the real disparity for some pixels and as a result the pixels in the warped right image might have been shifted too much to the right. It might appear at this stage that we have effectively increased our disparity search space. This is however not the case because warping reduces the *residual* disparity to small values, and so the overall disparity search range can be reduced in the following iterations. For instance the original search range could be from 0 to 128 pixels, whereas the search range in subsequent iterations might be as low as -8 to 8 pixels. We conduct a thorough evaluation of the effect of different disparity search ranges in the following Experimental section.

Finally, we apply the two post-processing steps as before (Left-right Check and removal of small disparity regions) to remove incorrect disparities to get the final warped disparity map D_W.

E. Unwarping and Filtering Warped Disparity Map

The depth map D_W contains disparities that relate pixels in the left image to pixels in the warped right image. This depth map needs to "unwarped" to get depth values relative to the original right image, i.e., to get a new D_L. We do this by adding to D_W, the disparity by which its corresponding pixel in the warped right image was shifted, i.e.,

$$D_L(x,y) = D_W(x,y) + D_I^{prev}(x - D_W(x,y), y) \quad (2)$$

as the warped right image was obtained using the interpolated depth map from the last iteration, denoted D_I^{prev} here.

Once we have the new depth map, D_L, we further filter it to prevent accumulation of errors over multiple iterations. In particular, we do the following: (i) we ensure that D_L does not have any disparities outside the original search range; (ii) pixels which had correctly estimated depth values in the last iteration (i.e., after the two post-processing checks) are reassigned their last values; and (iii) pixels that had depth values computed for the first time in this iteration, are allowed to keep their depth values only if they are within a threshold of the interpolated ("hypothesized") depth values, i.e., $D_L(x,y)$ is accepted only if $|D_L(x,y) - D_I^{prev}(x,y)| < t$ (we choose a value of 0.5 pixels for t).

The reason for doing steps (ii) and (iii), is that the interpolated depth map is not perfect and as a result there are regions with sharp discontinuities and other errors in the warped right image. These errors in the warped right image can cause incorrect depth values to be estimated for some pixels. Therefore, we always prefer to keep older disparities over new ones and furthermore discard disparities that don't agree with the interpolated disparities.

In fact, the whole algorithm can be viewed in a *hypothesize and test* framework where we propose depth hypotheses through interpolation and then either reject or accept these hypotheses based on the test of whether the hypothesized disparities match the depth estimates obtained from warping.

IV. Experimental Results

For evaluating our novel approach we applied it to block-matching stereo with five different cost functions and tested it on the KITTI Stereo Benchmark [5][2]. For cost functions we used *summed absolute difference* (SAD), *rank transform* (RT) [3], *census transform* (CENSUS) [3], *normalized cross-correlation* (NCC) and *summed normalized cross-correlation* (SNCC) [15]. In addition, we considered the effect of the warped disparity search range and interpolation type: (i) We varied the warped image disparity search range (denoted DR in the plots) from [-8:8] to [-128:128]. However, in the interest of space only a fraction of those results are presented here. (ii) We tested both horizontal and multi-directional interpolation. In general, we found multi-directional interpolation to give significantly better results when doing several iterations. All runs were done for 8 iterations (after the original disparity map computation) to see the effect of increasing iteration.

Finally, the metrics we used to evaluate the methods are: (i) depth map density; (ii) increase in density; (iii) percent of outlier/erroneous pixels; i.e., pixels that have a disparity error of greater than 3 pixels; (iv) average disparity error in pixels; (v) run-time ratios. Metrics (i), (iii), and (iv), are as used in the KITTI benchmark [5].

TABLE I: Run-time ratios for different cost functions and disparity ranges.

DR/Metric	SAD	NCC	SNCC	RT	CENSUS
[-8:8]	0.19	0.17	0.15	0.29	0.21
[-16:16]	0.30	0.29	0.27	0.39	0.32
[-32:32]	0.54	0.53	0.51	0.59	0.55

In Table I we show the run-time ratio, defined as the ratio of the computation time required for an iteration (after the original depth map has been computed) to the original depth map computation time, as a function of the warped disparity range, DR. We can see that for small DR and a low number of iterations, our algorithm has low computational requirements. To get additional insight into the ratios, consider that the local stereo methods tested here have a time complexity of $O(mnq)$ where $m \times n$ is the image size and q is the number of disparities searched. Disparity computation at each additional iteration has a time complexity of $O(mnp)$, where p is the number of warped disparities considered. In addition, the operations of interpolation, warping, and filtering have time

[2]Following the KITTI benchmark instructions we only use the 194 training images to evaluate all the cost functions and demonstrate the effect of parameters. The test set images are only supposed to be used for evaluating the best performing setting which we do later in the section.

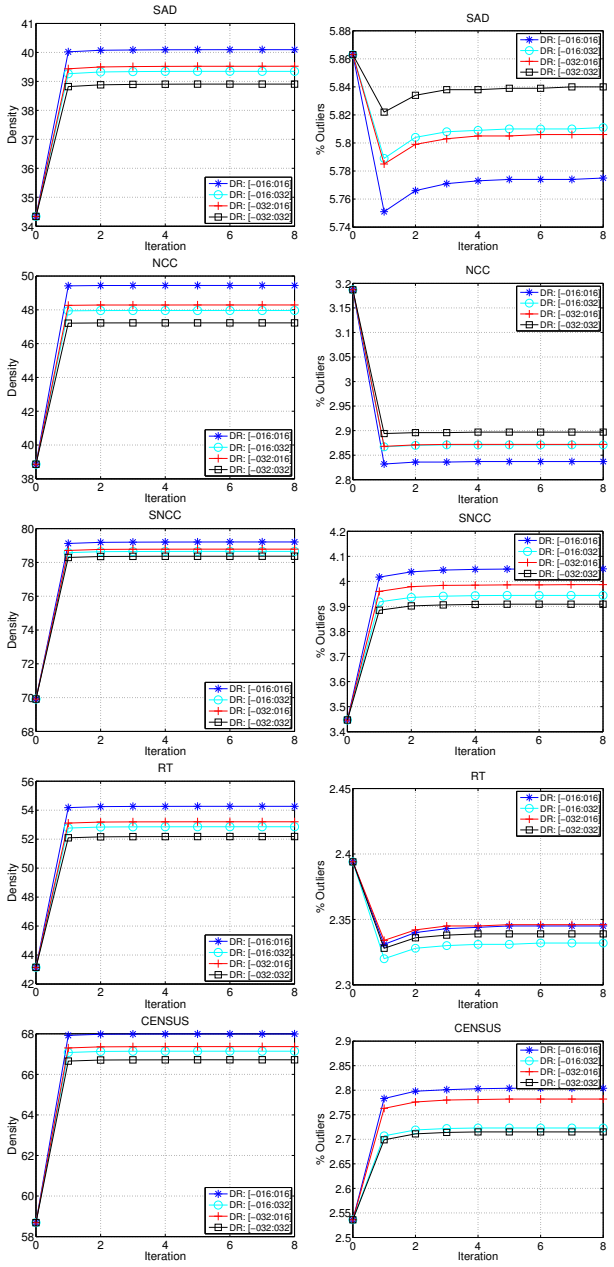


Fig. 4: [Horizontal Interpolation] For each cost function: Density vs. Iterations and %Outliers vs. Iterations for some symmetric and asymmetric disparity search ranges (DR).

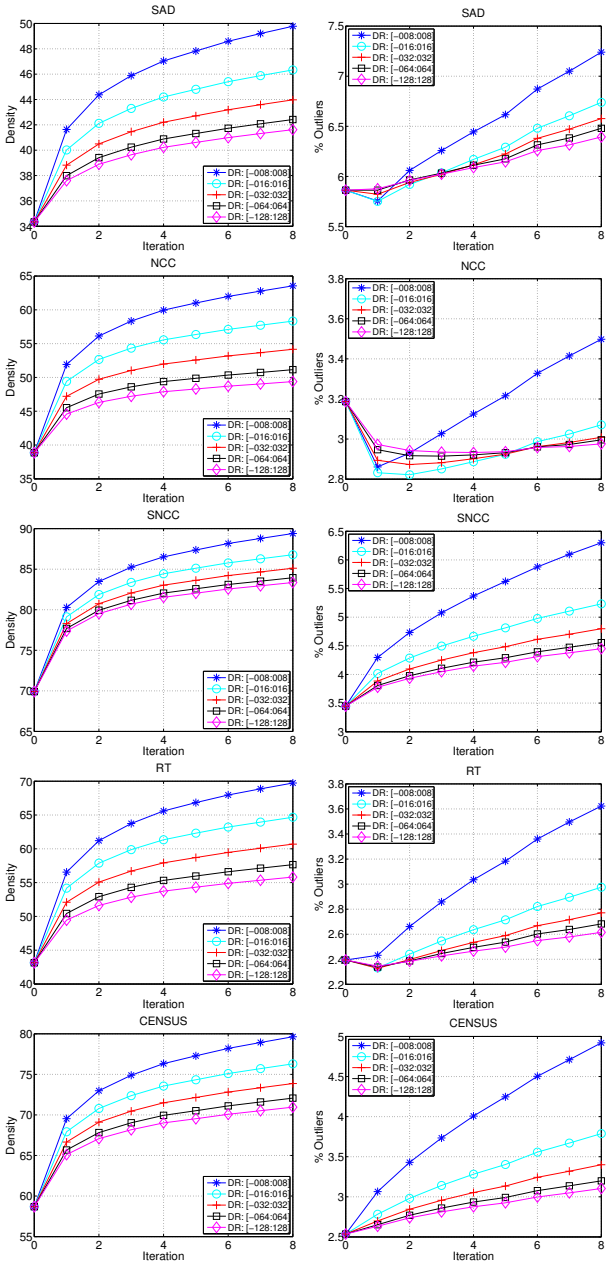


Fig. 5: [Multi-directional Interpolation] For each cost function: Density vs. Iterations and %Outliers vs. Iterations for symmetric DR from [-8:8] to [-128:128].

complexity of $O(mn)$ and will add to the computation time but not significantly. So, in general, we expect the run-time ratios to be somewhat larger than p/q. In our case $q = 129$ (corresponding to a search range of 0 to 128), and thus, $p/q = \{0.13, 0.26, 0.50\}$ for DR=$\{$[-8:8],[-16:16],[-32:32]$\}$ respectively. From Table I we see that as DR increases, the run-time ratios for the cost functions get closer to the p/q value as expected (with NCC and SNCC showing most agreement).

Fig. 4 plots, for each cost function, Density vs. Iterations and %Outliers vs. Iterations for disparity search ranges (DR) from [-16:16] to [-32:32]. This is for the case when we use *Horizontal Interpolation* to fill-in missing disparity values at each iteration. Things to note here: (i) The algorithm converges after just 1-2 iterations. This is to be expected because of step (iii) in Section III-E where we reject depth values too different from interpolated values. As a result there is no significant change in the depth map after the first iteration because we are always interpolating in the same direction. (ii) The density increases by up to 11% after the first iteration depending on the cost function. (iii) %Outliers actually decrease for SAD, NCC, and RT (by a small amount). For CENSUS, they go up by less than 0.3%. For SNCC, the increase is around 0.5%.

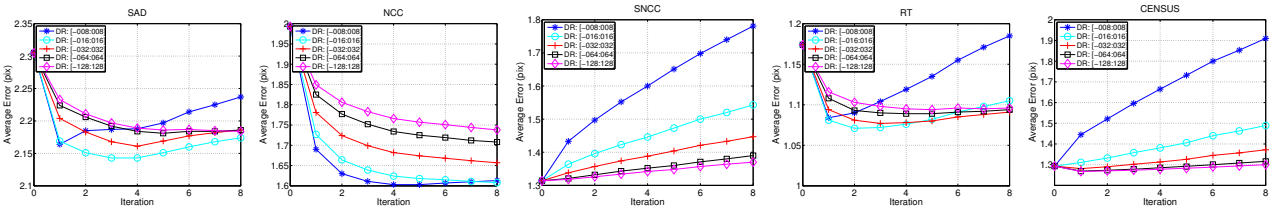


Fig. 6: [Multi-directional Interpolation] For each cost function: Average Disparity Error vs. Iterations for various DR.

Fig. 5 is similar to the last plot, except it is for multi-directional interpolation. It plots, for each cost function, Density vs. Iterations and %Outliers vs. Iterations for *symmetric* disparity search ranges (DR) from [-8:8] to [-128:-128]. In addition, Fig. 6 shows Average Disparity Error vs Iterations for the same DR and cost functions. Comparing Fig. 5 to Fig. 4, we can see the benefit of interpolating along multiple directions - each iteration results in density increases. We also see that for a large DR of [-128:128], we get the lowest increase in density and generally the lowest increase in %outliers (except for NCC). For the smallest DR of [-8:8], we get the largest increase in density along with the largest increase in %outliers (in general). Furthermore, for DR of [-8:8], the increase in %outliers and average disparity error is significantly higher as the number of iterations increase. A DR of [-16:16] exhibits more stable behavior and is probably a safer choice giving a good trade-off between computation, density, and outliers.

Another point of interest is that, in general, the average disparity error actually reduces for SAD, NCC, and RT, indicating that for these cost functions, the iterative scheme has the added benefit of reducing error. NCC in particular benefits most as the %outliers also drop for all DR (except [-8:8]).

Fig. 7 compares results for CENSUS for 3 additional small asymmetric DRs: [-8:16], [-16:8], and [-8:32]. Based on these plots we see that having a smaller negative disparity search benefits both density and %outliers. The probable reason for asymmetric DRs working better than similar symmetric DRs, is that the warped right image has many pixels which are not shifted from their original position and hence these pixels favor a smaller negative search range.

Fig. 8 shows plots of outlier percent variation with density increase for all metrics. These plots show that for a disparity search range of [-8:8] we get a 10-15% increase in density for all cost functions except SAD. Also, the plots show that for a disparity search range of [-16:16], density increases by 8-11% for all cost functions (except SAD) and that %outliers don't increase as much as for [-8:8].

In the KITTI Benchmark, the standard evaluation considers %outliers and average disparity error for depths maps with 100% density. As a result, all missing densities are filled with simple background interpolation. Fig. 9 plots %Outliers and Average Disparity Error vs Iteration using the KITTI scheme of filling remaining holes with background disparities. This evaluation demonstrates that %outliers actually fall by using our iterative technique for all cost functions. This means

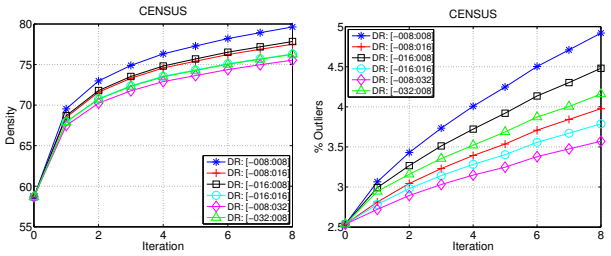


Fig. 7: [Multi-directional Interpolation] For CENSUS: Plots of Density vs. Iterations and %Outliers vs. Iterations for small asymmetric DR. They show that small negative search ranges are beneficial for both density and %outliers.

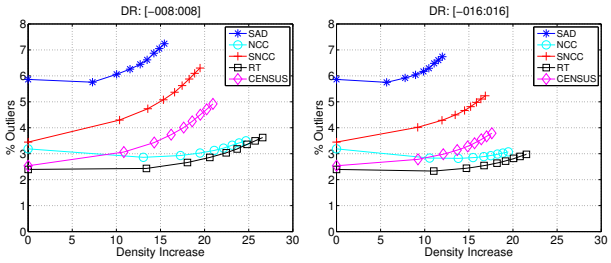


Fig. 8: [Multi-directional Interpolation] For DR of [-8:8] and [-16:16]: %Outlier vs Density Increase for all metrics. The results show that the density can be increased by 10% and more for only a small increase in %outliers.

that the disparity values computed by our iterative matching scheme are much better than the background filling values. We also see a reduction in the average disparity error for NCC, RT, and CENSUS. For SNCC and SAD the error decreases initially but later increases with iterations but still stays below or very close to the original disparity map error. In general, this highlights the performance of the iterative system compared to a simple baseline interpolation system (iterations 0 in Fig. 9). Although background filling always leads to 100% density it results in quite high %outliers and disparity errors (compare vertical axes of Fig. 9 from background filling results to Fig. 5 and Fig. 6 using only our iterative scheme). The reason is that simple interpolation has no testing for the correctness of the interpolation. In our framework, by providing a test for the interpolated disparities, we are able to increase density while keeping errors low.

Table II shows results from evaluating on the KITTI test image set. As explained earlier, we only do this for one parameter setting, in particular for the CENSUS cost function, DR of [16:16], 2 iterations, and Multi-directional Interpolation. The corresponding results for the training set for estimated pixels (see Fig. 5 and 6) are %outlier=2.98, average error=1.3px, and density=70.79% showing fairly close agreement.

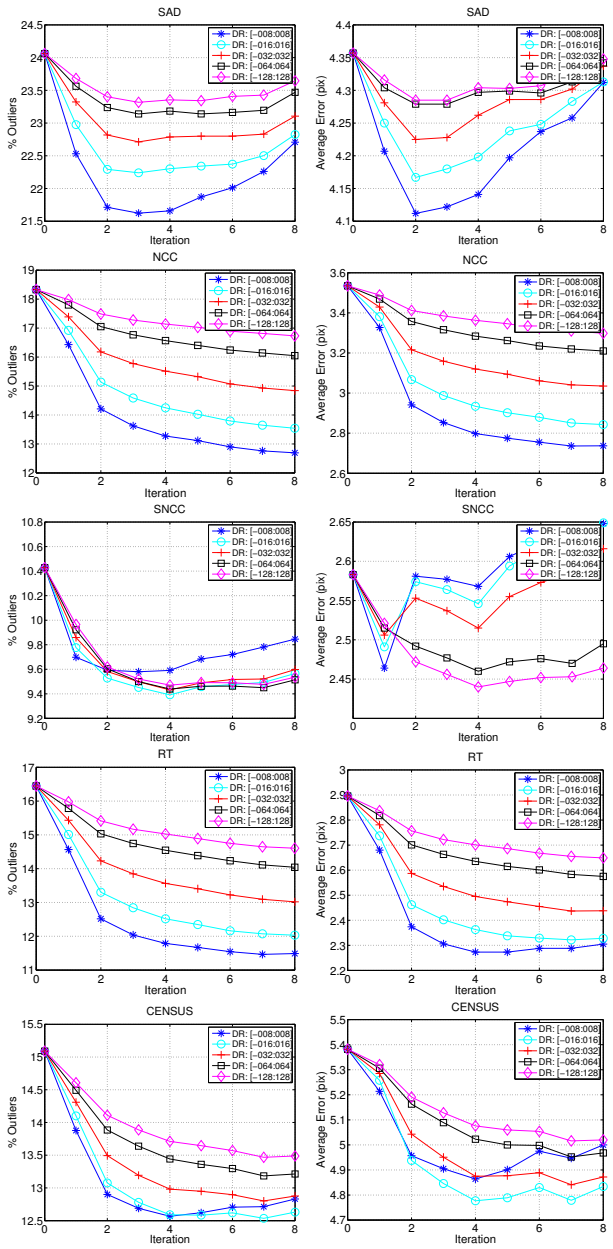


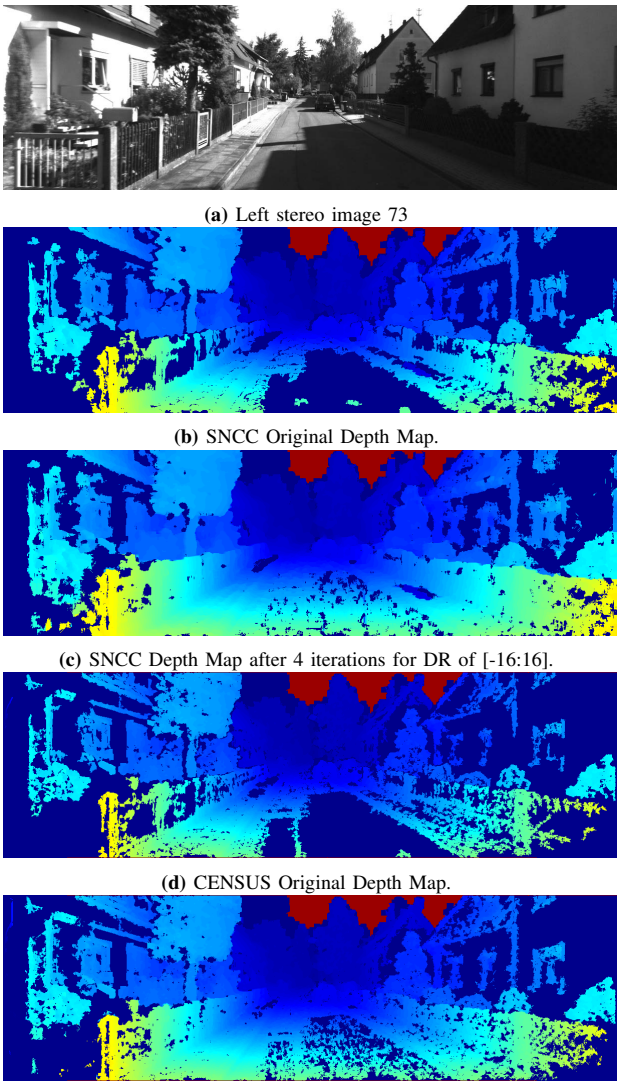
Fig. 9: [Multi-directional Interpolation] For each cost function: %Outlier vs. Iterations and Average Error vs. Iterations for fully interpolated depth maps.

Fig. 10 and 11 present qualitative results. These figures show that the iterative scheme is able to fill large holes in depth maps, even in challenging situations such as the presence of shadows, and works for both horizontal (road) and vertical (fence) structures.

To summarize, we note the following: (i) If we are just interested in a single iteration, then a DR of [-8:8] works quite well at increasing density and keeping %outliers low. (ii) For more iterations, a DR of [-16:16], or a similar asymmetric DR of [-8:16] or [-8:32], provides a good trade-off. (iii) NCC benefits greatly from the iterative scheme in terms of reduction in %outliers and average error. (iv) RT benefits most in terms of increase in density. It also shows a reduction in average disparity error. (v) Multi-directional interpolation works significantly better than just simple horizontal interpolation over multiple iterations. (vi) The method has low computational cost per iteration for small DR.

TABLE II: Results from the KITTI test set for CENSUS transform for 2 iterations, DR of [-16:16], and Multi-directional interpolation. 1st row shows results for estimated pixels only and 2nd rows shows results for all pixels after interpolation by the KITTI server.

Evaluation Type	%Outliers	Average Error	Density
Estimated	2.49%	1.3px	70.85%
Interpolated	10.78%	3.3px	100.00%



(a) Left stereo image 73

(b) SNCC Original Depth Map.

(c) SNCC Depth Map after 4 iterations for DR of [-16:16].

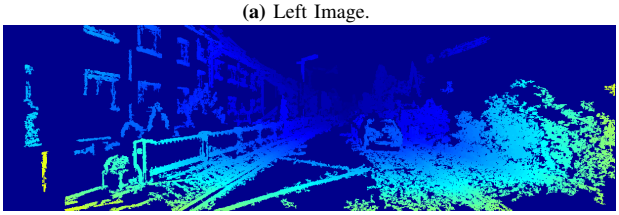
(d) CENSUS Original Depth Map.

(e) CENSUS Depth Map after 4 iterations for DR of [-16:16].

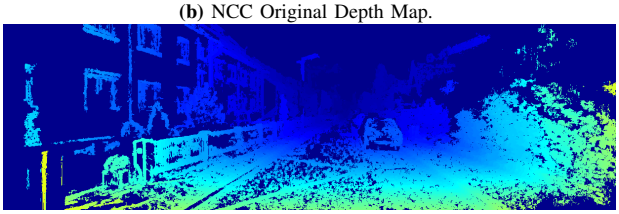
Fig. 10: The pictures show improvement in the depth maps (after 4 iterations) for a challenging situation. The road depths are estimated even in the presence of shadow which creates a region of very low texture. Additionally, the fence on left also has depths estimated.



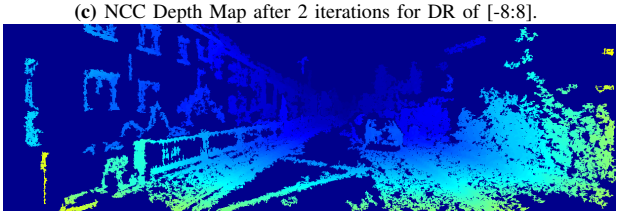
(a) Left Image.



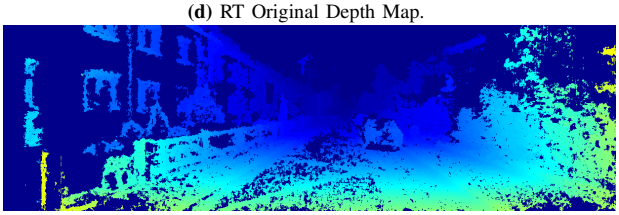
(b) NCC Original Depth Map.



(c) NCC Depth Map after 2 iterations for DR of [-8:8].



(d) RT Original Depth Map.



(e) RT Depth Map after 2 iterations for DR of [-8:8].

Fig. 11: The pictures show how a large region of missing depth in the road is filled after 2 iterations for both NCC and RT cost functions. Additionally, missing depths in the fence on left are also filled up.

V. Conclusion

In this work, we presented a novel processing scheme for stereoscopic depth computation that improves the density of the resulting depth maps. Instead of just computing the depth once we alternate between depth computation, interpolation, and stereo image warping for some iterations. The stereo images are warped according to the depth map computed in the last iteration step. Doing so, perspective effects in the stereo images are reduced which improves the correlation search in the stereo computation. Furthermore, the warping decreases the necessary search range so that only the first depth computation needs to be run with the full search range and all following iterations can use a much smaller search range. Due to this the run-time of the iterative stereo processing is only marginally longer than that of single stereo processing.

In order to test our new iterative processing scheme we applied it to local block-matching stereo for five different cost functions and evaluated the performance on the KITTI benchmark [5]. The results demonstrate that the iterative stereo processing scheme increases the density of the depth maps between 7-13% after just the first iteration, for a 15-29% increase in computation time per iteration, and only a slight change in outlier percentage depending on the cost function.

Since our scheme makes few assumptions about the actual stereo computation it should be possible to combine it with a number of stereo algorithms. In particular, in the future, we want to investigate the effect our technique has on Semi-global stereo methods [16]. Additionally, methods used for hypothesizing (i.e., filling-in) missing depth values are also open to further investigation.

References

[1] R. Szeliski, *Computer Vision: Algorithms and Applications*. Springer, 2010.

[2] M. Weber, M. Humenberger, and W. Kubinger, "A very fast census-based stereo matching implementation on a graphics processing unit," in *IEEE International Conference on Computer Vision (ICCV) Workshops*, 2009.

[3] R. Zabih and J. Woodfill, "Non-parametric local transforms for computing visual correspondence," in *Proceedings of Third European Conference on Computer Vision (ECCV)*, 1994, pp. 151–158.

[4] N. Einecke and J. Eggert, "Stereo image warping for improved depth estimation of road surfaces," in *Proceedings of the Intelligent Vehicles Symposium (IV)*, 2013, pp. 189–194.

[5] A. Geiger, P. Lenz, and R. Urtasun, "Are we ready for autonomous driving? the KITTI vision benchmark suite," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*, 2012, pp. 3354–3361.

[6] Q.-T. Luong, J. Weber, D. Koller, and J. Malik, "An integrated stereo-based approach to automatic vehicle guidance," in *Proceedings of the Fifth International Conference on Computer Vision*, 1995, pp. 52–57.

[7] M. Bertozzi and A. Broggi, "GOLD: A parallel real-time stereo vision system for generic obstacle and lane detection," *IEEE Transactions on Image Processing*, vol. 7, no. 1, pp. 62–81, 1998.

[8] M. Habbecke and L. Kobbelt, "Iterative multi-view plane fitting," in *Proceedings of the 11th International Fall Workshop Vision, Modeling, Visualization (VMV)*, 2005, pp. 73–80.

[9] N. Einecke, S. Rebhan, V. Willert, and J. Eggert, "Direct surface fitting," in *Proceedings of the Fifth International Conference on Computer Vision Theory and Applications (VISAPP)*, 2010, pp. 125–133.

[10] P. F. Felzenszwalb and D. P. Huttenlocher, "Efficient belief propagation for early vision," *International Journal of Computer Vision*, vol. 70, no. 1, pp. 41–54, 2006.

[11] S. Roy and I. J. Cox, "A maximum-flow formulation of the n-camera stereo correspondence problem," in *Proceedings of the Sixth International Conference on Computer Vision (ICCV)*, 1998, pp. 492–499.

[12] V. Kolmogorov and R. Zabih, "Computing visual correspondence with occlusions using graph cuts," in *Proceedings ot the International Conference on Computer Vision (ICCV)*, 2001, pp. 508–515.

[13] P. Perona and J. Malik, "Scale-space and edge detection using anisotropic diffusion," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 12, no. 7, pp. 629–639, 1990.

[14] A. Banno and K. Ikeuchi, "Disparity map refinement and 3D surface smoothing via directed anisotropic diffusion," in *IEEE Workshop on 3D Digital Imaging and Modeling (3DIM)*, 2009, pp. 1870 – 1877.

[15] N. Einecke and J. Eggert, "A two-stage correlation method for stereoscopic depth estimation," in *Proceedings of the International Conference on Digital Image Computing: Techniques and Applications (DICTA)*, 2010, pp. 227–234.

[16] H. Hirschmuller, "Accurate and efficient stereo processing by semi-global matching and mutual information," in *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*, vol. 2. IEEE, 2005, pp. 807–814.