StereoScan: Dense 3d Reconstruction in Real-time

Andreas Geiger, Julius Ziegler and Christoph Stiller
Department of Measurement and Control
Karlsruhe Institute of Technology
{geiger,ziegler,stiller}@kit.edu

Abstract— **Accurate 3d perception from video sequences is a core subject in computer vision and robotics, since it forms the basis of subsequent scene analysis. In practice however, online requirements often severely limit the utilizable camera resolution and hence also reconstruction accuracy. Furthermore, real-time systems often rely on heavy parallelism which can prevent applications in mobile devices or driver assistance systems, especially in cases where FPGAs cannot be employed.**

This paper proposes a novel approach to build 3d maps from high-resolution stereo sequences in real-time. Inspired by recent progress in stereo matching, we propose a sparse feature matcher in conjunction with an efficient and robust visual odometry algorithm. Our reconstruction pipeline combines both techniques with efficient stereo matching and a multi-view linking scheme for generating consistent 3d point clouds. In our experiments we show that the proposed odometry method achieves state-of-the-art accuracy. Including feature matching, the visual odometry part of our algorithm runs at 25 frames per second, while – at the same time – we obtain new depth maps at 3-4 fps, sufficient for online 3d reconstructions.

I. Introduction

Today, laser scanners are still widely used in robotics and autonomous vehicles, mainly because they directly provide 3d measurements in real-time. However, compared to traditional camera systems, 3d laser scanners are often more expensive and more difficult to seamlessly integrate into existing hardware designs (e.g., cars or trains). Moreover, they easily interfere with other sensors of the same type as they are based active sensing principles. Also, their vertical resolution is limited (e.g., 64 laser beams in the Velodyne HDL-64E). Classical computer vision techniques such as appearance-based object detection and tracking are hindered by the large amount of noise in the reflectance measurements.

Motivated by those facts and the emergent availability of high-resolution video sensors, this paper proposes a novel system enabling accurate 3d reconstructions of static scenes, solely from stereo sequences[1]. To the best of our knowledge, ours is the first system which is able to process images of approximately one Megapixel resolution online on a single CPU. Our contributions are threefold: First, we demonstrate real-time scene flow computation with several thousand feature matches. Second, a simple but robust visual odometry algorithm is proposed, which reaches significant speed-ups compared to current state-of-the-art. Finally, using the obtained ego-motion, we integrate dense stereo measurements from *LIBELAS* [12] at a lower frame rate and solve the associated correspondence problem in a greedy fashion, thereby increasing accuracy while still maintaining efficiency. Fig. 1 illustrates the input to our system and resulting live 3d reconstructions on a toy example.

[1]Source code available from: www.cvlibs.net

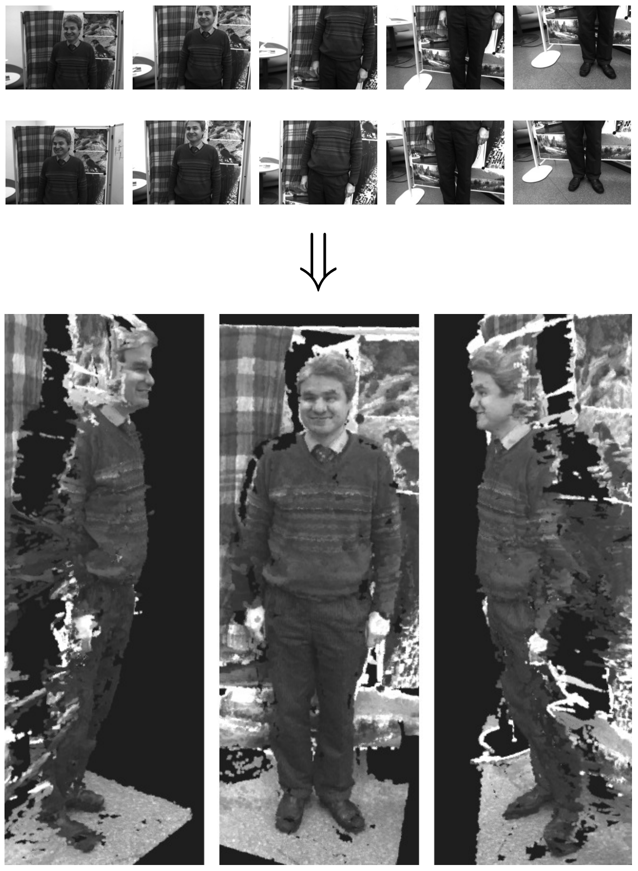


Fig. 1. **Real-time 3d reconstruction based on stereo sequences:** Our system takes stereo images (top) as input and outputs an accurate 3d model (bottom) in real-time. All processing is done on a single CPU.

II. Related Work

As video-based 3d reconstruction is a core topic in computer vision and robotics, there exists a large body of related work, sketched briefly in the following:

Simultaneous Localisation and Mapping (SLAM) [6], [19], [7], [26], [5] is the process by which a mobile robot incrementally builds a consistent map of its environment and at the same time uses this map to compute its own location. However, for computational reasons, most of the

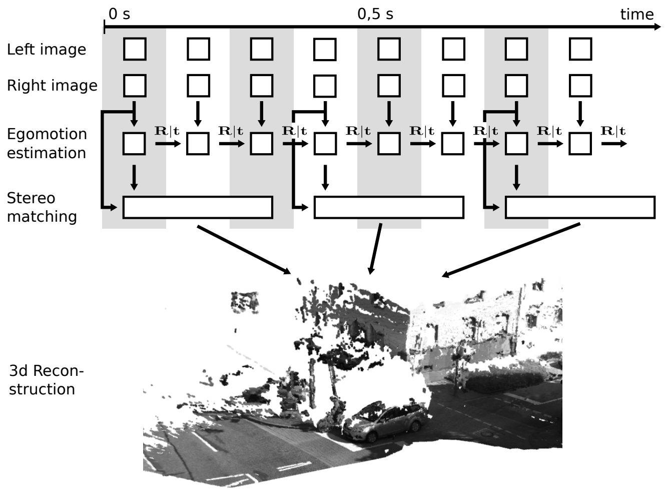


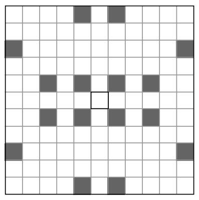
Fig. 2. **System overview:** We use two worker threads in order to obtain egomotion estimates and disparity maps in parallel: While the stereo matching and 3d reconstruction part runs at 3-4 fps (Sec. III C+D), our sparse feature matching and visual odometry system (Sec. III A+B) achieves 25 fps. Taken together, this is sufficient for online 3d reconstruction.

-1	-1	-1	-1	-1
-1	+1	+1	+1	-1
-1	+1	+8	+1	-1
-1	+1	+1	+1	-1
-1	-1	-1	-1	-1

(a) Blob detector

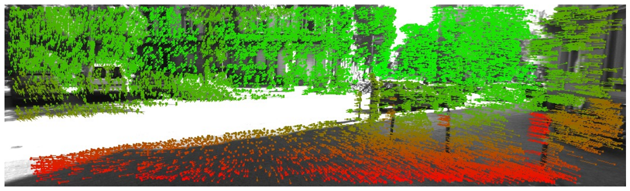
-1	-1	0	+1	+1
-1	-1	0	+1	+1
0	0	0	0	0
+1	+1	0	-1	-1
+1	+1	0	-1	-1

(b) Corner detector

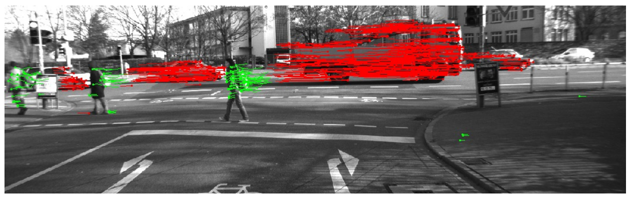


(c) Feature descriptor

Fig. 3. **Blob/corner detector and feature descriptor:** Our feature detections are minima and maxima of blob and corner filter responses. The descriptor concatenates Sobel filter responses using the layout given in (c).



(a) Feature matching (2 frames, moving camera)



(b) Feature tracking (5 frames, static camera)

Fig. 4. **Feature matching:** (a) Matching features in a circle, colors encode disparities. (b) Feature tracking, colors encode track orientation.

proposed approaches are only able to handle very sparse sets of landmarks in real-time, while here we are interested in a dense mapping solution.

With the seminal work by Hoiem et al. [14] learning-based approaches to geometry estimation from monocular images have seen a revival [23], [13]. Those methods typically segment images into superpixels and, based on local appearance as well as global constraints, infer the most likely 3d configuration of each segment. Even though impressive results have been demonstrated recently [13], those methods are still too inaccurate and erroneous to directly support applications like mobile navigation or autonomous driving.

3d reconstruction from uncalibrated image collections has been shown by Koch [17], Pollefeys [22], Seitz [24] et al. using classical Structure-from-Motion (SfM) techniques. Extensions to urban reconstruction have been demonstrated in [2], [9], [10]. More recently, the availability of photo sharing platforms like Flickr led to efforts of modeling cities as large as Rome [1], [8]. However, in order to obtain accurate semi-dense reconstructions, powerful multi-view stereo schemes are employed, which, even on small image collections, easily take up to several hours while making extensive use of parallel processing devices. Further, most of the proposed methods require several redundant viewpoints, while our application target is a continuously moving mobile platform, where objects can be observed only over short periods of time.

In [3] Badino et al. introduces *Stixel World* as medium-level representation to reduce the amount of incoming sensor information. They observe that free space in front of a vehicle is usually limited by objects with vertical surfaces, and represent those by adjacent rectangular sticks of fixed width, which are tracked over time [21]. Another kind of frequently employed mid-level representations are occupancy grids [15], [18], which discretize the 3d world into binary 2d cells. Though useful in many applications, those types of abstractions are not detailed enough to represent curbstones or overhanging objects such as trees, signs or traffic lights. Alternatively, 3d voxel grids can be employed. However, without waiving resolution, computational complexity increases dramatically. In this paper instead, we are interested in representing the perceived information as detailed as possible, but without losing real-time performance.

III. 3D Reconstruction Pipeline

Our 3d reconstruction pipeline consists of four stages: Sparse feature matching, egomotion estimation, dense stereo matching and 3d reconstruction. We assume two cores of a CPU available, such that two threads can carry out work in parallel: As illustrated in Fig. 2 the first worker thread performs feature matching and egomotion estimation at 25 fps, while the second thread performs dense stereo matching and 3d reconstruction at 3 to 4 fps. As we show in our experiments, this is sufficient for online 3d reconstruction of static scenes. In the following we will assume a calibrated stereo setup and rectified input images, as this represents the standard case and simplifies computations.

A. Feature Matching

The input to our visual odometry algorithm are features matched between four images, namely the left and right images of two consecutive frames. In order to find stable feature locations, we first filter the input images with 5×5

blob and corner masks, as given in Fig. 3. Next, we employ non-maximum- and non-minimum-suppression [20] on the filtered images, resulting in feature candidates which belong to one of four classes (i.e., blob max, blob min, corner max, corner min). To reduce computational efforts, we only match features within those classes.

In contrast to methods concerned with reconstructions from unordered image collections, here we assume a smooth camera trajectory, superseding computationally intense rotation and scale invariant feature descriptors like SURF [4]. Given two feature points, we simply compare 11×11 block windows of horizontal and vertical Sobel filter responses to each other by using the sum of absolute differences (SAD) error metric. To speed-up matching, we quantize the Sobel responses to 8 bits and sum the differences over a sparse set of 16 locations (see Fig. 3(c)) instead of summing over the whole block window. Since the SAD of 16 bytes can be computed efficiently using a single SSE instruction we only need two calls (for horizontal + vertical Sobel responses) in order to evaluate this error metric.

Our egomotion estimation mechanism expects features to be matched between the left and right images and two consecutive frames. This is achieved by matching features in a 'circle': Starting from all feature candidates in the *current left* image, we find the best match in the *previous left* image within a $M \times M$ search window, next in the *previous right* image, the *current right* image and last in the *current left* image again. A 'circle match' gets accepted, if the last feature coincides with the first feature. When matching between the left and right images, we additionally make use of the epipolar constraint using an error tolerance of 1 pixel. Sporadic outliers are removed by establishing neighborhood relations as edges of a 2d Delaunay triangulation [25] on the feature locations in the current left image. We only retain matches which are supported by at least two neighboring matches, where a match is supporting another match, if its disparity and flow differences fall within some threshold τ_{disp} or τ_{flow} respectively. If required, sub-pixel refinement via parabolic fitting can be employed to further improve feature localization.

Even though our implementation is very efficient, establishing several *thousands* to *ten thousands* of correspondences still takes time in the order of seconds, hence making it too slow for online applications. By transferring ideas already employed in previous works on stereo matching [12], further significant speed-ups are possible: In a first pass, we match only a subset of all features, found by non-maxima-suppression (NMS) using a larger NMS neighborhood size (factor 3). Since this subset is much smaller than the full feature set, matching is very fast. Next, we assign each feature in the current left image to a 50×50 pixel bin of an equally spaced grid. Given all sparse feature matches, we compute the minimum and maximum displacements for each bin. Those statistics are used to locally narrow down the final search space, leading to faster matching and a higher number of matches at the same time, as evidenced in the experimental section. Fig. 4 illustrates feature matching and tracking results using our method.

B. Egomotion Estimation

Given all 'circular' feature matches from the previous section, we compute the camera motion by minimizing the sum of reprojection errors and refining the obtained velocity estimates by means of a Kalman filter.

First, bucketing is used to reduce the number of features (in practice we retain between 200 and 500 features) and spread them uniformly over the image domain. Next, we project feature points from the previous frame into 3d via triangulation using the calibration parameters of the stereo camera rig. Assuming squared pixels and zero skew, the reprojection into the current image is given by

$$\begin{pmatrix} u \\ v \\ 1 \end{pmatrix} = \begin{pmatrix} f & 0 & c_u \\ 0 & f & c_v \\ 0 & 0 & 1 \end{pmatrix} \left[\begin{pmatrix} \mathbf{R}(\mathbf{r}) & \mathbf{t} \end{pmatrix} \begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix} - \begin{pmatrix} s \\ 0 \\ 0 \end{pmatrix} \right] \quad (1)$$

with

- homogeneous image coordinates $(u\ v\ 1)^T$
- focal length f
- principal point (c_u, c_v)
- rotation matrix $\mathbf{R}(\mathbf{r}) = \mathbf{R}_x(r_x)\mathbf{R}_y(r_y)\mathbf{R}_z(r_z)$
- translation vector $\mathbf{t} = (t_x\ t_y\ t_z)^T$
- 3d point coordinates $\mathbf{X} = (x\ y\ z)^T$
- and shift $s = 0$ (left image), $s =$ baseline (right image).

Let now $\pi^{(l)}(\mathbf{X}; \mathbf{r}, \mathbf{t}) : \mathbb{R}^3 \rightarrow \mathbb{R}^2$ denote the projection implied by Eq. 1, which takes a 3d point $\mathbf{X}$ and maps it to a pixel $\mathbf{x}_i^{(l)} \in \mathbb{R}^2$ on the left image plane. Similarly, let $\pi^{(r)}(\mathbf{X}; \mathbf{r}, \mathbf{t})$ be the projection onto the right image plane. Using Gauss-Newton optimization, we iteratively minimize

$$\sum_{i=1}^{N} \left\| \mathbf{x}_i^{(l)} - \pi^{(l)}(\mathbf{X}_i; \mathbf{r}, \mathbf{t}) \right\|^2 + \left\| \mathbf{x}_i^{(r)} - \pi^{(r)}(\mathbf{X}_i; \mathbf{r}, \mathbf{t}) \right\|^2 \quad (2)$$

with respect to the transformation parameters $(\mathbf{r}, \mathbf{t})$. Here $\mathbf{x}_i^{(l)}$ and $\mathbf{x}_i^{(r)}$ denote the feature locations in the current left and right images respectively. The required Jacobians $J_{\pi^{(l)}}$ and $J_{\pi^{(r)}}$ are readily derived from Eq. 1. In practice we note that even if we initialize $\mathbf{r}$ and $\mathbf{t}$ to $\mathbf{0}$, a couple of iterations (e.g., 4-8) are sufficient for convergence. To be robust against outliers, we wrap our estimation approach into a RANSAC scheme, by first estimating $(\mathbf{r}, \mathbf{t})$ for 50 times independently using 3 randomly drawn correspondences. All inliers of the winning iteration are then used for refining the parameters, yielding the final transformation $(\mathbf{r}, \mathbf{t})$.

On top of this simple, but efficient estimation procedure we place a standard Kalman filter, assuming constant acceleration. To this end, we first obtain the velocity vector $\mathbf{v} = (\mathbf{r}\ \mathbf{t})^T / \Delta_t$ as the transformation parameters divided by the time between frames Δ_t. The state equation is given by

$$\begin{pmatrix} \mathbf{v} \\ \mathbf{a} \end{pmatrix}^{(t)} = \begin{pmatrix} \mathbf{I} & \Delta_t \mathbf{I} \\ \mathbf{0} & \mathbf{I} \end{pmatrix} \begin{pmatrix} \mathbf{v} \\ \mathbf{a} \end{pmatrix}^{(t-1)} + \boldsymbol{\epsilon} \quad (3)$$

and the output equation reduces to

$$\frac{1}{\Delta_t} \begin{pmatrix} \mathbf{r} \\ \mathbf{t} \end{pmatrix}^{(t)} = \begin{pmatrix} \mathbf{I} & \mathbf{0} \end{pmatrix} \begin{pmatrix} \mathbf{v} \\ \mathbf{a} \end{pmatrix}^{(t)} + \boldsymbol{\nu} \quad (4)$$

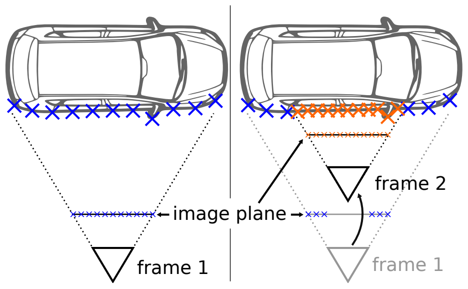
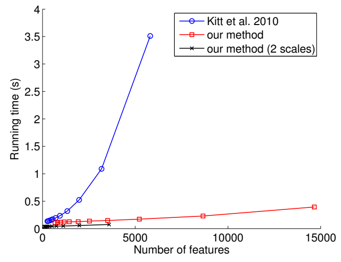
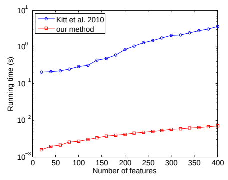


Fig. 5. **Multi-view reconstruction:** In order to fuse 3d points we greedily associate them by reprojection into the image plane of the current frame.



Stage	Time
Filter	6.0 ms
NMS	12 ms
Matching 1	2.8 ms
Matching 2	10.7 ms
Refinement	5.1 ms
Total time	36.6 ms

(a) Feature matching



Stage	Time
RANSAC	3.8 ms
Refinement	0.4 ms
Kalman filter	0.1 ms
Total time	4.3 ms

(b) Visual odometry

Fig. 6. **Sparse feature matching and visual odometry running times.** The tables show timings for individual parts of our algorithm when parameterized to the online settings (2 scales, NMS neighborhood 3 corresponding to approximately 500 to 2000 feature matches and 50 RANSAC iterations). For timings of the stereo matching stage we refer the reader to [12].

since we directly observe $\mathbf{v}$. Here, $\mathbf{a}$ denotes acceleration, $\mathbf{I}$ is the 6×6 identity matrix and $\boldsymbol{\epsilon}$, $\boldsymbol{\nu}$ represent Gaussian process and measurement noise, respectively.

C. Stereo Matching

For obtaining dense disparity maps, we use a method called *ELAS* [12], which is freely available. *ELAS* is a novel approach to binocular stereo for fast matching of high-resolution imagery. It builds a prior on the disparities by forming a triangulation on a set of support points which can be robustly matched, reducing matching ambiguities of the remaining points. This allows for efficient exploitation of the disparity search space, yielding accurate and dense reconstructions without global optimization. Also, the method automatically determines the required disparity search range, which is important for outdoor scenarios. *ELAS* achieves state-of-the-art performance on the large-scale Middlebury benchmark while being significantly faster than existing methods: For the outdoor sequences used in our experiments (≈ 0.5 Megapixel resolution), we obtain $3-4$ frames per second on a single i7 CPU core running at 3.0 GHz.

D. 3d Reconstruction

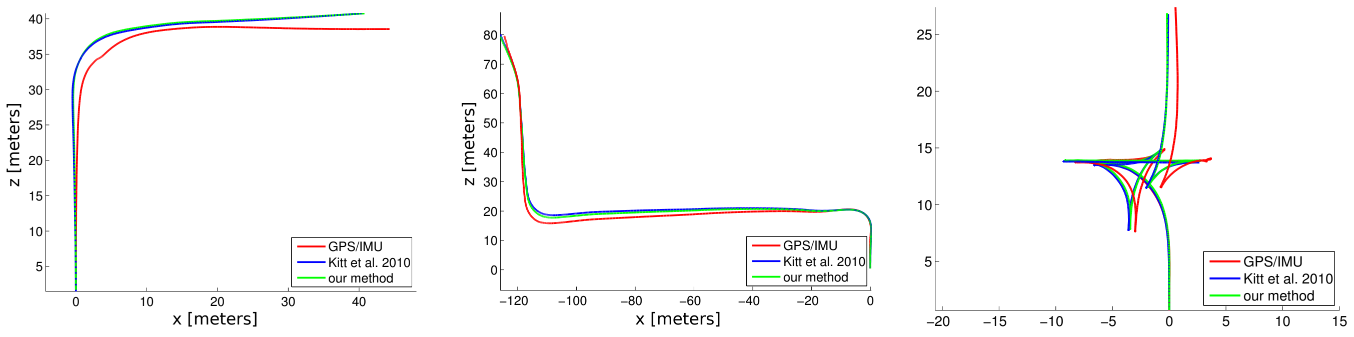
The last step in our reconstruction pipeline creates consistent point-based models from the large amount of incoming data (i.e., $\approx$ 500.000 points, $3-4$ times every second). The simplest method to point-based 3d reconstructions maps all valid pixels to 3d and projects them into a common coordinate system according to the estimated camera motion. However, without solving the association problem, storage requirements will grow rapidly. Further, redundant information can not be used for improving reconstruction accuracy. On the other hand, traditional multi-view optimizations such as bundle adjustment are computationally infeasible for dense real-time systems as the proposed one.

Instead, here we propose a greedy approach which solves the association problem by reprojecting reconstructed 3d points of the previous frame into the image plane of the current frame. In case a point falls onto a valid disparity, we fuse both 3d points by computing their 3d mean. This does not only dramatically reduce the number of points which have to be stored, but also leads to improved accuracy by averaging out measurement noise over several frames. Fig. 5 illustrates our approach for two frames: Parts of the points captured in frame one (blue) get fused with points captured in frame two (orange), after the camera underwent a forward movement. Our method only involves projections and pointer book keeping and hence can be implemented very efficiently: Appending a single disparity map to the 3d model typically takes less than 50 ms, hence only adding minor computations to the stereo matching and reconstruction thread.

Since our reconstructed sequences are relatively short, we do not consider the 'soft reset problem' in this paper. However, a simple solution would be to remove all 3d points associated with depth maps of 'outdated' poses.

IV. Experimental Results

In this section we compare our results to [16], a freely available visual odometry library. All experiments were performed on image sequences of the Karlsruhe dataset (www.cvlibs.net), which provides ground truth GPS+IMU data as well as stereo sequences at a resolution of 1344×391 pixels and 10 fps. Our real-time parameterization uses the standard settings for *LIBELAS* stereo matching [12], and sparse feature matching at 2 scales with 50 RANSAC iterations, an inlier threshold of 1.5 pixels and $\tau_{disp} = \tau_{flow} = 5$ pixels. For egomotion estimation, we empirically set the measurement noise parameters of the Kalman filter to $\boldsymbol{\nu} \sim \mathcal{N}(0, 10^{-2} \times \mathbf{I})$, $\boldsymbol{\epsilon}_{1..6} \sim \mathcal{N}(0, 10^{-8} \times \mathbf{I})$ and $\boldsymbol{\epsilon}_{7..12} \sim \mathcal{N}(0, \mathbf{I})$.



(a) 2009_09_08_drive_0010 (b) 2009_09_08_drive_0016 (c) 2009_09_08_drive_0021

Fig. 7. **Visual odometry results** on the Karlsruhe data set. Best viewed in color.

A. *Feature matching*

Fig. 6(a) illustrates sparse feature matching running times over the number of matched features. We compare the proposed method to a version evaluated at half-size resolution and refined at full resolution, and to the baseline by Kitt et al. [16]. We observe that our multi-stage matching approach helps in reducing running times significantly, while at the same time increasing the number of feature matches. This beneficial behaviour is mainly due to reduced ambiguities in the second matching stage. Also, note that a two scale matching approach, which computes features at half resolution and refines them at full resolution, further reduces running time, while preserving a reasonable amount of feature matches for visual odometry (500-2000): Using this setting we are able to achieve feature matching over 25 fps on a single CPU core, as shown in the table, which lists running times of individual parts ouf our algorithm.

B. *Visual Odometry*

As evidenced by Fig. 6(b), we are also able to cut visual odometry running times significantly with respect to the *CVMLIB*-based version of the algorithm presented in [16]. While Kitt et al. requires about one second to process 200 feature matches, 4.3 milliseconds are sufficient for our method, leading to speed ups of more than factor 200. This is mainly due to the relatively complex nature of the observation model employed in [16], which is based on trifocal tensors and requires inverting matrices growing linearly with the number of matched features, while our matrix inversions are constant in this number. In Fig. 7 we further compare our visual odometry trajectories to Kitt et al. and 'ground truth' output of a OXTS RT 3003 GPS/IMU system on the Karlsruhe dataset. Even though running much faster, our method achieves localization accuracy comparable to [16]. Please note that the GPS/IMU system can only be considered as 'weak' ground truth, because localization errors of up to two meters may occur in inner-city scenarios due to limited satellite availability.

C. *3d Reconstruction*

We also qualitatively evaluate our complete reconstruction pipeline. Fig. 8 illustrates 3d reconstructions obtained by our system for three different sequences (rows) and from four different viewpoints (columns).

V. Conclusion

In this paper we have demonstrated a system to generate accurate dense 3d reconstructions from stereo sequences. Compared to existing methods, we were able to reduce running times of feature matching and visual odometry by more than one or two orders of magnitude, respectively, allowing real-time 3d reconstructions from large-scale imagery on the CPU. We believe our system will be valuable for other researchers working on higher-level reasoning for robots and intelligent vehicles. In the future we intend to combine our visual odometry system with GPS/INS systems to reduce localization errors in narrow urban scenarios with restricted satellite reception. We also plan on handling dynamic objects and on using our maps for 3d scene understanding at road intersections [11].

VI. Acknowledgements

We thank the reviewers for their feedback and the *Karlsruhe School of Optics and Photonics* for financial support.

References

[1] S. Agarwal, N. Snavely, I. Simon, S. M. Seitz, and R. Szeliski, "Building rome in a day," in *ICCV*, 2009.

[2] A. Akbarzadeh, J. M. Frahm, P. Mordohai, B. Clipp, C. Engels, D. Gallup, P. Merrell, M. Phelps, S. Sinha, B. Talton, L. Wang, Q. Yang, H. Stewenius, R. Yang, G. Welch, H. Towles, D. Nister, and M. Pollefeys, "Towards urban 3d reconstruction from video," in *3DPVT*, 2006, pp. 1–8.

[3] H. Badino, U. Franke, and D. Pfeiffer, "The stixel world - a compact medium level representation of the 3d-world," in *DAGM*, 2009, pp. 51–60.

[4] H. Bay, T. Tuytelaars, and L. V. Gool, "Surf: Speeded up robust features," in *ECCV*, 2006.

[5] A. J. Davison, I. D. Reid, N. D. Molton, and O. Stasse, "Monoslam: Real-time single camera slam," *PAMI*, vol. 29, no. 6, pp. 1052–1067, 2007.

[6] M. W. M. G. Dissanayake, P. Newman, S. Clark, H. F. Durrant-whyte, and M. Csorba, "A solution to the simultaneous localization and map building (slam) problem," *IEEE Transactions on Robotics and Automation*, vol. 17, pp. 229–241, 2001.

[7] H. Durrant-Whyte and T. Bailey, "Simultaneous localisation and mapping," *IEEE Robotics and Automation Magazine*, vol. 2, p. 2006, 2006.

[8] J.-M. Frahm, P. Fite-Georgel, D. Gallup, T. Johnson, R. Raguram, C. Wu, Y.-H. Jen, E. Dunn, B. Clipp, S. Lazebnik, and M. Pollefeys, "Building rome on a cloudless day," in *ECCV*, 2010, pp. 368–381.

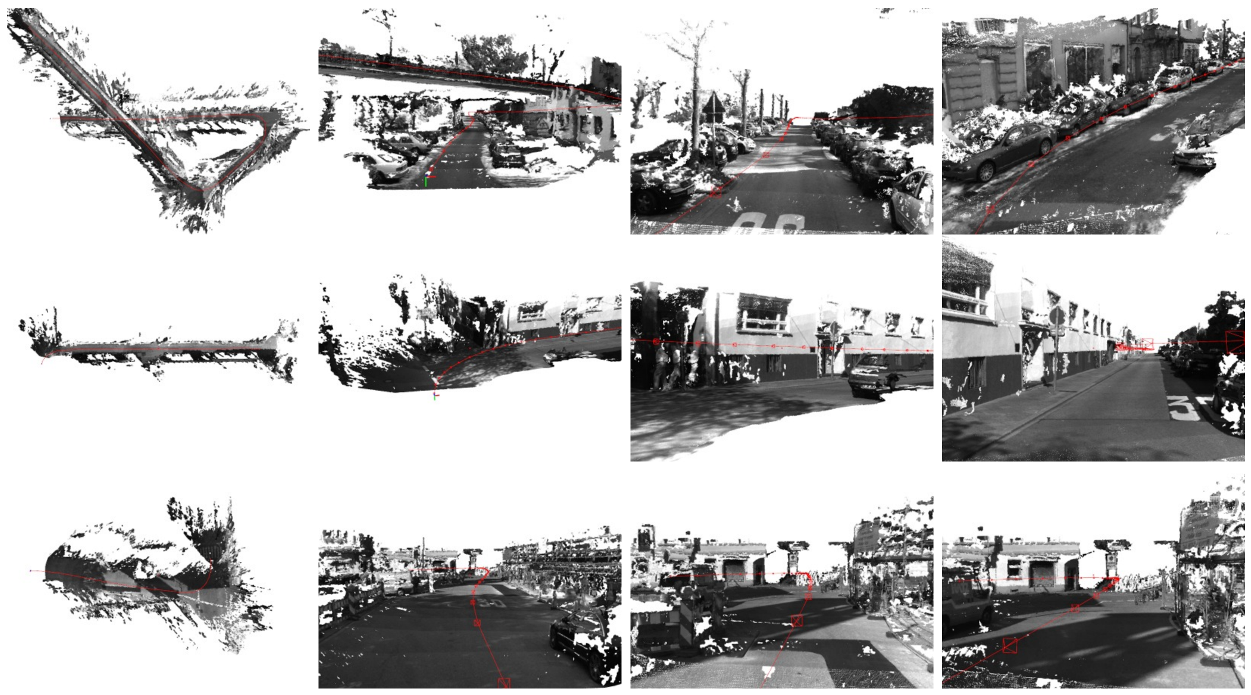


Fig. 8. **Inner-city stereo scans.** Four rendered views (columns) are shown for three sequences (rows). **Also see videos at: www.cvlibs.net**

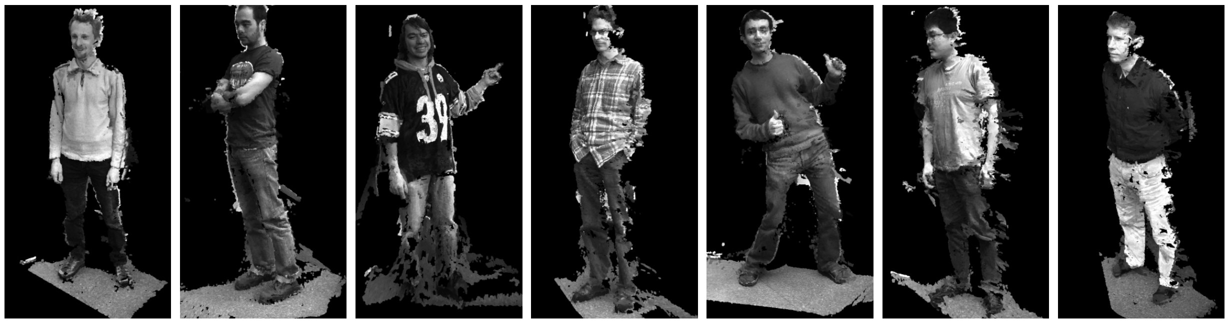


Fig. 9. **Novel viewpoints generated from real-time stereo scans of people in different poses with background automatically removed.**

[9] J.-M. Frahm, M. Pollefeys, S. Lazebnik, D. Gallup, B. Clipp, R. Raguram, C. Wu, C. Zach, and T. Johnson, "Fast robust large-scale mapping from video and internet photo collections," *ISPRS Journal of Photogrammetry and Remote Sensing*, 2010.

[10] D. Gallup, J.-M. Frahm, and M. Pollefeys, "Piecewise planar and non-planar stereo for urban scene reconstruction," in *CVPR*, 2010, pp. 1418–1425.

[11] A. Geiger, M. Lauer, and R. Urtasun, "A generative model for 3d urban scene understanding from movable platforms," in *Computer Vision and Pattern Recognition (CVPR)*, Colorado Springs, USA, June 2011.

[12] A. Geiger, M. Roser, and R. Urtasun, "Efficient large-scale stereo matching," in *Asian Conference on Computer Vision*, 2010.

[13] A. Gupta, A. A. Efros, and M. Hebert, "Blocks world revisited: Image understanding using qualitative geometry and mechanics," in *ECCV*, 2010.

[14] D. Hoiem, A. A. Efros, and M. Hebert, "Geometric context from a single image," in *ICCV*, 2005, pp. 654–661.

[15] C. Jennings and D. Murray, "Stereo vision based mapping and navigation for mobile robots," in *IEEE Conference on Robotics and Automation*, 1997, pp. 1694–1699.

[16] B. Kitt, A. Geiger, and H. Lategahn, "Visual odometry based on stereo image sequences with ransac-based outlier rejection scheme," in *IV*, 2010.

[17] R. Koch, M. Pollefeys, and L. J. V. Gool, "Multi viewpoint stereo from uncalibrated video sequences," in *ECCV*, 1998, pp. 55–71.

[18] K. Konolige, M. Agrawal, R. C. Bolles, C. Cowan, M. Fischler, and B. Gerkey, "Outdoor mapping and navigation using stereo vision," in *International Symposium on Experimental Robotics*, 2006.

[19] M. Montemerlo, S. Thrun, D. Koller, and B. Wegbreit, "Fastslam: A factored solution to the simultaneous localization and mapping problem," in *National Conference on Artificial Intelligence*. AAAI, 2002, pp. 593–598.

[20] A. Neubeck and L. V. Gool, "Efficient non-maximum suppression," in *ICPR 2006*, August 2006, in press.

[21] D. Pfeiffer and U. Franke, "Efficient representation of traffic scenes by means of dynamic stixels," in *IV*, 2010, pp. 217–224.

[22] M. Pollefeys, L. Van Gool, M. Vergauwen, F. Verbiest, K. Cornelis, J. Tops, and R. Koch, "Visual modeling with a hand-held camera," *IJCV*, vol. 59, no. 3, pp. 207–232, 2004.

[23] A. Saxena, M. Sun, and A. Y. Ng, "Learning 3-d scene structure from a single still image," in *ICCV*, 2007, pp. 1–8.

[24] S. M. Seitz, B. Curless, J. Diebel, D. Scharstein, and R. Szeliski, "A comparison and evaluation of multi-view stereo reconstruction algorithms," in *CVPR*, 2006, pp. 519–528.

[25] J. R. Shewchuk, *Applied Computational Geometry: Towards Geometric Engineering*, May 1996, vol. 1148, pp. 203–222.

[26] P. Smith, I. Reid, and A. Davison, "Real-time monocular slam with straight lines," in *BMVC*, 2006.