Real-time visual perception : detection and localisation of static and moving objects from a moving stereo rig

Benjamin Lefaudeux, Fawzi Nashashibi

Real-time visual perception : detection and localisation of static and moving objects from a moving stereo rig

Benjamin Lefaudeux
IMARA - INRIA Rocquencourt
Email: benjamin.lefaudeux@inria.fr

Fawzi Nashashibi
IMARA - INRIA Rocquencourt
CAOR - Mines Paristech
Email: fawzi.nashashibi@inria.fr

***Abstract*—Perception of the surrounding environment is one of the many tasks an automated vehicle has to achieve in complex and ever-changing surroundings. This typically includes several distinct sub-tasks, such as map-building, localisation, static obstacles detection, pedestrian detection,... Some of these tasks are nowadays very well known, such as map-building, whereas the perception, localisation and classification of moving objects from a moving vehicle are in many aspects a work in progress. In this paper, we propose a vision-based approach built on the extensive tracking of numerous visual features over time from a stereo-vision pair. Through on-the-fly environment 3D reconstruction, based on visual clues, we propose an integrated method to detect and localise static and moving obstacles, whose position, orientation and speed vector is estimated. Our implementation runs at the moment in a slow real-time (9fps), and should in the future be enclosed in a more complete, probabilistic pipeline.**

We present in the following our proposition for detection and localisation of independently moving objects from a moving platform, using only visual clues (stereo-vision cameras). Independent motion sensors, such as odometers or IMU units are thus not used in this algorithm, although they could be integrated to improve overall sensibility and robustness of the method. This paper is divided in three main parts. The first one recalls some techniques related to our implementation. A second part presents our exploitation of visual clues, and on-the-fly environment 3D reconstruction. The last part presents informative uses from this initial reconstruction and filtering, via the detection of navigable area and the detection, localisation, and size characterization of independently moving objects.

I. Existing techniques

The perception of a dynamic environment from a moving platform has been a broad research subject in the previous years, tackled by several techniques and associated pros and cons. We will present in the following an overview of some of them, as a non-exhaustive list of recent publications using vision to catch key elements of the environment. Some techniques are used in our proposition, and we try in this part to give a comprehensive overview of the state-of-the-art.
As regards intelligent vehicles needs, multiple challenges can be identified, each tackled by some of the following examples (order is non-relevant):

1) Ego-motion must be reliably estimated, as a prior to any further work. This allows for instance coherent exploitation of the flow of stereo-pairs acquisitions, or obstacle detection from monovision camera in a planar-world paradigm.
2) Moving parts must be detected from presumably static background.
3) Moving and static parts must be localised in space, in order to identify every possible concern.

The following sections are named according to a brief resume of their processing pipeline, and can be matched with this pattern. First and second section (**??** and **??**) follow a similar path : ego-motion is firstly estimated (via a least square optimisation in **??** and a RANSAC (*Random Sample Consensus*) in **??**). This allows motion detection, either in cartesian space (via the scene-flow, **??**), or in the picture space (**??**). Methods differ, but **??** and **??** are jointly estimated. Section **??** shows a different path : motion detection can be done prior to any ego-motion estimation, using typical human prior (pedestrian and cars are the most likely moving objects in an urban configuration). Pedestrian are first detected on stereo pairs using HOG (*Histogram of Oriented Gradients*) and SVM (*Support Vector Machines*). Ego-motion estimation and environment reconstruction (covering **??** and **??**) are then jointly processed. A last section, **??**, present an extensive reconstruction of the environment, without moving object detection.

A. Ego motion compensation and motion detection from optical flow

Badino et al. presented in 2008 ([**?**]) an algorithm to detect moving objects from a moving stereo-rig. Main ideas were ego-motion estimation from frame to frame, with an additional multi-frame refinement, over a set of 1200 followed points (with 2008 hardware, this number could probably be a lot higher with current hardware). Ego-motion estimation is done using a least-squares approach, common to many publications in this field and stemming from photogrammetry techniques (see [**?**] or [**?**] for example for some input on the

subject). Visual tracking uses speed-optimised KLT-tracker (see original publication from Lucas and Kanade [**?**]) together with a coarse-to-fine correlation method. Although specifics differ, since we intend to reconstruct (and filter) current environment to get information about accessible areas as well as moving objects typology, the principles of our work can very much be related to this publication. Main tools developed here should come as no surprise, being very common in the stereo-vision field, and used for SLAM purposes for some time (see for example [**?**] for an initial example of least-squares motion estimate from stereo-vision frames for SLAM purposes). We intend to more specifically focus on perceiving current environment features and improve stereo-vision accuracy via multi-sampling filtering (see **??**).

B. Absolute movement detection from optical flow and appropriate optimisation

Emphasizing movement detection, Agrawal et *al.* presented in [**?**] a strategy based once more on pure visual informations from a moving stereo pair. In this approach, moving parts segmentation is however based on the optimisation scheme, in the disparity space. Coordinates noise is indeed anisotropic in the 3D cartesian space in case of a stereo-vision input (a point extensively studied by Blostein ([**?**]) and which we use in **??**), a fact overlooked by standard *SVD* (*Singular Value Decomposition*) techniques used to gather point-cloud to point-cloud rigid transformations. This anisotropic noise stems from image quantization and feature correspondence to 3D-positions equations, quickly visible using the classical lens pinhole model. An interesting publication taking into account this noise distribution anisotropy to compute the rigid transformation between point clouds can be found in [**?**], by Matei et *al.*. In [**?**], points from a static background are selected randomly, on a RANSAC basis : the corresponding disparity to a random sampling from the set of tracked points is computed, and inliers are computed in the disparity space. Random sampling is repeated, and the winning draw in terms of inliers is kept, in a standard "Ransac" way. Non-linear Levenberg-Marquardt least-squares optimisation is then used to compute the optimal transformation from the winning draw, supposing a rigid body. Independent motion is finally detected comparing theoretical disparity (from ego-motion and past calculus) to the new measure, according to a correlation threshold.
In our case, motion estimation do not imply a Ransac strategy, because of the number of reference points involved (which we suppose to be much higher than the number of moving points). The method used to detect moving points is although different, ours being based in the reconstructed cartesian space, but the results are very comparable.

C. Visual recognition along with environment reconstruction

In a recent publication ([**?**], Schindler et *al* proposed a related framework on top of a similar stereo-vision setup, with an emphasis on initial pedestrian detection on pictures thanks to machine learning techniques . Once pedestrian candidates are located on the stereo pairs acquisitions, other parts of the picture are supposed to be static points, and are used to get the vehicle ego-motion using least-square estimates. Pedestrian detection is based on HOG detection and SVM classfiers, while corners are detected using Förstner corner detector ([**?**]) and used to estimate successive camera poses. Similarly to state-of-the-art monovision SLAM (Davison, [**?**]), an extended Kalman filter is used for pose estimation, robustified by RANSAC point selection and bundle adjustment on a sliding window. This allows in turn the localisation in the environment of tracked points placed on identified pedestrians, in this case in a probabilistic framework. Similarly to our approach, this paper present a partial reconstruction of the environment, over a sliding window, discarding older frames. We believe this gives an important short-term information for an intelligent vehicle, allowing for better recognition of moving objects (standing on the pavement, or on the road for example), and, as presented in **??**, for an extra source of navigation information (usable area,..). This approach achieves very convincing results in pedestrian tracking and localisation, but relies on the initial accuracy of its HOG-SVM pedestrian detection. We try in this proposition to adopt a more generic detection and tracking framework of moving objects, based on the tracking of a high number of visual features.

D. Dense environment reconstruction

Algorithms from the previous points have in common to be based on the the tracking of several features, as opposed to a dense tracking and reconstruction. Schindler et *al.* use dense depth maps for enhanced pedestrian depth localisation, but rely nonetheless on the tracking of discrete features for pose estimation and environment sliding reconstruction. Lateghan et *al.* present in [**?**] a new approach for dense environment reconstruction, ensuring few to no features are missed in the process. This relies on an initial EKF pose estimation (related to monovision SLAM, once more visible in [**?**]), robustified by RANSAC point selection, on a subset of the stereo pairs visual features. This gives a skeleton of environment features localisation, on which dense point clouds can be mapped. An extra improvement on the point cloud reconstruction is achieved using ubiquitous Kalman filtering for each point. This approach does not deal with moving objects though, visual reconstruction being the point for this publication.
Although we try in our proposition to focus on dealing with mobile objects, we share a common attempt for stereo-vision point cloud filtering, made possible by temporal tracking of the visual features. Our reconstruction, while not being dense, attempts to be "dense enough" not to miss any interesting feature. Contrary to the Lategahn approach however, we rely directly on the whole point cloud to estimate successive camera poses, via standard photogrametry techniques (namely SVD). In this case we suppose the number of static points is vastly superior to the number of moving points in the picture, which ensures a reliable estimation of the vehicle movements. In case this could not be achieved, an interaction with motion sensors would however be needed.

II. On the fly environment reconstruction from visual clues

In our approach, perception and localisation of moving objects start with a partial reconstruction of the environment. This implies to deal with the ego-motion of the vehicle, as well as to implement 3D perception from our inputs. We chose to stick to stereo-vision setup for visual spatial perception, because of its intrinsic robustness for the perception of moving objects from a vehicle standing still, a common use case in the transportation area in which monocular SLAM cannot perform.

Visual clues are gathered using a temporal stereo-vision framework, a variation of recent publications in this field ([**?**], [**?**]). Visual clues at this step are sparse, but numerous (1000 features ensures 20fps, 4000 running currently at 9fps). From this point, we obtain time-connected 3D points clouds which can be used to reliably and efficiently estimate ego-motion of the vehicle using a modified SVD-based routine. This gives a multi-sampled view of the same scene, visual features being tracked over time. Next step is a filtering of features position, based on this multiple sampling, as opposed to constraints-based filtering commonly applied. Information is in our case really gathered from the observations, and not stemming from a smoothing constraint, which we believe improves accuracy. These "4D" point-clouds are finally used to detect and track moving objects, as well as gather informations for ulterior path-planning tasks (navigable area, static obstacles,..).

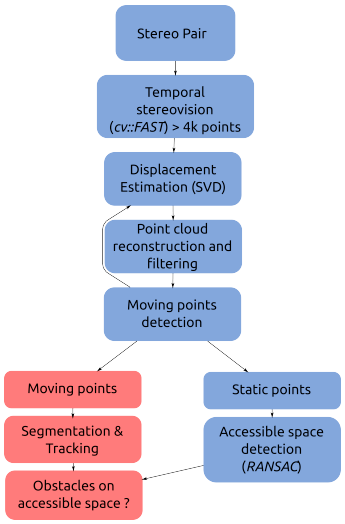

Fig. 1. Overall view of the algorithm

A. Temporal stereo-vision

1) Visual processing pipeline: The perception of spatial features from pictures, relies on the determination of distinct physical points which can be tracked from one view to another (being in time or regarding another camera). A lot of work have been invested in this field, from corners detection to feature tracking, which we benefited greatly.

We used FAST corner detection, presented by Rosten et *al.* in [**?**] and readily available in OpenCV library. Feature tracking uses a very common pyramidal KLT setup, whose principles were initially proposed by Lucas and Kanade in [**?**].

One pair of pictures is kept in a temporal buffer to improve tracking reliability over image pairs and time, tracking of the same visual features over time being a critical step of our algorithm.

Cycling steps are as follows :

1) **Corner detection** is applied on current left picture. Only the needed number of points are initially randomly selected among top features. Every detected corner (ranked up to the desired number) is to be tracked at first, whereas after the first loop this number is reduced to the lost points of the previous loop (100 to 300 points on average), corners successfully tracked being kept over time.
2) **Feature tracking** using pyramidal Lucas & Kanade tracker. This tracking is applied on a loop, one pair of stereo pictures being kept in the back buffer. Initial tracking looks for the features from current left picture to past left picture. The same features are then tracked on the past right picture (very fast step, the pictures being rectified), then from the past right to the current right picture, to finish with a tracking from the current right to the current left picture (again a very fast step).
3) **Tracking coherency check** is immediate, consisting in a comparison of the initial position of the tracked features to their position found after the tracking loop. In our experiments, mismatches of more than 0.5 pixels are rejected. This defines a list of points to be replaced, needed for the first step.

Overall, these steps take around **60ms** per pair of frames on a Intel Core i7 laptop. This is by far the slowest part of the algorithm, an extra load being assumed by the doubled tracking (current-past-current and left-right-left steps). This however ensures a reliable source of information, and we considered this is worth the extra time. Compared to other sensors (laser, range-cameras), this step may seem too slow and could be a case against the use of vision. One should however take into account the nature of the information gathered, every point being tracked in space and time at the end of this visual processing, contrary to typical laser acquisition which would for example need ICP to get time correspondence.

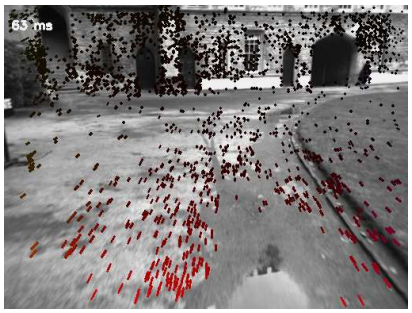

Fig. 2. Example of frame-to-frame feature tracking (FAST keypoints & pyramidal KLT), leading to Figure **??**

B. 3D reconstruction

1) Rigid transformation from connected point clouds: Following the visual tracking processing, 4D clouds are available : each and every visual feature tracked gives a set of positions in space over time. Visibility window over a set of frames is individual to tracked features, as they disappear or are lost in an independent manner, an index table needs to keep track of every visibility change. The first step is to compute an iterative transformation to account for ego-motion. Several techniques are readily known for this step, from Kalman filtering to optimisation techniques, some of them quickly presented in **??**. We chose to adapt the classical SVD implementation to account for specific stereo-vision noise, which is a contribution of this paper. This ego-motion estimation technique is very fast (in the *ms* range) and can handle a large number of points, contrary to the Kalman filter approach. Outliers selection and removal is handled iteratively to improve estimation robustness, which is detailed below.

2) Depth-weighted least-squares estimation: A thorough comparison of four standard algorithms to estimate rigid body transformation can be found in the paper of Eggert et *al.* ([**?**]), whose notations are kept in the following equations. Considering two point clouds *m* and *d*, the error to minimize under the common L_2 norm is written :

$$\Sigma^2 = \sum_{i=1}^{N} \|d_i - \hat{R}m_i - \hat{T}\|^2 \tag{1}$$

This translates nicely into the maximization of *Trace*($\hat{\text{R}}$. H), where *H* is the correlation matrix defined by the cross product of m^c and d^c vector coordinates (centered point clouds):

$$H = \sum_{i=1}^{N} m_i^c \cdot d_i^{cT} \tag{2}$$

Trace maximisation is achieved by SVD of the H matrix into the $U\Lambda V^T$ product, which defines the maxima as $\hat{R} = VU^T$. Translation vector is finally computed by subtracting the rotated cloud to the reference one. This is a well known algorithm, very fast for clouds of thousands of points, and whose precision is comparable to other leading quaternions technique in standard cases ([**?**]). Equation **??** is however questionable in our case, every point of the $\{m, d\}$ clouds being given the same weight under the L_2 isotropic norm. It is well known that stereo-vision noise is anisotropic and dependent on the distance to the pair of cameras (see [**?**] for an initial study on stereoscopic noise). This led for instance some of the work presented in **??** to be computed in the disparity space (notably independent movement detection in [**?**]), where noise is uniformly distributed. We propose an alteration of equation **??** to uniformly weight points contribution in the picture space, thresholded to take into account close points (with z the depth coordinate, and z_{th} the threshold distance to robustify our norm against close points) :

$$\omega(z) = max(z_{th}, z) \tag{3}$$

$$\Sigma^2 = \sum_{i=1}^{N} \| \frac{d_i^c}{\omega(d_{i_z})} - \frac{\hat{R}m_i^c}{\omega(m_{i_z})} \|^2 \tag{4}$$

(The following SVD-based optimisation resolution steps keeps the same). Considering stereo-vision equations in the standard pinhole model, this is an approximation to back-projecting points onto the picture frame, and computing error cost in this neat uniform space. Computing time is marginally superior, but we believe this specific SVD-based declination is more adapted to stereo-vision noise profile. In addition to his specific optimisation, the transformation estimation is robustified by an iterative method, removing worst points (in terms of L_2 norm in matching clouds) until the standard deviation of matched clouds in the same referential is low enough. A few iterations (5 at most) and a few hundreds discarded points are typically enough on KITTI benchmark sequences ([**?**]), which keeps computing time within real-time constraints.

3) Transformation accumulation: Following eq. **??**, we obtain iterative transformations consecutive to ego-motion. These transformation matrix are incrementally combined in order to keep an estimation of the motion from frame *k* to frame *k+l* in the back buffer, which is the simplest way to be able to bring every cloud back to the same referential. At this point, reconstruction is obviously biased over time, and a precise optimisation would be required to keep going on (see for example this extensive review from Triggs et *al.* [**?**]). This could be added to our pipeline, but current results show that precision is good enough for this reconstruction over a sliding window of 100 frames, which is enough for current purposes. Considering real-time computing constraints, a balance must be stroke depending on cloud accumulation bias and the possibility to deal with more clouds (to offset prominent stereo-vision noise at high range via sample accumulation). In our "sliding window" approach bias prove to be negligible (we always work in the current referential of the camera), but bundle adjustment would be needed in case the filtered point-clouds had to be kept over time.

4) Cloud filtering via position multi-sampling: At this point, we get a sampling of every tracked feature x_m position depending on its intrinsic visibility window W(x_m) over the [1,N] sliding reconstruction window.

$$\{x_m\} = \forall k \in W(x_m)\{\hat{x}_{m_k} \cdot \begin{bmatrix} R_k & T_k \\ 0 & 1 \end{bmatrix}\} \tag{5}$$

This allows for an improved position of the feature over time, as long as its visibility window extends. This can be opposed to common filtering techniques for depth maps, which inject *a priori* knowledge in the computation, supposing for example that the environment consists of smooth edges. In our case, we effectively get largely uncorrelated samples, which gives an effective increased information over each localised point time, without loss of generality. Similarly to

eq. **??**, we propose an inverse-depth based weighting over the $\{x_m\}$ positions, in order to partially account for stereo-noise specificities. This proved to greatly enhance feature position convergence over time (see **??**) .

$$\hat{x_m} = \frac{1}{\mathcal{N}(W(x_m))} \sum_{k}^{W(x_m)} \{x_m\}_k \cdot \frac{1}{depth_{x_k}} \tag{6}$$

III. Point cloud exploitation

A. Navigable space detection

Building on the point cloud at our disposal, we use a RANSAC (see Fischler and Bolles [**?**]) based strategy to detect the ground, supposing vehicle attitude and camera calibration is not known. Informations are still only picture-based, and we suppose to begin with that the ground is in the lower part, and mostly horizontal. RANSAC is very robust, but can be long in a big sampling space, so our strategy is divided in two steps :

- If the previous iteration did not converge, RANSAC sampling basis is the lower third part of the filtered point cloud. Every draw consists of 3 points, from which a plane is computed, driving to a number of inliers being selected. The best draw in terms of inliers is kept. Limit can be set on the expected ground slope, provided the vehicle attitude towards ground is well known.
- If the previous iteration did converge (inliers within a given range above a threshold), computation can be greatly speed up. Cloud-to-cloud transformation is used on the previous plane equation. RANSAC sampling basis is then selected around this transformed past plane (*+/-1m* in our experiments), and the same typical non-linear optimizing strategy is applied.

Ground selection interest is twofold : for once, it gives a simple means for selecting most interesting moving objects, the one susceptible to enter in collision. On the other hand, it gives an estimate of the navigable area (which may need to be crossed checked with prior data, depending on the considered vehicle and playground), that is where to go. This step initial implementation costs around 10 to 15 ms on a laptop, but this could be sped up considering its highly parallel nature (multi-threaded sample inlier computation). This step can be compared to numerous free space detection publications ([**?**], [**?**]), although our approach certainly is heavier because of the current scene reconstruction.

B. Moving objects detection

We try in this step to detect any independent motion, without any prior separating moving points from static ones (contrary to [**?**]). This is made difficult by stereo-vision noise, which can lead to strong sample-to-sample motion on static points, especially at a long range. Agrawal et *al.* propose a disparity-based detection, once ego-motion has been reliably estimated ([**?**]), but we have at this point more information : our features are tracked on a subset of the overall integrating sliding window, which translates into several tenth of successive positions in the same referential ; and we have some clues about the environment configuration. The detection strategy follows several steps, in order to refine from the initial point cloud counting 50 000 points to a more manageable number of refined candidates.

- Initial candidates are taken among the "worst" points of the cumulated cloud, in terms of L_2 norm. We compute the standard deviation for the whole cloud, points whose distance to their filtered counterparts (weighted mean) is above a threshold times this standard deviation for a few consecutive frames are selected. At this point, moving objects are actually among this candidate points, as well as noisiest points (long range). $\sigma_{k,l}$ is in this case the standard deviation computed between the clouds *(k, l)* back in the same referential.

$$\sigma_{k,l} = STD_{i \in \{k,l\}}(\hat{x_{ik}} - \hat{x_{il}}) \tag{7}$$

$$STD_{\{k,l\}}(\{x_{\{k,l\}}\}_{candidates}) > \sigma_{k,l} \tag{8}$$

- We then compute the autocovariance between the beginning and the end of the set of positions at disposal for every candidate. This effectively differentiates noisy static points to their moving counterparts, up to long-range stereovision bias.
- Selected moving points are then, for example, restricted to the ground neighbourhood. Initial segmentation is done on a closeness basis (K-Means clustering and Global Nearest Neighbour approach).

C. Output example and (lack of) benchmarking

In the following example, we use the data set from New College ([**?**]) gathered from a two-wheel moving platform. We believe this shows a difficult exercise, the amount of ego-motion (including pitch and roll) being arguably superior to most four-wheeled vehicles. Speed is however on the low side, which proved positive for our approach, as initial tests using the just released KITTI Vision Benchmark Suite ([**?**]) tend to show higher speeds and low sampling rate could be challenging. We did not at the time have any ground-truth as regards moving object detection and localisation, and the KITTI benchmark would help a lot in this field.
An example of a moving object detection and localisation, with estimated speed vectors, can be found in Figure **??**. In this 3D output, incremented visual features are drawn in green, while detected speed vectors are in red. Stereo cameras are "Point Grey Bumbleebee", running at 20 Hz and with a 512x384 grayscale definition.

Our method effectively detects and localises all moving pedestrians of the dataset, provided detection threshold is low enough and triggers false detections (this benchmark does not have a ground truth as regards pedestrian position over time). This dataset is however arguably a simple one as regards pedestrian presence, less than 10 people being visible over a few minutes. This proposition is however a first step in the detection chain, and should be coupled in the future with a probabilistic grid framework.

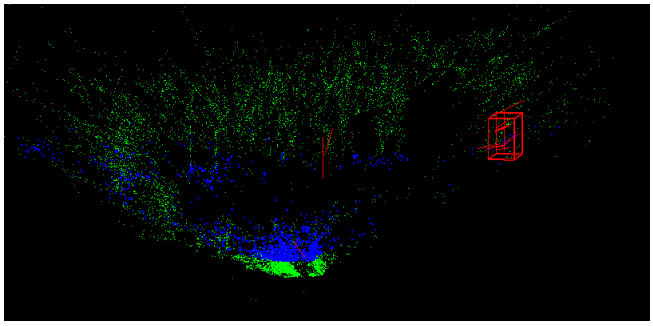

Fig. 3. Algorithm output. Vectors drawn in red for moving object. Ground points are in blue. Vehicle trajectory is also visible at the bottom end, in red

We did not have enough time to fully exploit KITTI benchmark, which is to our knowledge the first benchmark providing extensive ground truth over the detection and tracking of moving object. An example of on-the-fly reconstruction of the current environment is however visible in Figure **??**.

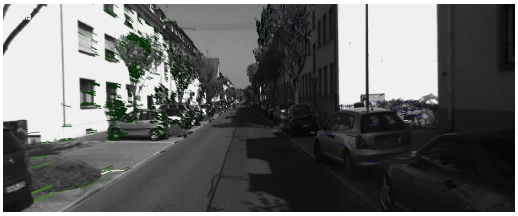

Fig. 4. Camera input leading to Figure **??**

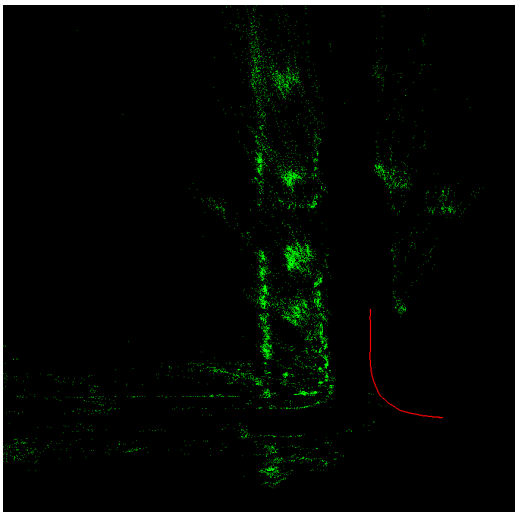

Fig. 5. Algorithm output. Scene is from above, vehicle just turned right, gathered trajectory is in red. Reconstrution runs at 9fps

IV. Conclusions

We present in this paper an integrated means of environment reconstruction and moving object detection, over a sliding window. While not aimed at visual mapping, our approach allows a noticeable increase in objects localisation accuracy over time, and effectively detects and localises moving objects from a moving stereo rig in their environment. Computing cost allows for real-time prospects, but this approach needs observability over time to estimate ego-motion and filtering, and high-speed operations would probably be challenging.

References

[1] H. Badino, U. Franke, C. Rabe, and S. Gehrig, "Stereo-vision based detection of moving objects under strong camera motion," in *International Conference on Computer Vision Theory and Applications*. Citeseer, 2008, pp. 253–260.

[2] D. Eggert, A. Lorusso, and R. Fisher, "Estimating 3-D rigid body transformations: a comparison of four major algorithms," *Machine Vision and Applications*, vol. 9, no. 5-6, pp. 272–290, Mar. 1997.

[3] B. Triggs, P. McLauchlan, R. Hartley, and A. Fitzgibbon, "Bundle Adjustment A Modern Synthesis," *Vision algorithms: theory and practice*, vol. 34099, pp. 153–177, 2000.

[4] B. D. Lucas and T. Kanade, "An Iterative Image Registration Technique with an Application to Stereo Vision," *Imaging*, vol. 130, pp. 121–129, 1981.

[5] K. Konolige and M. Agrawal, "FrameSLAM: From Bundle Adjustment to Real-Time Visual Mapping," *IEEE Transactions on Robotics*, vol. 24, no. 5, pp. 1066–1077, Oct. 2008.

[6] M. Agrawal, K. Konolige, and L. Iocchi, "Real-time detection of independent motion using stereo," in *Motion and Video Computing, 2005. WACV/MOTIONS'05 Volume 2. IEEE Workshop on*, vol. 2. IEEE, 2007, pp. 207–214.

[7] S. D. Blostein and T. S. Huang, "Error analysis in stereo determination of 3-D point positions," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, no. 6, pp. 752–765, 1987.

[8] B. Matei, "Optimal rigid motion estimation and performance evaluation with bootstrap," *Vision and Pattern Recognition, 1999. IEEE*, no. 2, 1999.

[9] K. Schindler, A. Ess, and B. Leibe, "Automatic detection and tracking of pedestrians from a moving stereo rig," *ISPRS Journal of*, 2010.

[10] W. Förstner and E. Gülch, "A fast operator for detection and precise location of distinct points, corners and centres of circular features," in *Proc. ISPRS intercommission conference on fast processing of photogrammetric data*. ISPRS Intercomission Workshop, Interlaken, 1987, pp. 281–305.

[11] A. J. Davison, "Real-Time Simultaneous Localisation and Mapping with a Single Camera," London, pp. 0–7, 2003.

[12] H. Lategahn, A. Geiger, and B. Kitt, "Visual SLAM for Autonomous Ground Vehicles," *Robotics*, pp. 1732–1737, 2011.

[13] P. Lenz, J. Ziegler, A. Geiger, and M. Roser, "Sparse Scene Flow Segmentation for Moving Object Detection in Urban Environments," *Intelligent Vehicles Symposium (IV), 2011*, pp. 926 – 932, 2011.

[14] E. Rosten and T. Drummond, "Fusing points and lines for high performance tracking," *Tenth IEEE International Conference on Computer Vision (ICCV'05) Volume 1*, pp. 1508–1515 Vol. 2, 2005.

[15] A. Geiger, P. Lenz, and R. Urtasun, "Are we ready for Autonomous Driving? The KITTI Vision Benchmark Suite," *Computer Vision and Pattern Recognition (CVPR)*, no. June, 2012.

[16] M. Fischler and R. Bolles, "Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography," *Communications of the ACM*, vol. 24, no. 6, pp. 381–395, 1981.

[17] H. Badino, U. Franke, and R. Mester, "Free space computation using stochastic occupancy grids and dynamic programming," in *Workshop on Dynamical Vision, ICCV, Rio de Janeiro, Brazil*. Rio de Janeiro: Citeseer, 2007, pp. 1–12.

[18] D. Pfeiffer, A. Barth, U. Franke, and A. Daimler, "Robust and Precise 3D-Modelling of Traffic Scenes based on Dense Stereo Vision," *vldb.informatik.hu-berlin.de*, vol. 1.

[19] M. Smith, I. Baldwin, W. Churchill, R. Paul, and P. Newman, "The New College Vision and Laser Data Set," *The International Journal of Robotics Research*, vol. 28, no. 5, pp. 595–599, May 2009.