Gipuma: Massively Parallel Multi-view Stereo Reconstruction

SILVANO GALLIANI[1], KATRIN LASINGER[1] & KONRAD SCHINDLER[1]

Abstract: We describe a method for dense multi-view matching and 3D point cloud reconstruction, which has been developed at the Institute of Geodesy and Photogrammetry of ETH Zurich. The method generates high-quality point clouds from oriented images very efficiently, by massively parallel processing on graphics processing units (GPUs). It is available as open-source code. Technically, the method is an extension of the PatchMatch Stereo algorithm: 3D depth values and surface normal vectors are iteratively propagated across the image and refined, in order to find a maximally photo-consistent depth map and normal field for each view. Photo-consistency is computed in a slanted tangent plane, such that the reconstruction does not suffer from fronto-parallel bias. We extend PatchMatch to 3D scene space, such that photo-consistency can be aggregated over multiple views, which allows for more robust and more accurate depth estimation. Moreover, the sequential propagation is replaced by a local, diffusion-like scheme, such that the computation can be massively parallelised. All computations are local, thus computation time is linear in the image size and inversely proportional to the number of parallel threads. Moreover, memory requirements are also modest, since only four values per pixel must be stored. Experiments on benchmark datasets show that our method delivers point clouds (respectively, surfaces) with high accuracy and completeness, across a range of applications.

1 Introduction

In this work we present a method to reconstruct dense, accurate 3D point clouds from oriented images. 3D shape reconstruction is a fundamental task of photogrammetry. Given that automatic camera orientation and point triangulation are essentially solved - at least for images taken in such a way that they are suitable for dense surface reconstruction - the most challenging step is to find a dense set of corresponding points between different images. The majority of the literature deals with the basic two-view stereo setup (e.g. INTILLE & BOBICK 1994; SCHARSTEIN & SZELISKI 2002; RANFTL et al. 2012; RHEMANN et al. 2011), but obviously using more than two views affords redundancy and is thus beneficial in terms of both accuracy and robustness. More viewpoints also mitigate the occlusion problem and typically yield more complete reconstructions (e.g. CAMPBELL et al. 2008; FURUKAWA & PONCE 2010).

The proposed method is a variant of the *PatchMatch Stereo* algorithm (BLEYER et al. 2011). On one hand, we adapt *PatchMatch* to operate in 3D scene space, such that the method can directly establish correspondence across multiple viewpoints. On the other hand, we modify the algorithm in such a way that it can be parallelised into thousands of computational threads on graphics processing units (GPUs).

[1] ETH Zürich, Photogrammetry and Remote Sensing, Stefano-Franscini-Platz 5, CH-8093 Zürich, E-Mail: [silvano.galliani, katrin.lasinger, schindler]@geod.baug.ethz.ch

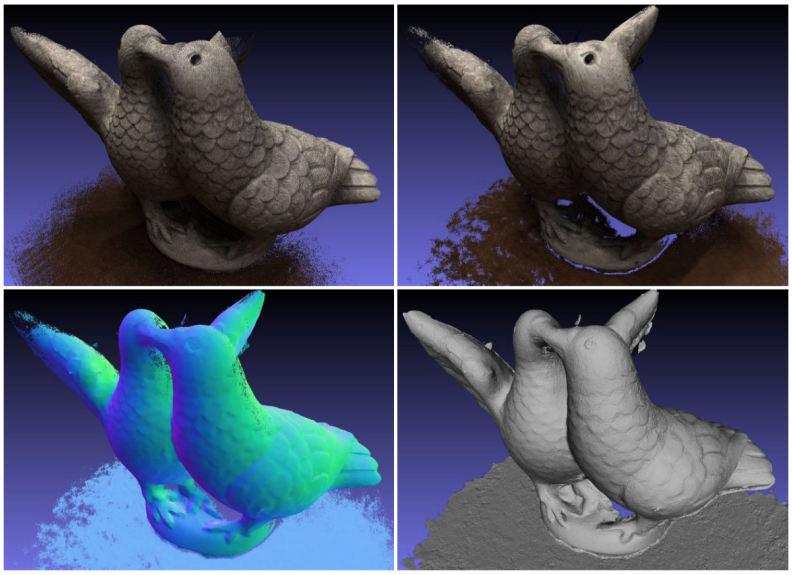

Fig. 1: Example reconstruction. *Top left*: Ground truth point cloud. *Top right*: reconstructed point cloud with texture. *Bottom left*: colour-coded surface normal. *Bottom right*: reconstructed surface.

1.1 Background and Related Work

Dense image matching seeks to establish correspondence densely for all pixels by maximizing photo-consistency. There are two main schools: *local methods*, which derive correspondences only from information in a local neighbourhood, and *global methods*, which formulate an objective function over all pixels. The simplest local method is naive block matching with a window around each pixel. Starting from there, various ideas have been developed to mitigate problems induced by the larger neighbourhood window, in particular the bias towards fronto-parallel surfaces (BLEYER et al. 2011; BURT et al. 1995; DEVERNAY & FAUGERAS 1994; EINECKE & EGGERT 2013; GALLUP et al. 2007), and the fattening of foreground objects along their silhouettes (FUSIELLO et al. 1997; KANADE & OKUTOMI 1994; YOON & KWON 2006). With local methods, smoothness of the reconstructed surface is accounted for only implicitly through the overlap of nearby support windows, or by local filtering of the depth map. Explicit models of smoothness lead to global methods: the correlations between nearby pixels give rise to global objective functions defined over the entire image domain, which can be (approximately) maximised either by discrete labeling (e.g., FELZELSZWALB & HUTTENLOCHER 2006; HIRSCHMÜLLER 2008; KOLMOGOROV & ZABIH 2001) or by variational inference, (e.g. RANFTL et al. 2012). The size of modern digital images calls for matching algorithms with low computational complexity (ideally at most linear in the number of pixels) and low memory footprint, especially in the multi-view case. This has lead to a renewed interest in local methods, and on benchmark datasets like DTU (JENSEN et al. 2014) or KITTI (GEIGER et al. 2012) local matchers have reached competitive results (BLEYER et al. 2011; FURUKAWA & PONCE 2010; RHEMANN et al. 2011).

Multi-view matching goes back to at least (OKUTOMI & KANADE 1993), where matching costs (deviations from photo-consistency) are accumulated over different stereo pairs. Another way to exploit multiple views is to iteratively grow a 3D point cloud from reliable seed points (e.g.,

FURUKAWA & PONCE 2010). Computationally efficient multi-view methods are often based on plane sweeping, i.e. moving a plane through 3D object space along its normal and exhaustively comparing the photo-consistency across multiple views for all possible plane positions (e.g. COLLINS 1996; GALLUP et al. 2007; HU & MORDOHAI 2012).
Another way to classify multi-view stereo methods is by the representation they are based on (SEITZ et al. 2006). The 3D scene can be represented by voxels, level-sets, polygon meshes, or depth maps. We point out that depth maps are a point-wise representation, in the sense that they define a 2.5D point cloud, similar to those generated with scanning devices. The remaining three representations additionally solve (at least implicitly) the step from discrete points to surfaces. Here we aim to reconstruct depth maps and fuse them into 3D point clouds, whereas surface fitting is seen as a post-processing step. If needed, we found that the generic Poisson surface reconstruction method (KAZHDAN & HOPPE 2013) works well with our point clouds, which by construction already include per-point surface normals.

2 Gipuma – Depth Map Computation from Multiple Views

The proposed multi-view matcher is based on the *PatchMatch* (BARNES et al. 2009) principle, which employs iterative propagation and refinement of randomly drawn initial values to quickly find a good solution in a large search space without having to test all possibilities. The resulting low memory footprint makes *PatchMatch Stereo* well-suited for large images or memory-constrained environments, including GPUs (BAILER et al. 2012; BAO et al. 2014; HEISE et al. 2013; Zheng et al. 2014). On the contrary a large portion of the computation time and/or memory footprint of many stereo algorithms is due to the fact that they exhaustively compute, and then compare, the cost for every putative disparity value (e.g. FELZENSZWALB & HUTTENLOCHER 2006; HIRSCHMÜLLER 2008; RHEMANN et al. 2011).

2.1 PatchMatch Stereo

We first briefly describe the original *PatchMatch Stereo* algorithm. That method parameterizes the 3D scene by a local tangent plane per pixel, i.e. at every pixel one stores not only a disparity (horizontal parallax), but also a surface normal. These values are initialised randomly for each pixel. Starting at the top left corner one then loops through the image and tests whether it makes sense to extend the local tangent plane to the lower or right neighbour pixel. If propagating the plane lowers the matching cost, then the corresponding disparity and normal replace the previous values. After the update the plane parameters are refined with fast bisection search. Having reached the lower right corner, the propagation is repeated in the opposite direction. Empirically, only two to three such iterations are needed to converge to a good result.
PatchMatch Stereo computes the photo-consistency in the tangent plane in disparity space, defined by the local disparity and normal, to avoid fronto-parallel bias. The method is compatible with different photo-consistency measures. In difficult lighting conditions the Hamming distance between Census signatures (ZABIH & WOODFILL 1994) is often the best choice. For more controlled lighting we found that the measure recommended in the original *PatchMatch Stereo* paper works very well, namely a linear combination of the absolute intensity difference and the absolute difference in gradient magnitude. The weighted average of these two quantities

(truncated for robustness) is computed separately per pixel and summed over the support window, using individual weights per pixel. These weights reflect the intensity difference to the central pixel of the window, assuming that discontinuities in intensity are likely to coincide with surface discontinuities.

Empirically we found that one can speed up the computation without loss in performance by using only every other row and column in the cost computation, as recommended in the context of the "sparse Census transform" (ZINNER et al. 2008). This is particularly valuable because *PatchMatch* needs larger support windows (typically between 11×11 and 25×25 pixels) than some other stereo methods, in order to reliably estimate the surface normal.

2.2 Multi-view Extension

Disparity is only defined between two images rectified to the photogrammetric normal case. We aim to adapt *PatchMatch* to the multi-view setting. We therefore define the tangent planes per pixel of the reference image in Euclidean scene space rather than in disparity space. This not only avoids epipolar rectification (which is a rather awkward procedure for more than two views), but also ensures that the estimated normal reside in 3D scene space, and can be used for later processing steps such as surface reconstruction (KAZHDAN & HOPPE 2013). Most importantly, working in scene space makes it possible to aggregate matching costs directly across multiple views, by warping all support windows with the same 3D tangent plane with the corresponding plane-induced homographies, see Figure 2. For a given reference image we always accumulate the pairwise matching costs to all stereo partners.

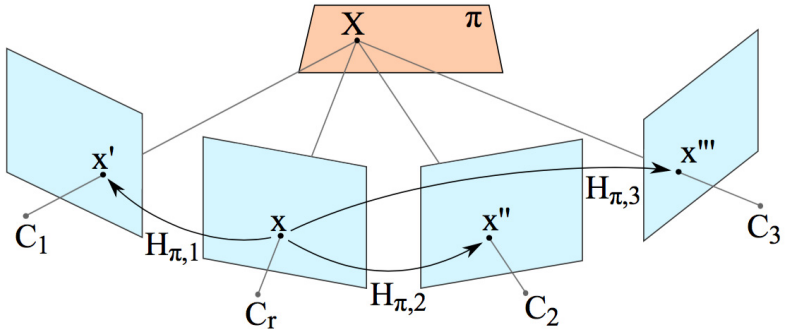

Fig. 2: Multi-view geometry of *Gipuma*: a tangent plane π in 3D scene space defines a planar homography between the reference view C_r and every other view C_i.

A small technical complication when working in scene space is to sample unbiased random normals as initialization. A fast method to pick random vectors that are uniformly distributed over a sphere can be found in (MARSAGLIA 1972). If the sampled normal points away from the camera, we reverse its sign. Moreover, it is better to sample uniformly from the range of possible disparities and convert the result to depth, rather than sampling depth values directly. This will account for the anisotropic depth resolution of stereo reconstruction and provide a finer sampling of depth values in the near field, where they make a difference; and a sparser sampling in the far field, where small variations do not produce an observable difference.

2.3 Surface Normal Diffusion

The original *PatchMatch* propagation scheme described above is inherently sequential. It has been proposed to alternate between row-wise and column-wise propagation (BAILER et al. 2012; BAO et al. 2014; HEISE et al. 2013; ZHENG et al. 2014), still one does not play to the strengths of modern multi-core GPU. Instead, we employ a diffusion-like propagation scheme. The reference image is partitioned into a checkerboard pattern of "red" and "black" pixels, see Figure 3. In this way, one can update all "black" pixels at once by propagating from nearby "red" ones, and vice versa. The red-black scheme is a standard trick to parallelize message-passing and related schemes, c.f. red-black Gauss-Seidel for linear equation solving. Even with this more "local" propagation scheme we find that 6-8 iterations are sufficient for the depth map to converge.

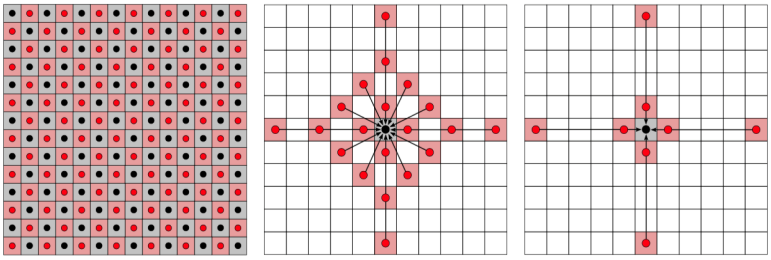

Fig. 3: Red-black scheme for massively parallel propagation. All black pixels are updated in parallel, by propagating tangent planes from the red pixels. *Center*: standard pattern. *Right*: reduced pattern optimised for speed.

3 Fusibile – 3D Point Cloud Generation

Like most multi-view reconstruction methods, we first compute a separate depth map for each view by repeatedly running *Gipuma*. These depth maps are then fused into a 3D point cloud, using the redundancy to eliminate blunders and improve accuracy. Given multiple depth maps covering the same object surface, most mismatches can be detected: if a 3D point is wrong due to a lack of texture or due to occlusion, then it is unlikely that it is consistent with the measurement in another view (even if that measurement is wrong, too). To filter out such cases, we visit each depth map in turn, convert the depth measurements to 3D points and project those points back to the other views. If the projection is not consistent with the observed disparity and normal in at least two other views, the point is removed. For those points, which are retained, the 3D positions and normals are averaged over the consistent views to further suppress noise.

The thresholds that determine whether a point's reprojection is consistent enough to be added to the point cloud can be tuned, to achieve more accurate or more complete reconstructions. The right trade-off depends on the application, for example computer graphics typically needs complete models that "look correct", whereas industrial metrology prefers high accuracy even at the cost of a sparser reconstruction. We test different options in the experiments. Since the fusion step always uses the same depth maps it is fast (≈15 seconds for 50 depth maps of size 1600 × 1200), so one can even interactively explore different possibilities.

4 Experiments

As a main dataset for quantitative evaluation we use the DTU large scale multi-view dataset (JENSEN et al. 2014). It consists of 80 different objects, each covered by 49-64 images of resolution 1600 × 1200 pixels. The scenes vary in reflectance, texture and geometry, see Figure 4. The images have been acquired with a robot arm to accurately position the cameras. Ground truth is captured by a structured light scanner. The error metrics defined by the authors of the dataset are mean and median reconstruction errors, calculated once for the 3D point cloud and once for a surface mesh derived from the points. Accuracy is defined as the distance from the estimated surface to the ground truth, and completeness as the distance from the ground truth to the surface (both measured in mm).

To find suitable stereo partners, we select images with reasonable angular baseline: for a given reference image, we chose all other views whose viewing directions differ by at least 10 and at most 30 degrees from the reference view. *Gipuma* in its high-accuracy setting achieves the best accuracy among the tested methods while at the same time delivering the second-highest completeness, see Table 1. When tuned for completeness, our method achieves the highest completeness at the second-highest accuracy. The runtime is ≈50 seconds per depth map with 9 stereo partners. We also test an extreme setting tuned for high speed (window size 15 instead of 25, 6 iterations instead of 8, at most 9 stereo partners per view). With these settings it takes ≈2.7 seconds to compute a depth map, respectively 7 minutes for a complete reconstruction, (including disk I/O). Even this version is competitive with the state of the art, see Table 1.

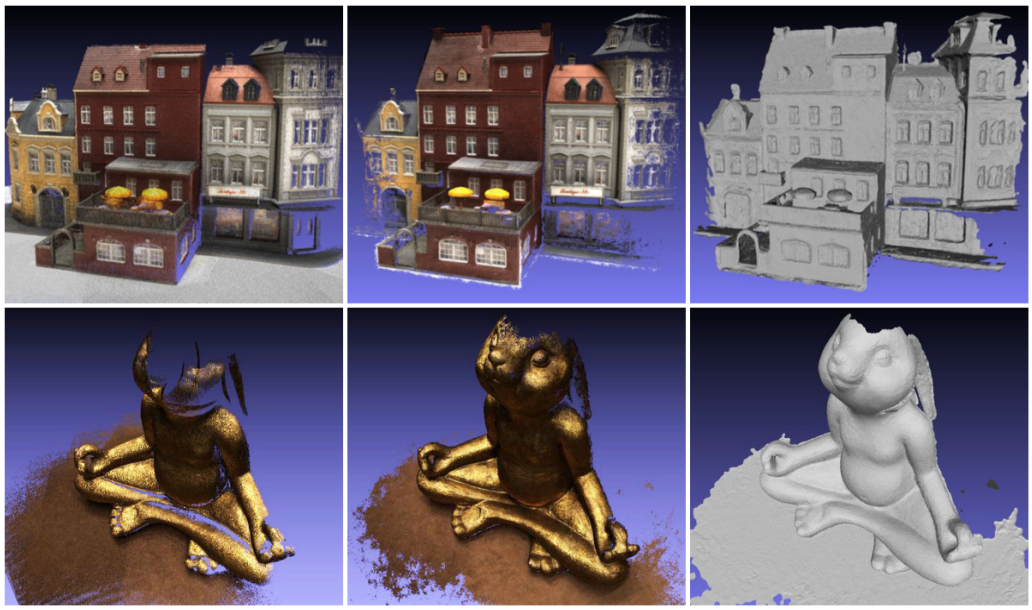

Fig. 4: Example objects from DTU dataset. *Left*: ground truth point cloud. *Center*: reconstructed point cloud. *Right*: reconstructed surface mesh.

We further evaluate *Gipuma* on the popular Middlebury benchmark (//vision.middlebury.edu/), although we note that it is by now rather saturated, and somewhat outdated (only 2 objects, image size 640 × 480 pixels). Our results rank second for the "full" version (363 views) of "Dino" and fifth for the "full" version (312 views) of the "Temple". For the smaller "ring"

(48/47 views) and "sparse" (16 views) versions of both objects we are also always in the top 8 out of ca. 50 submitted results. To generate a depth map from 10 views our method needs 2.5 seconds. Finally, we have also run *Gipuma* on oblique aerial images and on KITTI (cameras mounted in a vehicle), see qualitative examples in Figure 5.

Table 1: Quantitative comparisons on DTU data. Mean and median errors in mm (lower is better).

points	accuracy		completeness	
	mean	median	mean	median
Gipuma acc.	**0.273**	**0.196**	0.687	0.260
Gipuma comp.	0.379	0.234	**0.400**	0.188
Gipuma fast	0.289	0.207	0.841	0.285
TOLA et al. 2010	0.307	0.198	1.097	0.456
FURUKAWA & PONCE 2010	0.605	0.321	0.842	0.431
CAMPBELL et al. 2008	0.753	0.480	0.540	**0.179**

surfaces	accuracy		completeness	
	mean	median	mean	median
Gipuma acc.	**0.363**	**0.215**	0.766	0.329
Gipuma comp.	0.631	0.262	**0.519**	**0.309**
Gipuma fast	0.358	0.221	0.939	0.350
TOLA et al. 2010	0.488	0.244	0.974	0.382
FURUKAWA & PONCE 2010	1.299	0.534	0.702	0.405
CAMPBELL et al. 2008	1.411	0.579	0.562	0.322

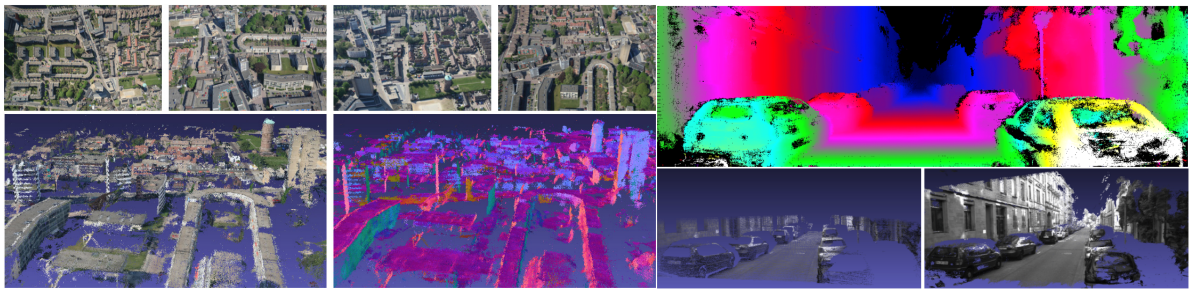

Fig. 5: Further *Gipuma*/*Fusibile* results. *Left*: Oblique aerial images. *Right*: Vehicle-mounted cameras from KITTI.

5 Conclusion

The *Gipuma*/*Fusibile* multi-view stereo suite has been designed to provide high-quality point clouds from multi-view imagery. It is optimised for parallel computation on off-the-shelf GPUs. Technically, the system is a multi-view extension of *PatchMatch Stereo* tailored to modern GPUs with thousands of parallel threads. Like the original *PatchMatch* it uses slanted support windows and does not suffer from fronto-parallel bias. It is thus particularly well-suited for locally smooth scenes with a large extent in depth. Quantitative results on the DTU and Middlebury benchmarks confirm that the software is at the same time more accurate than state-of-the-art methods such as PMVS and much faster. *Gipuma* and *Fusibile* are available as open-source software at https://github.com/kysucix/gipuma, respectively https://github.com/kysucix/fusibile.

6 Bibliography

BAILER, C., FINCKH, M. & LENSCH, H.P., 2012: Scale robust multi view stereo. European Conference on Computer Vision (ECCV), Springer Berlin Heidelberg, 398-411.

BAO, L., YANG, Q. & JIN, H., 2014: Fast edge-preserving Patchmatch for large displacement optical flow. IEEE Conference on Computer Vision and Pattern Recognition, 3534-3541.

BARNES, C., SHECHTMAN, E., FINKELSTEIN, A. & GOLDMAN, D.B., 2009: PatchMatch: A randomized correspondence algorithm for structural image editing. ACM Transactions on Graphics **28** (3), 24.

BLEYER, M., RHEMANN, C. & ROTHER, C., 2011: PatchMatch Stereo - stereo matching with slanted support windows. British Machine Vision Conference (BMVC) **11**, 1-11.

BURT, P., WIXSON, L. & SALGIAN, G., 1995: Electronically directed ”focal” stereo. 5th International Conference on Computer Vision (ICCV), 94-101.

CAMPBELL, N.D., VOGIATZIS, G., HERNÁNDEZ, C. & CIPOLLA, R., 2008: Using multiple hypotheses to improve depth-maps for multi-view stereo. European Conference on Computer Vision (ECCV), 766-779.

COLLINS, R.T., 1996: A space-sweep approach to true multi-image matching. Computer Vision and Pattern Recognition (CVPR), 358-363.

DEVERNAY, F. & FAUGERAS, O., 1994: Computing differential properties of 3-d shapes from stereoscopic images without 3-d models. Computer Vision and Pattern Recognition (CVPR), 208-213.

EINECKE, N. & EGGERT, J., 2013: Stereo image warping for improved depth estimation of road surfaces. Intelligent Vehicle Symposium (IVS), 189-194.

FELZENSZWALB, P.F. & HUTTENLOCHER, D. P., 2006: Efficient belief propagation for early vision. International Journal of Comuter Vision, **70**(1), 41-54.

FURUKAWA, Y. & PONCE, J., 2010: Accurate, dense, and robust multiview stereopsis. IEEE Transactions on Pattern Analysis and Machine Intelligence, **32**(8), 1362-1376.

FUSIELLO, A., ROBERTO, V. & TRUCCO, E., 1997: Efficient stereo with multiple windowing. Computer Vision and Pattern Recognition (CVPR), 858.

GALLUP, D., FRAHM, J.-M., MORDOHAI, P., YANG, Q. & POLLEFEYS, M., 2007: Real-time plane-sweeping stereo with multiple sweeping directions. Computer Vision and Pattern Recognition (CVPR), 1-8.

GEIGER, A., LENZ, P. & URTASUN, R., 2012: Are we ready for autonomous driving? The KITTI vision benchmark suite. Computer Vision and Pattern Recognition (CVPR), 3354-3361.

HEISE, P., KLOSE, S., JENSEN, B. & KNOLL, A., 2013: PM-Huber: PatchMatch with Huber regularization for stereo matching. International Conference on Computer Vision (ICCV), 2360-2367.

HIRSCHMÜLLER, H., 2008: Stereo processing by semi-global matching and mutual information. IEEE Transactions on Pattern Analysis and Machine Intelligence **30**(2), 328-341.

HU, X. & MORDOHAI, P., 2012: Least commitment, viewpoint-based, multi-view stereo. 3D Imaging, Modeling, Processing, Visualization and Transmission (3DIMPVT), 531-538.

INTILLE, S.S. & BOBICK, A. F., 1994: Disparity-space images and large occlusion stereo. European Conference on Computer Vision (ECCV), Springer Berlin Heidelberg, 179-186.

JENSEN, R., DAHL, A., VOGIATZIS, G., TOLA, E. & AANÆS, H., 2014: Large scale multi-view stereopsis evaluation. Computer Vision and Pattern Recognition (CVPR), 406-413.

KANADE, T. & OKUTOMI, M., 1994: A stereo matching algorithm with an adaptive window: Theory and experiment. IEEE Transactions on Pattern Analysis and Machine Intelligence **16** (9), 920-932.

KAZHDAN, M. & HOPPE, H., 2013: Screened Poisson surface reconstruction. ACM Transactions on Graphics **32** (3), 29.

KOLMOGOROV, V. & ZABIH, R., 2001: Computing visual correspondence with occlusions using graph cuts. International Conference on Computer Vision (ICCV), 508-515.

MARSAGLIA, G., 1972: Choosing a point from the surface of a sphere. Annals of Mathematical Statistics **43** (2), 645-646.

OKUTOMI, M. & KANADE, T., 1993: A multiple-baseline stereo. IEEE Transactions on Pattern Analysis and Machine Intelligence **15** (4), 353-363.

RANFTL, R., GEHRIG, S., POCK, T. & BISCHOF, H., 2012: Pushing the limits of stereo using variational stereo estimation. Intelligent Vehicles Symposium (IVS), 401-407.

RHEMANN, C., HOSNI, A., BLEYER, M., ROTHER, C. & GELAUTZ, M., 2011: Fast cost-volume filtering for visual correspondence and beyond. Computer Vision and Pattern Recognition (CVPR), 3017-3024.

SCHARSTEIN, D. & SZELISKI, R., 2002: A taxonomy and evaluation of dense two-frame stereo correspondence algorithms. International Journal of Computer Vision **47** (1-3), 7-42.

SEITZ, S.M., CURLESS, B., DIEBEL, J., SCHARSTEIN, D. & SZELISKI, R., 2006: A comparison and evaluation of multi-view stereo reconstruction algorithms. Computer Vision and Pattern Recognition (CVPR), 519-528.

TOLA, E., LEPETIT, V. & FUA, P., 2010: Daisy: An efficient dense descriptor applied to wide-baseline stereo. IEEE Transactions on Pattern Analysis and Machine Intelligence **32** (5), 815-830.

YOON, K.-J. & KWEON, I.S., 2006: Adaptive support-weight approach for correspondence search. IEEE Transactions on Pattern Analysis and Machine Intelligence **28** (4), 650-656.

ZABIH, R. & WOODFILL, J., 1994: Non-parametric local transforms for computing visual correspondence. European Conference on Computer Vision (ECCV), Springer Berlin Heidelberg, 151-158.

ZHENG, E., DUNN, E., JOJIC, V. & FRAHM, J.-M., 2014: PatchMatch based joint view selection and depthmap estimation. Computer Vision and Pattern Recognition (CVPR), 1510-1517.

ZINNER, C., HUMENBERGER, M., AMBROSCH, K. & KUBINGER, W., 2008: An optimized software-based implementation of a census-based stereo matching algorithm. Advances in Visual Computing, 216-227.