2013 IEEE/RSJ International Conference on
Intelligent Robots and Systems (IROS)
November 3-7, 2013. Tokyo, Japan

eVO: a realtime embedded stereo odometry for MAV applications

Martial Sanfourche†, Vincent Vittori‡ and Guy Le Besnerais†

Abstract— **The navigation of a miniature aerial vehicle (MAV) in GPS-denied environments requires a robust embedded visual localization method. In this paper, we describe a simple but efficient stereo visual odometry algorithm, called eVO, running onboard our quadricopter MAV at video-rate.**

The proposed eVO algorithm relies on a keyframe scheme which allows to decrease the estimation drift and to reduce the computational cost. We study quantitatively the influence of the main parameters of the algorithm and tune them for optimal performance on various datasets.

The eVO algorithm has been submitted to the KITTI odometry benchmark [1] where it ranks first at the date of submission, with an average translational drift of 1.93% **and an average angular drift of less than** 0.076 **degres/m. Besides, we have made several experiments with our MAV with egolocalization given by eVO, for instance for autonomous 3D environment modeling.**

I. INTRODUCTION

For an automatic agent (autonomous vehicle, robot), the navigation task requires the ability to compute an achievable trajectory and to control the execution of the path following, for example by minimizing the distance between the current estimated agent localization and the goal to reach defined as a 3D point. For this, an accurate estimate of the current localization of the agent in a fixed coordinate frame is essential.

Nowadays, vehicles equipped with high resolution inertial measurement unit (IMU) and high grade GPS receiver can achieved high accuracy self localization. However, these solutions are not suited for small vehicles with very limited payload, such as MAV. Other solutions may consist in installing tag landmarks [2] or external vision-based localization systems such as Vicon®. Although very accurate, such solutions reduce drastically the range of the agents. To access unprepared areas and GPS-denied environment (ie. where GPS measurements are not available or can not be trusted: indoor, urban canyons, etc.), the most versatile solution consists to equip the agent with vision sensor(s) eventually coupled with a low-grade IMU. This requires an accurate vision-based localization algorithm.

Many sensor configurations and a huge number of solutions have been studied and proposed. In the indoor context, the problem of 2D navigation based on a laser-rangefinder is commonly considered as solved. The arrival of low-cost RGBD cameras combined with GPU has led to a reemergence of dense and direct localization processes based on ICP approaches with impressive results [3]. Finally, binocular and monocular passive approaches have reached a high level of maturity in the last decade. For example, the two NASA's Mars Exploration Rovers (Spirit and Opportunity) have been using successfully stereo visual odometry since 2003.

From a methodological point of view, we distinguish the visual odometry (VO) approach, term popularized by Nister et al. [4] even if seminal works dates from the 80s [5], [6] and visual Simultaneous Localisation And Mapping (SLAM) approach, popularized by Davison [7]. The VO techniques focuses on ego-motion estimation and integrates motion to get an estimate of the trajectory, while the SLAM approach builds a landmark map against which the agent localizes itself.

Our objective is to define a stereo-based self localization system able to work at video frame-rate (15 to 25Hz) on the embedded computer of our Ascending Technologies Pelican depicted in figure 1. In order to do so, we have developed a simple but efficient stereo algorithm, called eVO, which retains good properties of both VO and SLAM approaches.

We have conducted a thorough parametric study of the performance of the algorithm on various datasets, either acquired by our own stereorig most often on the MAV, or on stereo datasets related to autonomous navigation of a car in a urban environment, provided by the KITTI website [1]. These data have trajectories length ranging from 400m to 5km and are made of 300 to 5000 frames. We have observed an average drift of less than 2% on the translation estimates and less than 0.01 degres/m of average rotational drift. These performances allow the proposed eVO algorithm to be ranked first on the online KITTI odometric benchmark at the date of submission.

The paper is organized as follows. The section II briefly reviews existing approaches. The proposed system is described globally in the section III, while the section IV and V focus on the implementation of its components. The methodology of evaluation is discussed in section VI. Discussions about the adopted keyframe VO scheme and about the lessons learned from our parametric studies are exposed respectively in section VII and VIII.

II. RELATED WORK

Visually aided localization is an active research field and a huge amount of literature exists. In this section, we discuss recent advances in the stereo VO and stereo visual-SLAM areas. For a more detailed review, we advise the reading of the recent two-parts tutorial of Scaramuzza and Fraundorfer [8], [9].

† ONERA - The French Aerospace Lab, Palaiseau, France firstname.name@onera.fr

‡ ONERA - The French Aerospace Lab, Palaiseau, France now at MARUM, University of Bremen, Germany vittori.vincent@gmail.com

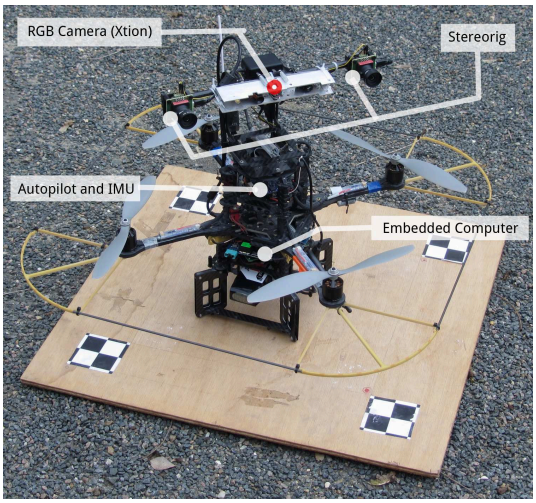

Fig. 1. Our AscTec Pelican MAV on its landing pad. The stereo-baseline is 28 centimeters long.

The VO scheme based on 2D-3D or 3D-3D matching is known since 30 years. Several improvements have been proposed during the last decade. In [4], Nister introduces the concept of reference frame in VO, a concept called keyframe in the monocular SLAM of [10]. In [11], an innovative pose estimation algorithm is proposed, by segmenting distant 3D points (used for rotation estimation) and closer ones (used for translation). Some authors take advantage of the quadrifocal [12] or trifocal [13] relationships to estimate the camera movement.

VO estimations are often combined with a multi-view optimization process. In [14], the motion estimated by VO is used as the prior to a stereo EKF-SLAM. In [15], a sliding window bundle adjustment is initialized by the VO outputs. At a higher level, an EKF filter fuses visual estimation and inertial measurements. The Relative Bundle Adjustment by Sibley and Mei [16] is a graph-based SLAM approach. Each node of the graph corresponds to an camera pose while an edge is the relative motion between two poses. In [17], the same authors present results obtained on a 2 kilometers trajectory.

Previous solutions have been first demonstrated on terrestrial robots, but stereovision systems are now available on MAV too. In this context, the problem is to deal with the limited payload and computational resources onboard. In [18], the authors describe an EKF filter fusing VO measurements and inertial ones. The data are acquired from their MAV but are processed offline on a ground station. In [19], the stereorig is combined with a scanning range finder. Stereo provides relative motions between successive views which are then fused with lidar data in an EKF-SLAM filter. The data processing is realized off-board, on the ground station.

More recently, the PixHawk team in [20] have demonstrated the ability of their quadrotor platform to explore autonomously an unknown environment. A visual odometry algorithm runs onboard. Off-board, the ground-station processed the send stereo-pair in order to model the environment. The MAV trajectory is refined here by a SLAM algorithm based on the g2o framework [21].

We have adopted a similar approach. However the subject of [20] is not odometry but mapping for autonomous exploration. In contrast we focus here on the stereo visual odometer, so as to optimize its performance and to provide evaluations of its accuracy.

III. SYSTEM OVERVIEW

The proposed system is called eVO for *"embedded Visual Odometer"*. Contrary to the "dead-reckoning visual odometer" (DRVO) [22], [23] based on the combination of frame-to-frame motion, eVO uses a persistent map containing 3D landmarks localized in a global frame, as in SLAM approach. We focus here on the way adopted to manage the map.

Initially empty, the map is built on the fly by adding or deleting 3D landmarks when it is necessary. Inspired by the solutions proposed in [10], [24], eVO adopts a keyframe-based scheme. It consists in distinguishing automatically some frames, the keyframes, used to update the landmark map which serves for ego-localization. In contrast, "standard" frames are only used to track the landmarks in the video and to localize the camera with respect to the current landmark map. These scheme is very computationally efficient as the processing of a "standard" frame needs much less operations than a keyframe (see figure 2), reducing drastically the processing time (table I).

Our approach differs from the SLAM approaches of [10], [24] in the lack of the stage of multi-view refinement of the map (by a bundle adjustment for instance). In others words, the 3D localization of a landmark, set during its initialization, can not be updated; it is set once and for all. This choice may seem crude but (i) in our stereo configuration, the 3D localization is instantaneous and the structure-from-motion approach is less crucial than in monocular case; (ii) we are constrained in terms of computational power and the multi-view refinement is quite expensive. This aspect brings eVO closer to dead-reckoning approach. While DRVO combines frame-to-frame motion integrating measurement noise at each frame, we integrate a localization error only at each keyframe. The advantage: the drift in the trajectory estimation grows more slowly.

In summary, eVO is an hybrid object, halfway between a visual SLAM solution (persistent map) and a DRVO (integration of error measurement without multi-view refinement process). The structure of our algorithm is depicted in the block-diagram of the figure 2. More details on each components will be given in the next section.

IV. DETAILED DESCRIPTION OF EVO

Here we are giving a detailed description of eVO components. Note that OpenCV is used for the most part of the low level image processing (features detector and tracking, template matching).

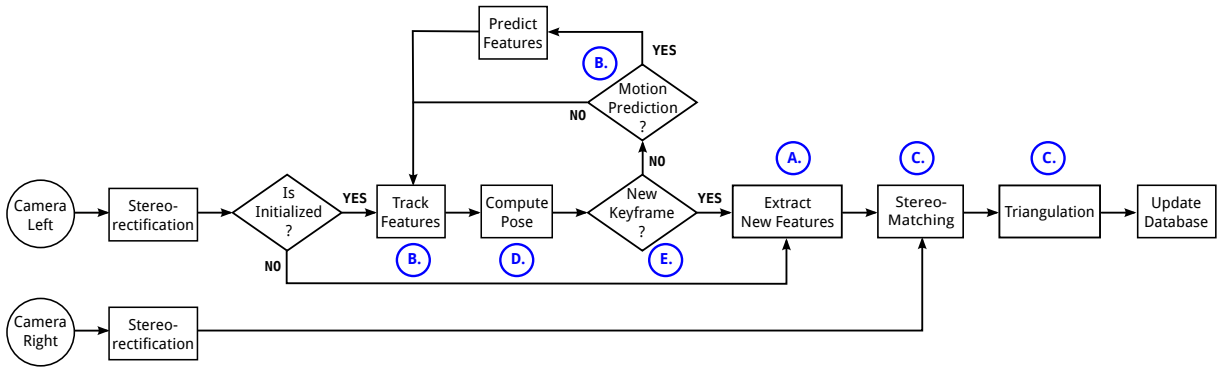

Fig. 2. Architecture of eVO algorithm. The letters surrounded by a circle refer to the subsections in the part IV.

A. Feature detection

Our visual stereo odometer is based on sparse features tracked through the video sequence. The tracking is initialized by a features detector module which is designed to maintain a pre-defined amount of features to track $\mathscr{N}_f$ (between 250 and 350 for VGA images). When a new keyframe is detected, most of the features have been successfully tracked and the module initializes as many features as it is necessary to reach the appropriate number of features ($\mathscr{N}_f$).

The extraction process takes into account successively two geometrical constraints: (i) a minimal separation distance between two candidates feature; (ii) a maximal dispersion over the image plane. The former constraint is generally included in the feature extractors. For the latter one, we adopt the classical "bucketing" strategy: the image support is subdivided in $\mathscr{N}_r$ non-overlapping regions (8×6 regions for VGA images) and the $\frac{\mathscr{N}_f}{\mathscr{N}_r}$ more relevant features inside each region are kept. To deal with homogeneous regions, a relaxation technique is used to increment the amount of maximum features by region.

Two kinds of feature detector have been evaluated: the Harris detector (Shi-Tomasi [25]) and the FAST detector [26]. The impact of the feature type is studied in section VIII.

B. Temporal matching

As in [27], the features are tracked through the video sequence acquired by the left camera using KLT [25]. Note that the tracking is done between successive views contrary to what the keyframe approach might suggest.

In order to prune pre-emptively wrong temporal matchings, the fundamental matrix is robustly estimated thanks to Least Median of Squares scheme (LMedS) [28]. As this estimation can be unstable in case of small relative motion, this step is automatically disabled when the features motion is less than a threshold.

We have also evaluated an active search process, where we use a prediction of the motion to guide the search for temporal matches. Without inertial data, as for instance in the KITTI datasets, we use a simplistic motion prediction model supposing constant linear and angular speeds. The motion estimated between the two previous frames is then used for motion prediction. If inertial data are available (as for instance in our MAV), we only compensate a global rotation of the image. Both methods helps to reduce the search area for temporal matching.

C. Stereo matching and Triangulation

The features detected in a keyframe are localized in by 3D triangulation in the global frame, using the current camera pose estimate, after the stage of stereo-matching.

Inspired by dense stereovision algorithms, stereo matching is done by exhaustive search along the epipolar lines. In practice, we test a range of disparities corresponding to 3D points located at least 70 centimeters from the stereorig and the Zero-mean Normalized Cross-Correlation (ZNCC) is used as the image similarity criterion. A similar approach is proposed in [15].

In order to reduce the processing time, we adopt a coarse-to-fine multi-scale approach with a two-levels image pyramid. At the lowest resolution, the image is reduced by a factor 4 in each direction and the size of the ZNCC window is set to 3×3 pixels. The matching decided at this level is propagated to the full resolution level. Here a local research is done in a region of radius 6 pixels with a 9×9 ZNCC window. In our configuration, this approach permits to reduce the number of tested hypotheses from 220 to 68.

In a final step, a threshold is used to prune ambiguous associations.

D. Camera pose estimation

From the temporal matchings provided by KLT, associations are established between 3D landmarks stored in the map and current image features. Given these 2D-3D matchings, the camera pose (position and attitude) is in a robust manner within a RANSAC procedure [29]. As is customary, the process takes place in two stages.

In a first time, the RANSAC procedure returns an initial solution and a set of inliers. The pose is estimated with the 3-Point algorithm [29], [30]. In practice, we have implemented our own RANSAC framework with an online adaptation of the number of iterations as proposed by Peter Kovesi [31] and a bucketing strategy to assure a minimal separation distance

Features	SHI-TOMASI [25]		FAST [26]	
Frame type	Keyframe	Standard	Keyframe	Standard
Average (ms)	74.24	12.36	56.08	12.42
Std (ms)	4.62	3.55	5.63	3.61
Min (ms)	62.27	5.75	40.78	5.56
Max (ms)	99.93	32.78	72.75	31.45

TABLE I

PROCESSING TIME OF EVO FOR ONE 672×480 STEREO PAIR ON A CORE2DUO 1.86GHZ. MEASURES OBTAINED BY AVERAGING ON 10 MONTE-CARLO RUNS.

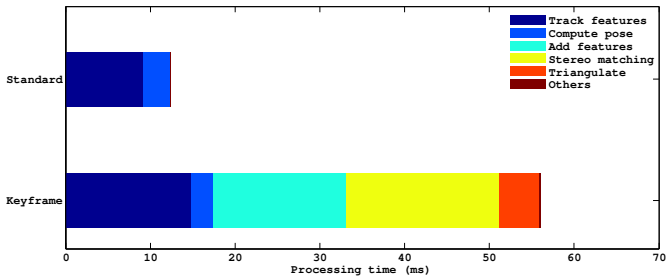

Fig. 3. Relative computing time of eVO components. Measures made by averaging over 10 monte-carlo runs using FAST features detector [26].

between the image features selected in the triplet given to P3P algorithm. The P3P method produces generally multiple solutions. To deal with this drawback, all the solutions are considered as many as random sampling in the RANSAC voting process.

In a second time, the pose is refined by non linear least-squares optimization using the inliers detected by RANSAC. This part relies on the SBA code [32]

E. New keyframe generation

As proposed in [10], a new keyframe is initialized as soon as the ratio between the number of successfully tracked features and the number of 3D points visible on the last keyframe drop under a threshold, denoted by τ, set by default to $\tau = 0.8$. We discuss the algorithm sensitivity to parameter τ in section VII.

V. IMPLEMENTATION ON A MAV AND PROCESSING TIME

The eVO version embedded on the MAV works on ROS (Robot Operating System, www.ros.org) and exploits the multi-threading capacity offered by the two processing units of the Intel Core2Duo®. Despite the linear form of the block-diagram in figure 2, some tasks can be executed in parallel. For instance, in the case of a keyframe, the rectified left image can be processed simultaneously to track features and to find new feature candidates. The price to pay is a rearrangement of the processing flow and a synchronization process in order to select the correct number of new features. In practice, this coarse parallelization allows to reduce the processing time by approximately 8ms.

We present in Table I the processing times measured on the embedded computer (Ascending Technologies Mastermind Intel Core 2 Duo 1.86 Ghz working on Ubuntu 12.04 32bits). Table I first illustrates the huge difference between keyframes and standard frames processing time, due to the fact that for the latter ones, the 3D landmarks generation is bypassed. Regarding the different features detectors, the gain is significant, with 5 to 6 factor in favor of FAST.

Using FAST, we then obtain a mean computation rate better than 20Hz. The influence of the features detector on the localization performance will be discussed later on.

The figure 3 shows how the computational time is distributed over the different components of the processing chain. For a standard frame, tracking features consumes most of the processing time (75%). For a keyframe, the computing time is equally distributed between the various tasks.

VI. GENERAL EVALUATION

A. Datasets and performance measures

Our system has been evaluated on multiple and varied data. Some were acquired using our own stereorig, either hand-held or carried by the MAV. No ground-truth state is available for these data but we have followed closed trajectories in order to use the drift between the first and last frames as a performance indicator. An example of such experiment is presented in figure 4.

We have also used the KITTI's odometry dataset [1] composed of 22 video sequences acquired by a car equipped with several sensors (Velodyne ® lidar, high resolution IMU and GPS-RTK, stereorig). The collection of video covers a large range of environment (highway, suburban or town center) and trajectory profiles (loops, road sections) from one hundred meters to a few kilometers. The first half of the collection is supplied with ground-truth in order to adjust the parameters of algorithms. The second half of the collection is used to benchmark algorithms in a blind manner.

The KITTI Team provides also some performance metrics together with a tool to compute them on the estimated trajectories. These metrics are: a translational drift expressed in percentage of the total traveled distance and a rotational drift expressed in terms of degree by traveled meter. Scores are are averaged on all possible sub-sequences of variable lengths, from $5m$ to $400m$.

As our system includes random sampling scheme (RANSAC) we have done Monte-Carlo simulations and measured statistical indicators (average performance, standard deviation, median, min-max values).

B. Result on the KITTI Odomety Benchmark

The figure 5 presents the estimated trajectories obtained after 25 Monte-Carlo runs on the sequence 08 of the KITTI odometry dataset. This trajectory in suburban environment is 2 kilometers long and comprises many moving objects (vehicles, pedestrians and cyclists). In average, the estimated trajectory in the horizontal plane (XZ) is well estimated with a drift of only 4 meters. As usual in odometry, large angular errors occur at each important turn change. The estimation

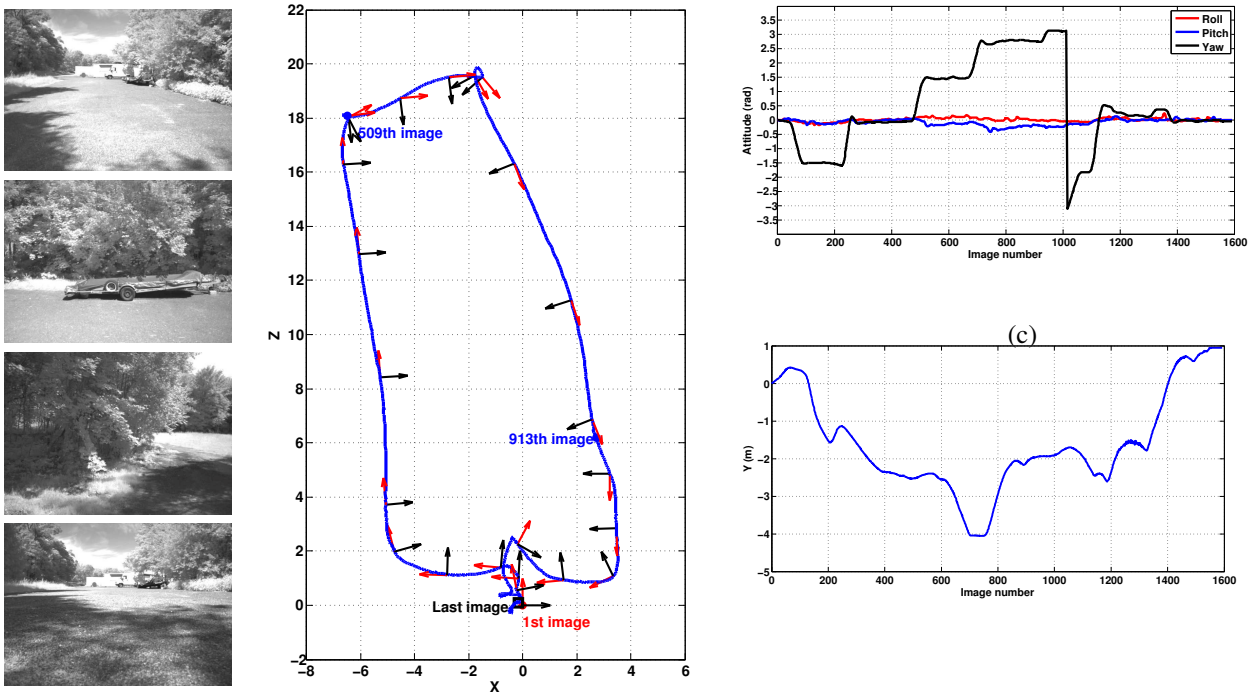

Fig. 4. Trajectory estimated by eVO from the sequence 20120727.3 acquired during an outdoor flight of the MAV. (a) 4 frames of the video sequence (the 1st, 509th, 913th and last image). (b) Estimated trajectory. The red and black arrows indicate the attitude of our MAV (red : the front of the MAV, black : its right). (c) Estimated attitude. The measurements provided by the embedded AHRS are not precise enough to serve as ground truth. (d) Estimated height profile (the Y axis points downward). Note that the starting point is approximately 80cm above the landing pad, hence the total drift is lower than 50 centimeters.

along the third dimension shows a bias at the beginning which is probably due to an error in the ground truth, and a significant variance at the end. We could constrain the eVO estimator to keep a constant height above the ground, but we choose not to do so, as we intend to use the same algorithm for MAV data.

At the date of submission, eVO ranks first on the Kitti Odometry benchmark with an average translation drift of 1.93% and a angular error of $0.0076^\circ/m$, as shown in the Table II. The submitted version of eVO was configured as follows:

- Feature detector: Shi-Tomasi
- Feature tracked: 480 features
- Image bucketing: 20 by 8 regions
- Active search: enabled
- Ransac threshold: 1.0
- landmark ratio for keyframe decision: $\tau = 0.8$

VII. EVALUATION OF THE KEYFRAME SCHEME

We discuss here the advantages provided by the key-frame scheme in the ego-localization performance, beyond the computational efficiency discussed previously. First, we compare our algorithm with a classical dead-reckoning visual odometer (DRVO) build with the same software components; then, the influence of two relevant parameters is evaluated. All the results obtained on MAV sequences are summarized in Tables III, IV, while results on KITTI dataset are shown in Table V.

A. eVO vs. DRVO

The main difference between eVO and DRVO concerns the integration of the estimated motion. eVO integrates the motion through the addition of new landmarks in the onboard map at each new keyframe, while DRVO integrates motion itself without referring to a map. Note however that, contrarily to SLAM, eVO doesn't update the localization of the landmarks previously recorded in the map when acquiring a new stereo pair.

As expected, the keyframe scheme permits to reduce the localization drift, even for settings which favor the generation of new keyframes. This is the case when choosing $\tau = 1.0$, which means generating a new keyframe as soon as one landmark is lost by the tracking process: the performance of this setting are presented in the second lines of the Tables. The gain is particularly important with MAV data as shown by comparing the total localization error presented in the two first rows of tables III and IV. On the KITTI dataset, the benefit is less significant with a 10%-reduction of the drift, see Table V.

B. Parameters controlling the key-frame selection

Keyframe generation is related to the ability to track successfully landmarks. It depends evidently on the sequence's framerate, on the trajectory (speed, number of turns) and on the distance to the scene (normalized by the baseline). It depends also on algorithmic parameters such as the threshold on the ratio of landmarks tracked since the last keyframe (τ)

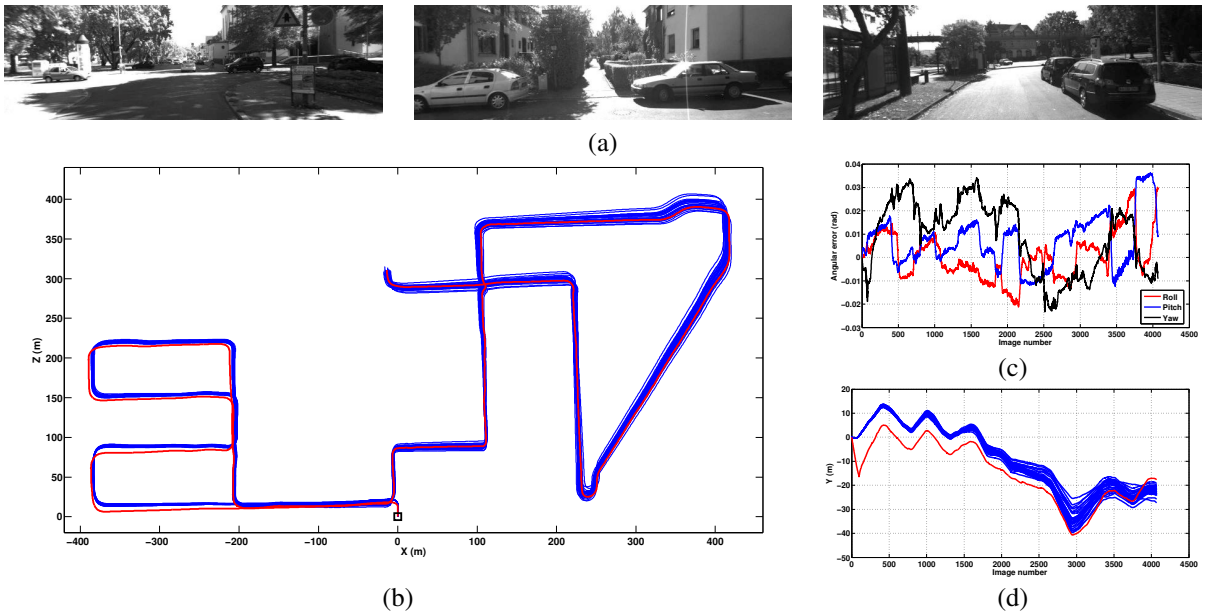

Fig. 5. Result of eVO on the sequence "08" of the KITTI odometry dataset. (a) Three images of the sequence. (b) Trajectories on the XZ plane (red: ground truth, blue: estimated). In red the ground truth. In blue, 25 trajectories obtained after as many monte-carlo runs. (c) Average angular errors (in radians). (d) Trajectories in the 3rd dimension.

Rank	Method	Translation	Rotation	Runtime	Environment
1	eVO	1.93 %	0.0076 [deg/m]	0.05 s	2 cores @ 2.0 Ghz (C/C++)
2	D6DVO	2.10 %	0.0083 [deg/m]	0.03 s	1 core @ 2.5 Ghz (C/C++)
3	MFI	2.14 %	0.0105 [deg/m]	0.1 s	4 cores @ 3.0 Ghz (C/C++)
4	GT_VO3pt	2.21 %	0.0117 [deg/m]	1.26 s	1 core @ 2.5 Ghz (C/C++)
5	VISO2-S	2.27 %	0.0152 [deg/m]	0.05 s	1 core @ 2.5 Ghz (C/C++)
6	BoofCV-SQ3	2.27 %	0.0111 [deg/m]	0.14 s	1 core @ 2.5 Ghz (Java)
7	TGVO	2.44 %	0.0105 [deg/m]	0.06 s	1 core @ 2.5 Ghz (C/C++)
8	SVO	2.45 %	0.0109 [deg/m]	0.05 s	2 cores @ 2.5 Ghz (C/C++)
9	VO3pt	2.93 %	0.0116 [deg/m]	0.56 s	1 core @ 2.0 Ghz (C/C++)
10	VO3ptLBA	3.17 %	0.0180 [deg/m]	0.57 s	1 core @ 2.0 Ghz (C/C++)
11	MSD VO	3.50 %	0.0166 [deg/m]	0.07 s	1 core @ 2.8 Ghz (C/C++)
12	MLM-SFM	4.07 %	0.0104 [deg/m]	0.03 s	5 cores @ 2.5 Ghz (C/C++)
13	VOFS	4.21 %	0.0158 [deg/m]	0.51 s	1 core @ 2.0 Ghz (C/C++)
14	VOFSLBA	4.35 %	0.0189 [deg/m]	0.52 s	1 core @ 2.0 Ghz (C/C++)
15	VISO2-M	13.79 %	0.0372 [deg/m]	0.1 s	1 core @ 2.5 Ghz (C/C++)

TABLE II

KITTI ODOMETRY BENCHMARK CHART AT 2013-03-19.

or the detection of erroneous temporal associations thanks to the robust estimation of fundamental matrix (a module denoted Fcheck in the following).

We first study how the drift varies with respect to the ratio τ while the Fcheck module is activated. On the MAV sequences (Tables III and IV) the lower the parameter τ, the lower the average localization error but the higher the dispersion of the results. On the Kitti dataset (Table V) we observe that the choice $\tau = 0.6$ leads to larger errors. In our opinion, this is due to a lower framerate and a higher vehicle speed, which means that odometry uses tracked features which are farther from the camera and are badly localized. Finally, we choose $\tau = 0.8$ as a good trade-off.

The Fcheck procedure has also a significant influence on the number of keyframes. If this validation step is bypassed, the number of keyframes is reduced by half in all processed sequences (for the same ratio τ). On the majority of our tests, this comes with an error growth, particularly on the KITTI dataset where the translational drift grows from 1.46 to 1.63. In practice, we choose to enable Fcheck by default.

VIII. PARAMETRIC STUDY

eVO is controlled by about 10 parameters. Among those we already talked about the Ransac threshold, the number of features to track, the feature type, the "active search" and "Fcheck" procedures or the parameter τ.

We focus here on the comparison between two features detector (FAST and Shi-Tomasi) and on the the "active search" procedure. The first comparison is interesting because the two detectors have very different characteristics : FAST is known to be more computationally efficient but less repeatable than Shi-Tomasi. The second comparison seems to us important because the procedure permits significant performance gain on some difficult sequences.

Method	Error X (m)	Error Y (m)	Error Z(m)	Keyframe number
DRVO	-2.4 ± 0.04	-0.5 ± 0.05	2.0 ± 0.04	2039
EVO $\tau = 1.0$	-1.28 ± 0.08	-0.17 ± 0.11	1.20 ± 0.08	1958
EVO $\tau = 0.8$	-0.95 ± 0.18	-0.30 ± 0.11	1.01 ± 0.14	738
EVO $\tau = 0.6$	-0.73 ± 0.21	-0.06 ± 0.25	0.83 ± 0.18	364

TABLE III

LOCALIZATION ERROR AT THE END OF THE CLOSED TRAJECTORY AND NUMBER OF KEYFRAMES FOR DIFFERENT ALGORITHMS OR ALGORITHM SETTINGS. PROCESSED VIDEO SEQUENCE : 20110727.2, A 150 METERS LONG STEREO SEQUENCE WITH 2039 IMAGES. THE STEREORIG WAS HAND-HELD.

Method	Error X (m)	Error Y (m)	Error Z(m)	Keyframe number
DRVO	-6.8 ± 0.4	2.05 ± 0.2	4.7 ± 0.4	1675
EVO $\tau = 1.0$	-0.80 ± 0.2	0.23 ± 0.24	0.45 ± 0.12	1522
EVO $\tau = 0.8$	-0.61 ± 0.34	0.23 ± 0.35	0.4 ± 0.17	448
EVO $\tau = 0.6$	-0.33 ± 0.60	0.35 ± 0.47	0.21 ± 0.3	214

TABLE IV

LOCALIZATION ERROR AT THE END OF THE CLOSED TRAJECTORY AND NUMBER OF KEYFRAMES FOR DIFFERENT ALGORITHMS OR ALGORITHM SETTINGS. PROCESSED VIDEO SEQUENCE : 20120724.3, A 70 METERS LONG STEREO SEQUENCE WITH 1675 IMAGES. ACQUIRED WITH THE MAV.

For our different experiments, we have set the Ransac threshold between 1.0 and 1.5 without major effects on the results.

Each test consists in multiple Monte-Carlo runs on 9 different video sequences (a total approximately 18000 stereo frames) for different settings of the parameters.

A. Feature detector

In the section V, we have shown the benefit of FAST detector in terms of processing time, we address here its performance in terms of trajectory estimation. The table VI shows the KITTI angular and translational drifts for the feature Shi-Tomasi and FAST detectors. The results are almost identical, hence FAST appears as the best choice.

We suppose that the effect of the lower repeatability performance of FAST is partly erased by the KLT-based features tracking.

Method	Translational Error (%)	Angular Error (%)	Keyframe ratio
DRVO	1.56 ± 0.007	$0.0166 \pm 8e^{-5}$	100%
EVO $\tau = 1.0$	1.45 ± 0.015	$0.0145 \pm 1e^{-4}$	99.8%
EVO $\tau = 0.8$	1.46 ± 0.014	$0.0144 \pm 2e^{-4}$	79.6%
EVO $\tau = 0.6$	1.53 ± 0.017	$0.0151 \pm 2e^{-4}$	37.8%

TABLE V

ANGULAR AND TRANSLATION DRIFT INDICATORS MEASURED ON KITTI ODOMETRY DATASET FOR DIFFERENT ALGORITHMS OR ALGORITHM SETTINGS.

FAST	Translational (%)	Angular drift (deg/m)
mean	1.46188	0.014459
std	0.0145939	0.000194284
median	1.4594	0.0145
SHI-TOMASI	Translational (%)	Angular drift (deg/m)
mean	1.48145	0.0145641
std	0.0129765	0.000191239
median	1.4818	0.0146

TABLE VI

ANGULAR AND TRANSLATION DRIFT INDICATORS MEASURED ON KITTI ODOMETRY DATASET FOR TWO DIFFERENT FEATURE DETECTORS. THESES STATISTICS ARE MEASURED ON 50 MONTE-CARLO RUNS WHILE OTHER PARAMETERS WERE IDENTICAL.

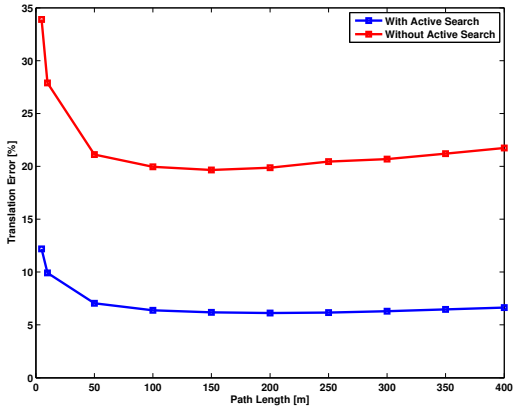

Fig. 6. Translation error in function of path length with and without active search mode. Results obtained on the sequence 01 (highway). The errors are reduced by a factor 2.5.

B. Active search

In the KITTI dataset, the active search described in Section IV-B provide a huge improvement in the global accuracy. On the 11 training sequences, the mean translational drift is reduced from 2.45% to 1.69%, and the mean angular error from $0.0175^\circ.m^{-1}$ to $0.0164^\circ.m^{-1}$. The improved temporal tracking of the features is confirmed by a 15% reduction of the number of keyframes. If we look more closely at the results, we note that the gain is mainly due to a dramatic reduction of the error on a complex video sequence (Sequnce 01 of KITTI) acquired on a highway as shown in figure 6.

In experiment with the MAV, the effect of active search (ie. rotation compensation) is less significant, except for processing time (due to reduced searching areas).

IX. CONCLUSION AND FUTURE WORKS

In this paper we have presented eVO, a carefully engineered stereo-based visual odometer, designed to run at video

rate on the embedded computer of a MAV. This objective is achieved by combining a keyframe-based scheme and outlier detection mechanisms.

The performance have been evaluated on a large collection of data, presenting different use-cases: low altitude flight of a MAV at low speed or large-scale trajectory of a car moving at different speed in various environments. In the KITTI odometry contest, eVO has been compared to a dozen of concurrent algorithms and leads this comparison chart at the date of submission.

We plan to improve eVO by fusing it with inertial measurements with the goal of reduce the angular drift: preliminary results show that significant gain can be expected. Some works are in progress in this direction.

Finally, let us mention that eVO is a part of an online 3D environment mapping system by a MAV combining on-board trajectory estimation and off-board mapping system. The depthmaps are computed from stereo pair in an optical-flow approach thanks to an improved version of the FOLKI-GPU algorithm [33]. An example of our mapping system can be watched on `http://www.youtube.com/watch?v=LuVq03Op3_M`.

REFERENCES

[1] A. Geiger, P. Lenz, and R. Urtasun, "Are we ready for autonomous driving? the kitti vision benchmark suite," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Providence, RI (USA), June 2012, pp. 3354 – 3361.

[2] E. Olson, "AprilTag: A robust and flexible visual fiducial system," in *IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, May 2011, pp. 3400–3407.

[3] S. Izadi, D. Kim, O. Hilliges, D. Molyneaux, R. Newcombe, P. Kohli, J. Shotton, S. Hodges, D. Freeman, A. Davison, and A. Fitzgibbon, "Kinectfusion: real-time 3d reconstruction and interaction using a moving depth camera," in *Proceedings of the 24th annual ACM symposium on User interface software and technology*, New York, NY (USA), 2011, pp. 559–568.

[4] D. Nister, O. Naroditsky, and J. Bergen, "Visual odometry for ground vehicles applications," *Journal of Field Robotics*, vol. 23, no. 1, pp. 3–20, 2006.

[5] H. Moravec, "Obstacle avoidance and navigation in the real world by a seeing robot rover," Ph.D. dissertation, Standford University, Standford,CA, 1980.

[6] L. H. Matthies, "Dynamic stereo vision," Ph.D. dissertation, Carnegie Mellon University, Pittsburgh, PA, USA, 1989, aAI9023429.

[7] A. Davison, I. Reid, N. Molton, and O. Stasse, "Monoslam: Real-time single camera slam," *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, vol. 29, no. 6, pp. 1052–1067, 2007.

[8] D. Scaramuzza and F. Fraundorfer, "Visual odometry: Part i - the first 30 years and fundamentals," *IEEE Robotics and Automation Magazine*, vol. 18, no. 4, pp. 80–92, December 2011.

[9] F. Fraundorfer and D. Scaramuzza, "Visual odometry: Part ii - matching, robustness, and applications," *IEEE Robotics and Automation Magazine*, vol. 19, no. 2, pp. 78–90, June 2012.

[10] E. Mouragnon, M. Lhuillier, M. Dhome, F. Dekeyser, and P. Sayd, "Real time localization and 3d reconstruction," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, vol. 1, 2006, pp. 363–370.

[11] M. Kaess, K. Ni, and F. Dellaert, "Flow separation for fast and robust stereo odometry," in *IEEE International Conference on Robotics and Automation (ICRA)*, Kobe, Japan, May 2009, pp. 3539–3544.

[12] A. I. Comport, E. Malis, and P. Rives, "Accurate quadrifocal tracking for robust 3d visual odometry," in *IEEE International Conference on Robotics and Automation (ICRA)*. Roma, Italy: IEEE, April 2007, pp. 40–45.

[13] B. Kitt, A. Geiger, and H. Lategahn, "Visual odometry based on stereo image sequences with ransac-based outlier rejection scheme," in *IEEE Intelligent Vehicles Symposium (IV)*. San Diego, CA (USA): IEEE, June 2010, pp. 486–492.

[14] P. Alcantarilla, L. Bergasa, and F. Dellaert, "Visual odometry priors for robust ekf-slam," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2010, pp. 3501–3506.

[15] K. Konolige, M. Agrawal, and J. Sola, "Large-scale visual odometry for rough terrain," in *The 13th International Symposium of Robotics Research*, Hiroshima, Japan, November 2007.

[16] G. Sibley, C. Mei, I. Reid, and P. Newman, "Adaptive relative bundle adjustment," in *Robotics Science and Systems Conference*, 2009.

[17] C. Mei, G. Sibley, M. Cummins, P. Newman, and I. Reid, "Rslam: A system for large-scale mapping in constant-time using stereo," *International Journal of Computer Vision*, pp. 1–17, 2010, special issue of BMVC.

[18] J. Kelly and G. S. Sukhatme, "An experimental study of aerial stereo visual odometry," in *IFAC Symposium on Intelligent autonomous vehicles*, 2007.

[19] M. Achtelik, A. Bachrach, R. He, S. Prentice, and N. Roy, "Stereo vision and laser odometry for autonomous helicopters in gps-denied indoor environments," in *Proceedings of the SPIE Unmanned Systems Technology XI*, vol. 7332, Orlando, Florida, 2009.

[20] F. Fraundorfer, L. Heng, D. Honegger, G. H. Lee, L. Meier, P. Tanskanen, and M. Pollefeys, "Vision-based autonomous mapping and exploration using a quadrotor mav," in *IEEE/RSJ International Conference on Intelligent robots and systems (IROS)*, Algarve, Portugal, October 2012, pp. 4557–4564.

[21] R. Kümmerle, G. Grisetti, H. Strasdat, K. Konolige, and W. Burgard, "g2o: A general framework for graph optimization," in *IEEE International Conference on Robotics and Automation (ICRA)*, Shanghai, China, May 2011, pp. 3607–3613.

[22] A. Mallet, S. Lacroix, and L. Gallo, "Position estimation in outdoor environments using pixel tracking and stereovision," in *IEEE ICRA*, 2000.

[23] A. Howard, "Real-time stereo visual odometry for autonomous ground vehicles," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2008, pp. 3946–3952.

[24] G. Klein and D. Murray, "Parallel tracking and mapping for small ar workspaces," in *International Symposium on Mixed and Augmented Reality*, Nara, Japan, November 2007.

[25] J. Shi and C. Tomasi, "Good features to track," in *1994 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 1994, pp. 593 – 600.

[26] E. Rosten and T. Drummond, "Machine learning for high-speed corner detection," in *European Conference on Computer Vision*, vol. 1, May 2006, pp. 430–443.

[27] J. Kelly, S. Saripalli, and G. Sukhatme, "Combined visual and inertial navigation for an unmanned aerial vehicle," in *Field and Service Robotics*, ser. Springer Tracts in Advanced Robotics, C. Laugier and R. Siegwart, Eds. Springer Berlin / Heidelberg, 2008, vol. 42, pp. 255–264, 10.1007/978-3-540-75404-6_24.

[28] P. J. Rousseeuw, "Least median of squares regression," *Journal of the American Statistical Association*, vol. 79, no. 388, pp. 871–880, 1984. [Online]. Available: http://www.jstor.org/stable/2288718?origin=crossref

[29] M. A. Fischler and R. C. 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.

[30] S. Umeyama, "Least-squares estimation of transformation parameters between two point patterns," *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, vol. 13, no. 4, pp. 376–380, April 1991.

[31] P. D. Kovesi, "MATLAB and Octave functions for computer vision and image processing," Centre for Exploration Targeting, School of Earth and Environment, The University of Western Australia, available from: <http://www.csse.uwa.edu.au/∼pk/research/matlabfns/>.

[32] M. A. Lourakis and A. Argyros, "SBA: A Software Package for Generic Sparse Bundle Adjustment," *ACM Trans. Math. Software*, vol. 36, no. 1, pp. 1–30, 2009.

[33] A. Plyer, G. L. Besnerais, and F. Champagnat, "Folki-gpu: A powerful and versatile cuda code for real-time optical flow computation," in *GPU Technology Conference*, San Jose, CA (USA), October 2009.