Monocular vSLAM and Fast Piecewise Planar Reconstruction using πMatch

Diogo Vaz, Carolina Raposo and João P. Barreto

***Abstract*— πMatch is a recent monocular vSLAM pipeline with the particularity of being a feature-based method that, unlike other non-direct approaches, provides dense reconstructions. It uses Affine Correspondences (ACs) to recover both the camera motion and the 3D planes of the scene, and efficiently tackles problems faced by other direct and non-direct methods. Despite its important advantages, it has two main bottlenecks that hamper real-time performance. This paper advances the πMatch pipeline by modifying two of its modules, leading to a higher accuracy in the camera motion estimation, as well as a dramatic improvement in the computational efficiency, with a speedup of over 40×. The main source of improvement comes from a new Markov Random Field (MRF) formulation that allows a very fast and accurate dense segmentation of the images and subsequent Piecewise Planar Reconstruction (PPR). Reconstruction results on a challenging loop-closing sequence demonstrate the clear superiority of the proposed MRF approach, when compared to a sophisticated point-based method, both in terms of computational efficiency and quality of the 3D model.**

I. Introduction

Simultaneous Localization and Mapping (SLAM) is an important topic in robotics due to its numerous applications such as autonomous navigation [1], [2] and 3D reconstruction of indoor environments [3]. SLAM is a generic process to estimate the position of a device, and, at the same time, build a 3D map of its surrounding environment from the data acquired by one or more sensors attached to it. When these sensors are cameras, the problem is referred to as Visual SLAM (vSLAM) and there are typically two families of methods to accomplish it: direct and feature-based/non-direct methods. The latter are based in point correspondences across frames, having the advantages of being fast, relatively robust to outliers and changes in illumination, and able to cope with wide-baselines [4], [5]. The former use the information of the entire image, having the advantages of providing dense reconstructions as opposed to a sparse point cloud [6], [7]. Both approaches have difficulties in handling situations of dynamic foreground, pure rotation and/or presence of multiple rigid motions.

Our previous work in monocular vSLAM presented in [8] introduced, for the first time, an approach that is different from the existing ones in the sense that it is feature-based but relies in plane primitives, which are obtained by using affine correspondences (ACs), as opposed to point correspondences. This plane-based approach, dubbed πMatch, is able to conciliate the benefits of direct and non-direct methods, being computationally efficient, handling wide-baselines, and providing dense 3D models by performing dense pixel labelling using a standard Markov Random Field (MRF) formulation [9], [10]. In addition, the method is able to handle situations not only of outliers and changes in illumination, but also dynamic foreground, pure rotation and multiple motions. These advantages are summarized in Table I.

	Direct	Features	πMatch [11]
Robust to outliers	✘	✔	✔
Wide baselines	✘	✔	✔
Moving objects/Pure rotation	✘	✘	✔
No prior information	✘	✘	✔
Dense 3D models	✔	✘	✔

TABLE I: Comparison between direct, non-direct and πMatch monocular VSLAM methods.

The motivation for this work arose from the fact that a MATLAB implementation of the πMatch algorithm is unable to run in near-real time, limiting its usability. Thus, we implemented the algorithm in C/C++ and achieved an average speed up of approximately 1.7×, which was still unsatisfactory for a near-real time application. A more careful analysis of the computational performance allowed us to identify two main bottlenecks to be removed.

This paper develops further this promising new paradigm for vSLAM, improving its computational efficiency and resilience to scale drifts in long sequences. In particular, we propose modifications to 2 of the original modules of the pipeline, leading to a speed up of over 40× with respect to a straightforward C++ implementation while improving overall accuracy and robustness.

The modified modules are:

i) **AC extraction:** the new module is 2.6× faster than the original one and ensures a uniform spreading of the features in the images, benefiting the subsequent steps of the pipeline and thus improving the overall estimation accuracy.

ii) **MRF for planar segmentation of images:** the new MRF formulation makes use of superpixels [12] to significantly speed up the segmentation of images into planes (40×), while maintaining accuracy.

It is important to note that the proposed changes may have applications in other pipelines. As as example, any pipeline that employs MRF for piecewise planar segmentation of images [9], [13] can use the formulation proposed in this paper for that task.

The authors acknowledge FCT and COMPETE2020 program for generous funding through project VisArthro with reference PTDC/EEI-AUT/3024/2014.

The authors are with the Institute of Systems and Robotics, Dept. of Electrical and Computer Engineering, University of Coimbra, Portugal.

The source code is available online from `http://arthronav.isr.uc.pt/~diogovaz/piMatch/PiMatchCpp.zip`.

II. Overview of πMatch

πMatch [8] is a monocular vSLAM method that relies on plane features to estimate the camera motion and a PPR of the scene. It can be divided into several sequential modules, as follows. For each pair of frames, the method starts by extracting affine features in each image, which are then matched. The output of this first module, denoted by *ACs*, is a set of ACs. In the next module (Π *Det*), these ACs are clustered into coplanar regions by using a metric proposed in [11]. Each plane cluster yields an homography, estimated in a robust manner (using RANSAC), which is decomposed into two rigid transformations that constitute hypotheses for the camera motion. This step is denoted *M Hyp*. Module *Cam M* receives these motion hypotheses as input and merges them in a PEaRL formulation [14], being also able to identify and handle situations of dynamic foreground, multiple motions and pure rotation. Another PEaRL step is performed for merging and refining the plane hypotheses (Π *Merge*). πMatch has a final step (*MRF*) for dense pixel labelling, and subsequent PPR, which is optional due to its very high computational cost.

A. Translation from Matlab to C++

The original implementation of πMatch presented in [8] was done in MATLAB, being unable to run sufficiently fast for online applications. Thus, we started by implementing the pipeline in C++ and, despite the average speedup of 1.7×, the overall computational performance is still unreasonable for a vSLAM pipeline.

In order to find possible bottlenecks and assess the time complexity of each module of the pipeline, we did a profiling using a set of 6 image pairs (shown in Fig. 2) that contains different situations of lighting, texture, and type of scene (indoor/outdoor). This set of images, with resolution of 720p, is used throughout the paper as study cases. They were acquired with the left and right cameras of a Bumblebee stereo pair, so that the ground-truth of the camera motion

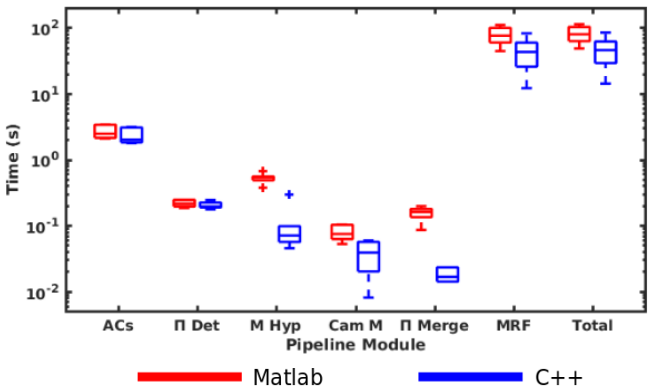


Fig. 1: Distribution of computational times per module, in seconds, of the Matlab (red) and C++ (blue) versions.

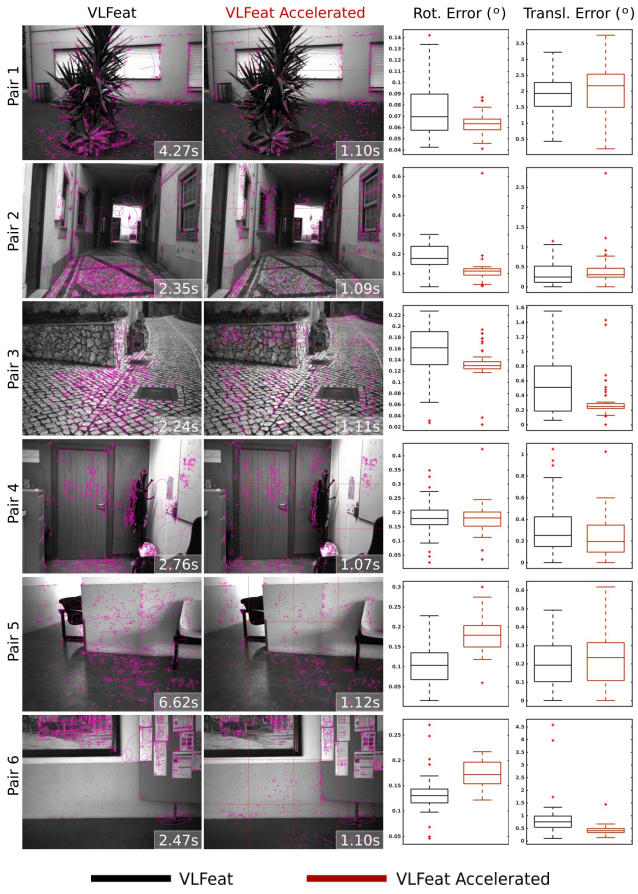


Fig. 2: ACs extracted on the 6 test pairs (first image shown) using VLFeat and the proposed VLFeat Accelerated approach, and their average times. Distribution of rotation and translation errors obtained with each method, for 50 different runs of the algorithm.

$(\mathrm{R}_{GT}, \mathbf{t}_{GT})$ is known. All tests were performed on an Intel Core i7-3610QM CPU @ 2.30GHz processor.

Fig. 1 shows the distribution of times, per module, obtained with the Matlab and C++ implementations. It becomes clear that the MRF stage is the main bottleneck of the pipeline, being more than 20 times slower than the other modules. Besides MRF, only the first step of AC extraction takes over 1 s to complete[1], and thus it is identified as another bottleneck. This paper proposes solutions for these two main issues, which are described next.

III. Fast Establishment of ACs

Empirical observation showed that the good functioning of πMatch depends on the quality and spatial spreading of ACs and not only on the number of these correspondences, i.e., it is relatively indifferent to have 100 or 1000 ACs on the same plane. Also, the computational performance of the method highly depends on the number of ACs. Thus, improvements

[1]Please note that the incongruence with the execution times reported in [8] is due to the fact that different processors and image resolutions were used. Also, in [8], the algorithm was executed in a batch manner, i.e., AC extraction was performed for several images simultaneously.

in performance pass by limiting the number of ACs, while assuring that they properly sample the images and represent and different planes.

In the original pipeline presented in [8], AC establishment is performed using the standard implementation provided by the VLFeat library [15]. This step includes feature detection, affine shape estimation and matching. Detection is performed in scale space with saliencies being chosen as points whose derivative along scale is above a certain threshold δ. Also, the matching process requires the computation of the distance between all the features in the two images of the pair, becoming prohibitive for a large number of points. The strategy employed in [8] to limit the number of saliencies is by increasing δ, but, depending on the texture of the image, this may cause problems of high concentration of features in some regions and lack of features in others.

In order to solve this problem, the proposed AC extraction method, named VLFeat Accelerated, divides the image into blocks, which enables to speed up detection by parallelizing the process, as well as to limit the number of ACs per block while assuring spreading. The limitation of the number of ACs, in each block, is not performed by increasing the threshold δ, since this would lead to no detection in poorly textured blocks. In this method, δ is kept low and only a part of the detected features is considered. A straightforward way of limiting the number of features is to randomly select them. However, it was experimentally observed that many features did not provide a match, being discarded. This problem arose because of the poor quality of the selected features. VLFeat accelerated solves this problem by selecting the best features as the ones that provide a higher value of the derivative along scale. This selection method provides a significantly higher number of matches than random selection, benefiting the subsequent steps of the pipeline. As a last step, the matching process is accelerated by parallelization.

A. Performance of VLFeat Accelerated

The performance of the new AC extraction module, both in terms of computational efficiency and accuracy, is assessed using the set of 6 study case images. Due to the random nature of the motion hypotheses generation module, different results may be obtained in different runs of the algorithm. Thus, we performed 50 runs of the pipeline using as AC extractor both VLFeat and VLFeat Accelerated, and computed the rotation error as the angular magnitude of the residual rotation between the estimated one and R_{GT} and the translation error as the angle between the estimated translation and $\mathbf{t}_{GT}$. The distributions of these errors, as well as the ACs extracted by each method, are shown in Fig. 2.

Results show that using VLFeat Accelerated instead of VLFeat typically leads to higher accuracies. This can be explained by the much more uniform spatial spreading of the features that is achieved. In addition, the execution times are presented on the bottom right corner of each image, showing an average speed up of VLFeat Accelerated over VLFeat of 2.6×. Besides being significantly faster, VLFeat Accelerated also provides much more stable computational times, evinced by the difference in their standard deviations, which are 0.011s as opposed to 1.328s for VLFeat.

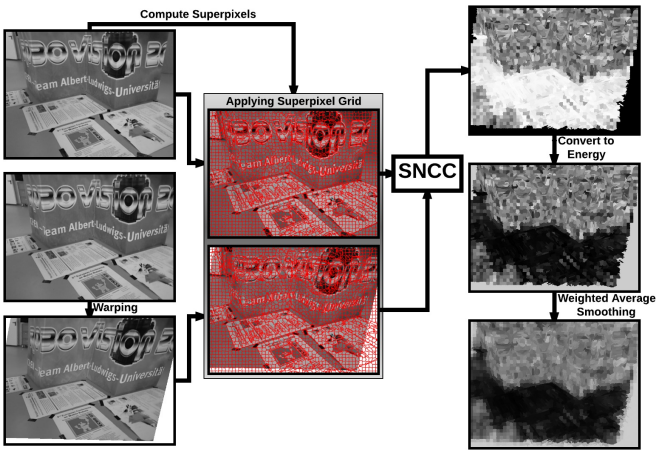


Fig. 3: Schematic representation of the data term computation for one plane.

IV. Fast Segmentation of Images into Planes

The purpose of the final step of the πMatch pipeline is to perform a pixel-wise labelling of the images into planar regions, having as input the camera motion and the planes of the scene. After this labelling, the 3D points are reconstructed according to the plane they are assigned to, providing a PPR of the scene. This dense segmentation is formulated as a discrete optimization problem using a standard MRF approach [9], where the nodes of the graph are the image pixels ($p \in \mathcal{P}$, where $\mathcal{P}$ is the set of pixels) and the labels l are the plane hypotheses (plus the discard label $l_\emptyset$ used to identify non-planar objects). The cost function to be minimized contains data and smoothness terms, as follows.

$$E(\mathbf{l}) = \underbrace{\sum_{p \in \mathcal{P}} \mathrm{D}_p(l_p)}_{\text{Data Term}} + \lambda_S \underbrace{\sum_{(p,q) \in \mathcal{N}} \mathrm{V}(p,q)}_{\text{Smoothness Term}}, \qquad (1)$$

where the data term function $\mathrm{D}_p(l_p)$ is defined by the normalized cross-correlation (NCC) between two images, $\mathrm{V}(p,q)$ is the spacial smoothness term, λ_S is a weighting constant, $\mathcal{N}$ is the 4×4 neighbourhood of p and $\mathbf{l}$ is the labelling. Even for small resolution images, this is a complex optimization problem due to the high number of nodes, being inadequate for use in vSLAM approaches.

A. Superpixel-based MRF

The solution to the complexity issue passes by reducing the number of nodes in the graph such that a near real-time dense planar segmentation of the images is achieved. To accomplish this, we propose to use superpixels as nodes because they divide the image into similar regions in terms of texture, and it is reasonable to assume that neighbouring pixels with identical values belong to the same plane.

The first image of the pair is fragmented into a grid of Preemptive SLIC superpixels [12]. The second image is warped by the homographies associated to the candidate

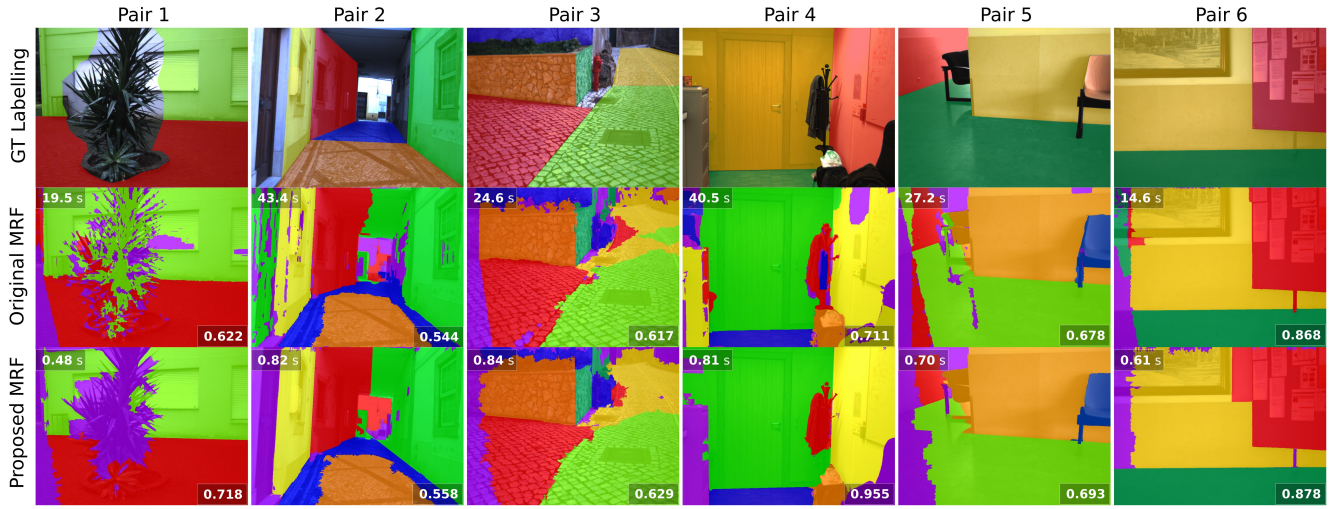


Fig. 4: Planar segmentation results obtained with the original and the proposed MRF formulations. The Jaccard indices are shown on the bottom right corner of each image and the computational times appear on the top left corner. The manual labelling of the images, considered as Ground Truth (GT) is shown on the top row.

planes, generating a number of transformed images equal to the number of planes. In the original MRF formulation, the NCC is computed for each pixel between the first and the transformed images, being a slow process. We propose a Superpixelwise Normalized Cross Correlation (SNCC) to quickly and directly determine, for each superpixel, the photo-consistency between the two images. The photo-consistency measure is the NCC calculated with superpixels as windows, which is then converted to the NCC Energy (NCCE) by $\text{NCCE}(p,l) = -0.5(\text{NCC}(p,l)-1)$ because we are minimizing a cost function. Since SNCC is computed without overlay of windows, the resulting NCCE transitions between neighbour superpixels are abrupt and need to be smoothed with a weighted averaging function:

$$\text{NCCEavg}(S,l) = \alpha_0 \text{NCCE}(S,l) + \sum_{i=1}^{M} \alpha_i \text{NCCE}(N_i,l) \quad (2)$$

where α_0 and α_i $(i = 1 \dots M)$ are a set of weights whose sum is 1, M is the number of neighbour superpixels and N_i is neighbour superpixel i of S. These steps are illustrated in Fig. 3.

In this MRF formulation, the data term is defined as

$$\text{D}_p(l) = \begin{cases} \min(\text{NCCEavg}(p,l), \text{D}_{max}) & \text{if } l \neq l_\emptyset \\ \text{D}_\emptyset & \text{if } l = l_\emptyset \end{cases}, \quad (3)$$

where D_{max} and $\text{D}_\emptyset$ are constants, and the smoothness term becomes

$$\text{V}(p,q) = \begin{cases} 0 & \text{if } l_p = l_q \\ G \cdot \text{T} & \text{if } l_p = l_\emptyset \vee l_q = l_\emptyset \\ G \cdot \min(d(p,q), \text{T}) + t & \text{otherwise} \end{cases}, \quad (4)$$

where $G = \frac{1}{\lambda_{grad} \nabla I^2 + 1}$ and λ_{grad}, T and t are tuning parameters. Regarding functions $d(p,q)$ and ∇I^2, novel definitions are proposed. Two superpixels p and q are considered neighbours if they are adjacent. For neighbouring superpixels, the line segment that links their centroids is intersected with their inner borders, yielding two distinct pixels. These pixels are reconstructed into 3D points according to the labels l_p and l_q of their corresponding superpixels and the distance between the two 3D points defines $d(p,q)$. These pixels are also used in the definition of function ∇I, as it is the distance between their RGB colours.

Experiments showed that besides being dramatically faster than the standard MRF formulation, this approach is able to provide proper planar segmentations of the images. However, since superpixels are a coarse approximation to pixels, there are cases in which the labelling near the transitions of planes has faults, and thus a more sophisticated formulation is required.

B. Improvement by Adding Lines

Given the information about the planes in the scene provided by the pipeline, we propose to use the lines of intersection between the estimated planes to ensure correct transitions in the labelling. The 3D lines are projected on the image and the superpixels that they intersect are subdivided. In order to force label transition in the subdivided superpixels, the smoothness term is reformulated by multiplying a new function $f(p,q)$ by the third expression of the branch function in Equation 4. $f(p,q)$ is equal to a constant, lower than one, if the centroids of the superpixels p and q are separated by one of the lines projected on the image. This constant forces the transition between planes, because if the labels assigned to neighbour superpixels are different and a line separates them, the energy is reduced. The constant can be tuned to force more or less the transitions.

The proposed subdivision of the superpixels allows to locate the plane transitions and thus delineate them in the image, even when is has no clear edges. The improvement

comes from the fact that these superpixels are more powerful than the conventional ones as they are obtained from both 3D scene and 2D image information.

C. Dense Labelling Experiments

The proposed MRF formulation is compared to the original one by assessing their accuracies using the 6 test image pairs for which a manual planar segmentation, considered as ground truth, was performed, as shown in the top row of Fig. 4. In order to provide a fair comparison, the same camera motion and planes are used as input to both MRF approaches. Fig. 4 shows the obtained labelling, providing a qualitative assessment of the segmentation results. On the bottom right corner of each labelling image, the average Jaccard index computed for all plane labels is shown. The Jaccard index is the ratio between the number of pixels equally labelled (in the ground-truth and in the evaluated labelling) and the number of pixels of the union set. When the index is equal to one, the ground-truth and evaluated labellings are equal. This provides a quantitative evaluation of the methods. The execution time of each method, in seconds, for each test pair, is shown on the top left corner of the images.

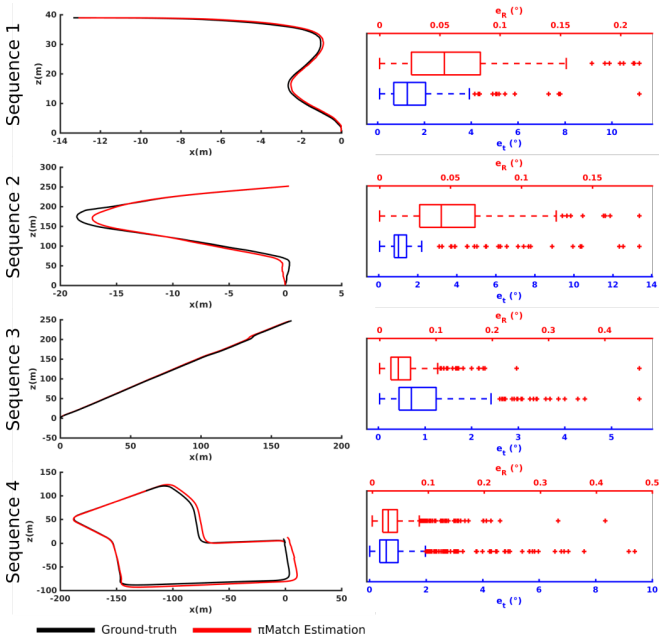
The proposed MRF approach has an average computational time of 0.707±0.113 s, being 40 times faster than the original MRF whose execution time is 28.296±9.084 s. This dramatic improvement in the computational efficiency is crucial for vSLAM applications that require online execution. When comparing the labelling accuracy of both MRF methods, it can be seen that the new formulation is superior to the original one as it provides a higher Jaccard index for all images. A more careful analysis of the obtained labellings also shows that the proposed approach is more effective in discarding non-planar objects that appear close to the camera.

This experiment demonstrates that the proposed MRF formulation is superior to the original one, being able to provide better segmentation results in a fraction of the time. This makes the proposed method an important alternative for planar segmentation schemes, being useful both in pipelines that require online execution and applications that can run offline.

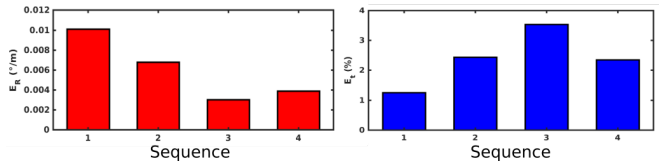
V. Experiments in Large-Scale Sequences

This section reports experiments on 4 sequences of the KITTI dataset [16], with different lengths, to assess the accuracy of the πMatch pipeline updated with the proposed modifications to two of its modules. Instead of running in a sequential manner as in [8], and in order to achieve better computational performance, another modification is performed to the architecture of the pipeline. It consisted in organizing the pipeline in a multi-thread structure implemented in C++, where the modules of AC extraction, plane-based Structure from Motion (SfM), discrete optimization and MRF are executed in distinct threads. The computational times reported in this section were measured with this modification.

Experiments were performed with the ACs extractor configured to divide the image into 16 blocks and to limit



(a) Obtained trajectories and per-pair motion errors.



(b) Motion errors using the metrics presented in [16]

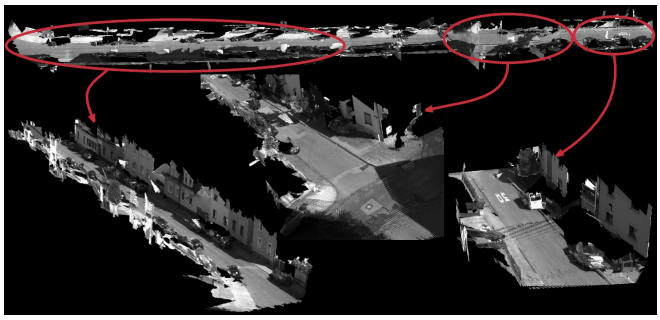
Fig. 5: Trajectories and obtained motion errors for 4 sequences of the KITTI dataset, with different lengths: Sequence 1 - 125 frames, Sequence 2 - 268 frames, Sequence 3 - 395 frames and Sequence 4 - 1101 frames.

the maximum number of outputted ACs to 800, for computational efficiency. In order to numerically evaluate the global performances for the analysed sequences, the average rotation (E_R) and translation (E_t) errors are computed using the error metrics proposed in [16]. The distribution of rotation (e_R) and translation (e_t) errors, computed as explained in Section III-A using the ground truth, and the obtained trajectories are also shown in Fig. 5.

Analysing the results, it can be seen that they are similar to the ones reported in [8] for the first 3 smaller sequences. However, for the 1101-frame sequence, the new pipeline was able to outperform the original one, providing a significantly smaller scale drift. This indicates that the new module for AC extraction,VLFeat Accelerated, provides higher quality ACs than the original one, benefiting the scale estimation, which is a difficult problem in monocular vSLAM/SfM.

Fig. 6 shows the reconstruction results obtained with the new MRF formulation for sequences 3 and 4. The high quality dense PPRs that were obtained are not only due to the good segmentation ability of the proposed MRF, but also because the planes in the scene are very accurately estimated.

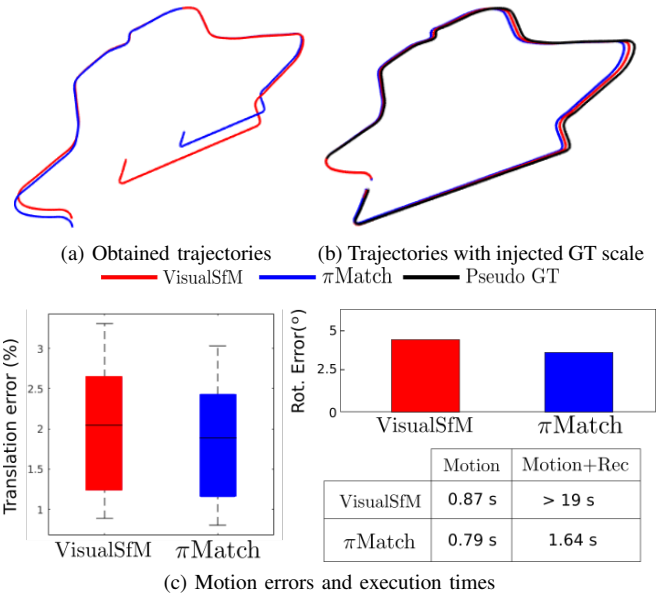
Using the new version of the pipeline, inserted in a multi-thread architecture, it achieves an average computational time



(a) Sequence 3 (395 frames)

(b) Sequence 4 (1101 frames)

Fig. 6: Reconstruction results for Sequences 3 and 4. Some areas are shown in greater detail for better visualization.



(a) Obtained trajectories (b) Trajectories with injected GT scale

	Motion	Motion+Rec
VisualSfM	0.87 s	> 19 s
πMatch	0.79 s	1.64 s

(c) Motion errors and execution times

Fig. 7: (a) Trajectories provided by the proposed version of πMatch and monocular VisualSfM and (b) the same trajectories with their relative scales corrected using the GT scales. (c) Loop-closing errors computed as described in [13] for the trajectories shown in (b) and execution times per frame.

of 1.151 s per frame, being several times faster than the original, as shown in Fig. 1. If the discrete optimization of planes and the MRF segmentation steps are removed, the method achieves an average time of 0.719 s per frame.

The average motion per frame in the KITTI dataset is approximately 0.5 m. In order to assess the performance of the proposed pipeline in a more challenging sequence, the loop closing trajectory presented in [13] was used. This is a 1370-frame stereo sequence acquired with a moving car that travelled 1100 m, having an average displacement of 0.8 m per frame. Although the average per-frame displacement is not very high, the vehicle travelled more than 0.8m in more than 40% of the trajectory. Not only the wide baselines but also the fact that this sequence has many curves and variations in altitude make it very challenging, especially in the estimation of scale.

The performance of the proposed approach is compared with the sophisticated point-based algorithm VisualSfM [17]. The monocular sequence resulting from the acquisition of the left channel is fed to both methods and results show that both of them have difficulty in handling the wide baselines, providing a high loop-closing error (Fig. 7a). In order to identify the source of error, and since this is a stereo sequence, the complete dataset (left and right channels, and extrinsic calibration of the stereo rig) was used as input to the VisualSfM pipeline, and the estimated motions are considered as a pseudo ground truth. The relative scales of the pseudo GT are injected in the trajectories estimated by the monocular VisualSfM and πMatch, and the resulting paths are depicted in Fig. 7b. It can be seen that all three trajectories are almost identical, demonstrating the accurate estimation of the rotation and direction of translation by our method. The errors computed using the loop-closing error metrics described in [13], obtained with these modified trajectories, are shown in Fig. 7c, as well as their computational times, per frame. Results show that the methods are equivalent in terms of accuracy, having the important difference that πMatch is significantly faster than VisualSfM. In the case of pure motion estimation, without dense reconstruction, πMatch provides not only the camera motion but also the 3D planes in the scene, as opposed to VisualSfM that outputs a sparse point cloud. When dense reconstructions are required, πMatch is more than an order of magnitude faster than VisualSfM, while providing much more visually pleasant 3D reconstructions. Fig. 8 evinces this fact by showing the 3D reconstructions provided by πMatch and VisualSfM, where some areas can be seen in greater detail. VisualSfM's dense reconstructions are typically populated by noisy 3D points and contain gaps where texture is low. On the other hand, πMatch provides clean reconstructions, where only the structural planes are shown. We believe that the ability to produce such dense and accurate 3D models from monocular sequences is an interesting advance in the literature, especially since this is performed at nearly 1fps. Note that such a scheme could be used for online indoor reconstruction of environments, by mounting a camera on a robot travelling at about 1m/s.

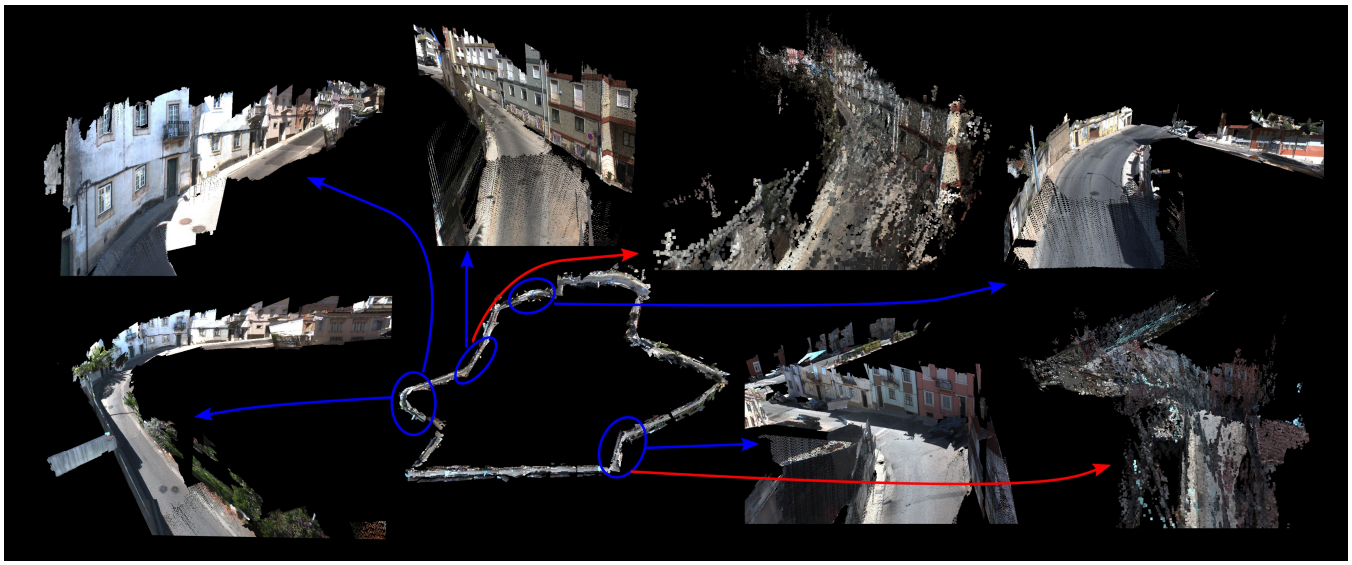


Fig. 8: 3D reconstructions of the 1370-frame loop closing sequence obtained with πMatch (blue arrows) and VisualSfM (red arrows), where some areas are shown in greater detail for better visualization.

VI. Conclusions

This paper advances the state-of-the-art in monocular vSLAM by improving πMatch, which is a recent feature-based algorithm that provides accurate PPRs of the scene. The improvements consisted in modifying two of its modules, yielding higher quality ACs, and dramatically faster planar segmentations, up to the point that the pipeline can be used in online applications. Although the problem of robust scale estimation is still unsolved, experiments show that the quality of the extracted ACs leads to lower scale drifts for medium baseline sequences, when compared to the original pipeline. Drifts in scale occur in more challenging sequences and, as future work, we intend to use the detected planes, which are more constant over time than points, to tackle this problem. A possible idea is to perform a final optimization for correcting the scale drift using loop closing and information about the planes that are shared across frames.

References

[1] H. Lategahn, A. Geiger, and B. Kitt, "Visual SLAM for autonomous ground vehicles," in *ICRA*, 2011, pp. 1732–1737.

[2] G. Ros, A. Sappa, D. Ponsa, and A. M. Lopez, "Visual slam for driverless cars: A brief survey," in *Intelligent Vehicles Symposium (IV) Workshops*, vol. 2, 2012.

[3] S. Choi, Q.-Y. Zhou, and V. Koltun, "Robust reconstruction of indoor scenes," in *CVPR*, June 2015.

[4] G. Klein and D. Murray, "Parallel tracking and mapping for small AR workspaces," in *2007 6th IEEE and ACM International Symposium on Mixed and Augmented Reality, ISMAR*, 2007.

[5] R. Mur-Artal and J. D. Tardos, "ORB-SLAM2: An Open-Source SLAM System for Monocular, Stereo, and RGB-D Cameras," 2017.

[6] R. A. Newcombe, S. J. Lovegrove, and A. J. Davison, "DTAM: Dense tracking and mapping in real-time," in *Proceedings of the IEEE International Conference on Computer Vision*, 2011, pp. 2320–2327.

[7] J. Engel, T. Schöps, and D. Cremers, "LSD-SLAM: Large- Scale Direct Monocular SLAM,," in *European Conference on Computer Vision (ECCV)*, vol. 8690, 2014, pp. 834–849. [Online]. Available: http://link.springer.com/10.1007/978-3-319-10605-2

[8] C. Raposo and J. P. Barreto, "πMatch: Monocular vSLAM and Piecewise Planar Reconstruction Using Fast Plane Correspondences," pp. 380–395, 2016.

[9] M. Antunes, J. P. Barreto, and U. Nunes, "Piecewise-planar reconstruction using two views," *Image and Vision Computing*, vol. 46, pp. 47–63, 2016.

[10] A. Bódis-Szomorú, H. Riemenschneider, and L. V. Gool, "Fast, approximate piecewise-planar modeling based on sparse structure-from-motion and superpixels," in *CVPR*, 2014, pp. 469–476.

[11] C. Raposo and J. P. Barreto, "Theory and Practice of Structure-From-Motion Using Affine Correspondences," *CVPR*, 2016.

[12] P. Neubert and P. Protzel, "Compact watershed and preemptive SLIC: On improving trade-offs of superpixel segmentation algorithms," in *ICPR*, 2014, pp. 996–1001.

[13] C. Raposo, M. Antunes, and J. P. Barreto, "Piecewise-planar stereoscan: Sequential structure and motion using plane primitives," *TPAMI*, pp. 1–1, 08 2017.

[14] A. Delong, A. Osokin, H. N. Isack, and Y. Boykov, "Fast approximate energy minimization with label costs," *IJCV*, vol. 96, 2012.

[15] A. Vedaldi and B. Fulkerson, "VLFeat - An open and portable library of computer vision algorithms," *Design*, vol. 3, 2010.

[16] A. Geiger, P. Lenz, and R. Urtasun, "Are we ready for autonomous driving? the KITTI vision benchmark suite," in *CVPR*, 2012.

[17] C. Wu, "Towards linear-time incremental structure from motion," in *Proceedings of the 2013 International Conference on 3D Vision*, ser. 3DV '13. Washington, DC, USA: IEEE Computer Society, 2013, pp. 127–134.