Fast vision-based relocalization for MAVs

Quim Sànchez

Abstract— **Visual odometry techniques are becoming increasingly popular in robotic vehicles as a means to provide navigation information. This is further relevant in the context of Micro Aerial Vehicles (MAVs) where the payload constrains impose strong limitations on the choice of navigation sensors. Modern Visual Odometry algorithms, although quite sophisticated, are prone to failures when the tracking is lost due to image blur or sudden large changes in the image content. The loss of the tracking requires finding the location of the vehicle with respect to a previously built map (kidnapped robot problem). In this work we study and propose some relocalization solutions for Visual Odometry algorithms. This work is intended to work with SVO (Semi-direct Visual Odometry) but the proposed framework can be used with other methods. The relocalization method from PTAM is used as a base line and new alternatives based on space geometry and machine learning are proposed.**

I. INTRODUCTION

Micro Aerial Vehicles (MAVs) are about to play a major role in tasks like search and rescue, environment monitoring, security surveillance, inspection and goods delivery (Amazon). However, for such operations, navigating based on GPS information only is not sufficient. Fully autonomous operation in cities or other dense environments requires MAVs to fly at low altitudes where GPS signals are often shadowed or indoors, and to actively explore unknown environments while avoiding collisions and creating maps. Precisely autonomous operations requires MAVs to rely on alternative localization systems. For minimal weight, power consumption and budget a single camera can be used for this propose.

Real-time monocular Visual Odometry (VO) algorithms can be used to estimate the 6 DoF pose of a camera relative to its surroundings. This is attractive for many applications such as mobile robotics (and not only aerial) and Augmented Reality (AR) because cameras are small and self-contained and therefore easy to attach to autonomous robots or AR displays. Further, they are cheap, and are now often pre-integrated into mobile computing devices such as PDAs, phones and laptops.

SVO (Semi-direct Visual Odometry) [4] is a very fast VO algorithm able to run at more than 300 frames per second on a consumer laptop. It builds a map based on keyframes and salient points. Most monocular VO are feature-based where scale and rotation invariant descriptors (SIFT, SURF...) are extracted and matched in order to recover the motion from frame to frame while finally refining the pose with reprojection error minimization with the map. SVO uses a different approach by using direct methods. Instead of matching descriptors, it uses intensity gradients to minimize the error between patches around detected salient points to estimate the frame to frame transformation. Finally, it uses Bundle Adjustment to align with the map and avoid or minimize drift.

The main problem with most existing monocular VO implementations (including SVO) is a lack of robustness. Rapid camera motions, occlusion, and motion blur (phenomena which are common in all but the most constrained experimental settings) can often cause the tracking to fail. While this is inconvenient with any tracking system, tracking failure is particularly problematic for VO systems: not only is the camera pose lost, but the estimated map could become corrupted as well.

This problem is accentuated during a fast agile maneuver (e.g., a flip) and so a good relocalization is important when these are intended to be performed.

II. RELATED WORK

A. Place Recognition

Klein and Murray present in [7] the relocalization method used in PTAM [6]. PTAM is a VO algorithm based on keyframes that are used during the relocalization. The relocalization method consists of two steps. First, given the current frame, the most similar keyframe is retrieved, and its know pose is used as a baseline. As a measure of similarity the cross correlation, being the difference between subsampled, blurred and zero-mean images is used. The small blurry images are stored every time there is a new keyframe and the small blurry image of a new frame is computed during the relocalization to be compared with the keyframes.

Other methods can be used for image retrieval, for example using bag of words [14]. Nistér and Stewenius [11] propose to use a tree structure to store words in order to handle much larger vocabulary or have a much faster retrieval. Every node of the tree would have k child nodes which are the clustering results of k-means. The tree is build by recursive k-means. This structure is expensive to build because k-means is very resource consuming. During the online process, new words can be appended to the final leaves.

Özuysal et al. [12] proposes a simplified random forest classifier which relates image patches to objects. It is

simplified because instead of using a tree structure, they use a linear structure applying all the binary tests to the patch. The result of the tests is a binary descriptor, the list of binary tests is called Fern. Every object is trained with multiple random warps of the known view to introduce information from possible different views of the object. In the end every object can be represented with many binary descriptors and every descriptor should output a probability distribution of possible objects represented. Evaluating multiple Ferns and joining the produced distributions, the final classification is achieved.

B. Pose Estimation

Geometric methods are typically used to find the transformation from the found keyframe using the classic pipeline of salient points detection, feature extraction and matching. The five-point algorithm can then be used to find the scaled 6 DoF transformation or the full 6 DoF with the three-point algorithm if depth is known [5].

During the second step of the relocalization, the transformation from the retrieved frame to the query frame must be calculated. This transformation will be finally appended to the know keyframe pose. In PTAM, an image alignment algorithm, Efficient Second-Order Minimization method (ESM) [2], is employed. ESM is a Gauss-Newton gradient descent algorithm, which can be used with different image warp functions. It is similar and based on the Lucas-Kanade [1] algorithm but using Second-order functions. Therefore, it results in a faster convergence.

C. Joint Place and Pose estimation

One approach to solve the relocalization problem was proposed by Williams [15]. In their implementation, they use Random Forest classifiers to characterize a salient object in space. To do so, the classifier needs to be trained with as many as possible representations of the object (multiple views). Therefore, the first time an object is found, multiple warps of the patch are used to initialize its presence in the classifier. On later encounters with the object, the classifier is incrementally trained with additional data. During the relocalization phase salient points are classified using the trained classifier and the three-point algorithm is used to recover the 6 DoF position. This method is memory expensive and requires a GPU to generate the patch warps.

Shotton et al. [13] also propose a method to solve the place recognition and pose estimation problem simultaneously using random forests. RGB-D data is used to train the classifier. In this case, all the information is encoded in the classifier so no previous data storing or computing (salient point detection, descriptor extraction, etc...) is needed. The classifier is trained to an individual RGB-D pixel, and an RGB-D pixel query will output a probability distribution over the position in $\mathbb{R}^3$. This can be applied to all pixels of a frame or to a sparse subset selection of them. Ideally, the camera pose can be inferred from only three pixels, but as the output of the classifier can be very noisy, a second step is applied. From the output from many pixels an energy function is minimized using preemptive RANSAC in order to find a pose that agrees with most of the distributions.

To train this method, a very complete dataset of RGB-D images with 6 DoF poses from the environment associated to them is needed. That makes it difficult to be used with SLAM problems where the map get populated incrementally. An online training method should be developed.

III. Approach

We propose two different approaches to address the relocalization proble. First, a local approache is based on the PTAM implementation, where two steps are performed. The first step has been named *Place Finder* and the second *Real Pose Recognition*. Multiple methods will be proposed to solve the second step. Then, on the other side, a global approach will be proposed. In this case machine learning methods (*ferns*) will be used to recognize points in space.

A. PTAM method

PTAM [6] is a VO algorithm based on keyframes and so the relocalization method proposed is based on keyframes as well. Every keyframe is associated with a camera pose that will be used to relocalize. During the relocalization there are two steps involved. We called the first step *Place Finder* and the second *Real Pose Finder*.

1) Place Finder: During this step, the algorithm tries to find the keyframe image most similar to the last acquired image. The pose associated with the most similar keyframe is used as an initial rough estimation of the current pose. The similarity score should be invariant to small view point changes because the new acquired image will, most probably, never be taken from the same pose as any of the keyframes. Also it should be fast to compute.

The used similarity score is the Cross Correlation between images meaning the sum of the squared error between two zero-mean images as in 1. To make to computation faster both images are resized become 40×30. Then, to make the images more resistant to view point changes it is blurred with a 3×3 Gaussian kernel with $\sigma = 2.5$. The resulting image is a resized, blurred and zero-mean image called *small-blurry-image*.

$$d_{CC} = \sum_{x,y} [(I(x,y) - \bar{I}) - (G(x,y) - \bar{G})]^2 \qquad (1)$$

2) ESM Real Pose Finder: The second step of the PTAM relocalization algorithm found, in order to refine the pose of the most similar keyframe to explain the current pose of the camera. During this step, in the implementation from PTAM, only rotations are corrected. An image alignment through optimization algorithm (ESM [9]) is used to find

the $SE(2)$ transformation between the two, followed by a minimization to find a transformation in the world frame.

The Extended Second order Minimization algorithm (ESM) [9] is based on the algorithm proposed by Lucas and Kanade [1] in 1981. The goal of both algorithms is to align one template image T to a different input image I through a parametrised warping function.

With this goal an error function is defined on which the Gauss-Newton or Levenberg-Marquardt schemes can be applied. The error is the squared difference between the template image and the warped input image

$$e = \sum_x [I(W(x;p)) - T(x)]^2 \tag{2}$$

The warping function is going to be in $SE(2)$, that is, translation and rotation of the image plane. The algorithm assumes that a current estimate of p exists and iteratively tries improves it by increments Δp. On every iteration equation 3 is solved on Δp and then it is used to update p as in eq. 4.

$$\min_{\Delta p} \sum_x [I(W(x;p+\Delta p)) - T(x)]^2 \tag{3}$$

$$p \leftarrow p + \Delta p \tag{4}$$

The solution that minimizes eq. 3 on Δp can be found in a least squares sense. The equation needs to be derived and then set it equal to zero.

The ESM algorithm used in PTAM is very similar to the derived above with the difference that while the Lucas-Kanade takes the gradient from the input image, ESM used both the gradient of the input image and the gradient of the template image and averages them.

$$\nabla I = \frac{1}{2}[\nabla I_t + \nabla I_q] \tag{5}$$

In Figure 1 the error of the overlapped images is visualized. It can be seen that translation and rotation are well corrected but there still is a misalignment caused mostly by a change on scale which is not taken into account during the alignment.

3) Alternative Real Pose Finder: During the mapping of an area, the VO algorithm finds landmarks in the world frame which are associated with detected featured points in keyframes (i.e. every featured point in an image is related to a $3D$ position in the world frame). Given a new image, some extracted featured points can be related to a keyframe using descriptor-matching which at the same time are related to world positions. From this information the full 6 DoF translation $SE(3)$ can be computed using the prospective three-point algorithm (P3P).

Fig. 1: $SE(2)$ transformation error visualization

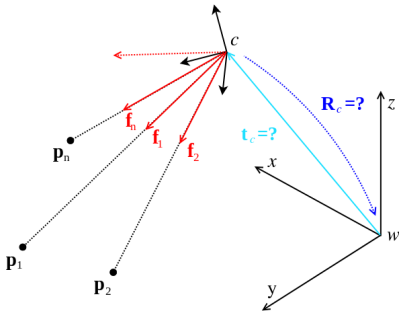

Fig. 2: From camera frame vectors f (or image pixels if the camera calibration is available) and fix points p the transformation between the two coordinate frames can be computed using the P3P algorithm. Image taken from [8]

First, descriptors from every featured point are extracted. Second, a brute force KNN matching is performed from all the extracted descriptors from on image and all the descriptors of the second image. The first, and second most similar descriptor are retrieved. Then, only good matches are kept, that is, using the matching technique described by Lowe [10], only matches with a descriptor ratio between the first and the second closest match of 0.8 or less are kept.

Finally, the three-point algorithm is fed with the pixel positions from the query image and the landmarks from the other. Because there are still outliers after the described simple filtering, this process is run in RANSAC [3] framework.

Figures 3 is an example of the described above.

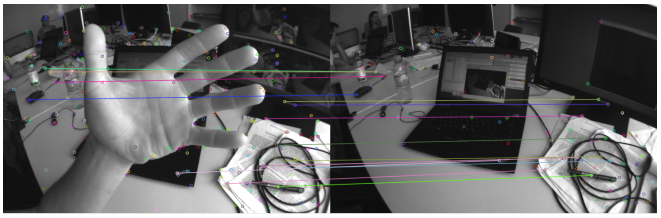

Fig. 3: Accepted matches using SIFT

IV. Using ferns

As said previously with the three-point algorithm it is possible to recover the 6 DoF of the camera pose from the relation from pixel coordinates and points in space. In this case machine learning techniques are used to model this relationship. In the classifier scheme, an object in space is a class and multiple views seen from the camera should all be classified as this class.

A *fern* [12] is a descriptor made from a set of binary tests such as in equation 6. When used as a classifier, every possible evaluation of a *fern* will contain a posterior probability distribution for every class.

$$f_j = \begin{cases} 1, & \text{if } I(d_j, 1) < I(d_j, 2) \\ 0, & \text{otherwise} \end{cases} \tag{6}$$

One *fern* is usually not descriptive enough to correctly classify. In [12] it is claimed that with 50 *ferns* and $S = 11$ a problem with 200 different classes is tractable. Regarding storage requirements, it would involve $50 \times 2^{11} \times 200 = 20480000$ elements to be stored in memory. If these are stored as *float* then 78 MB are needed, which is tractable.

V. Results

The descrived methods have been tested in a desktop scene covering an area of 7x3 meters. The path taken to acquire training and testing data covering the mentioned area can be seen in 4. The dataset contains 84 frames for training and 69 for testing.

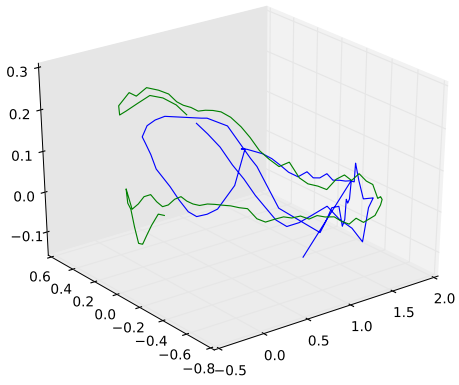

Fig. 4: In green, the path used for training. In blue, the path used for testing

A. Multi Relocalizer

1) Extended Second-order Minimization Real Pose Finder: The performance of ESM Real Pose Finder is not great in this dataset as seen in 6 although it was able to correctly relocalize in other datasets.

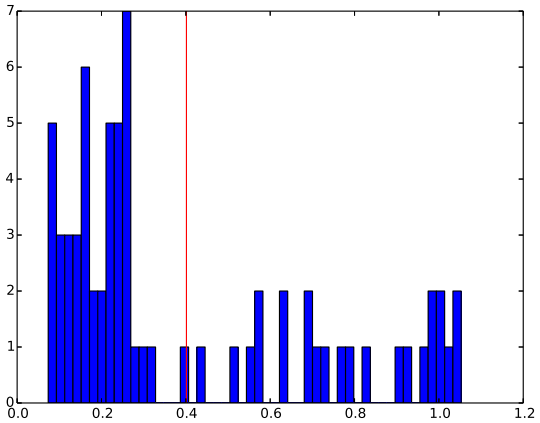

Fig. 5: Translation error histogram using CC *Place Finder* and ESM *Real Pose Finder*

2) Three-point Real Pose Finder: On the other side, the Three-point *Real Pose Finder* works very well as seen in 6. It can correctly retrieving the pose of 42 of the 69 frames. This method is very dependent on the results of the *Place Finder*, and on this dataset, the cross correlation method was not very effective. Better results on it would help iprove the results of this method and the ESM method.

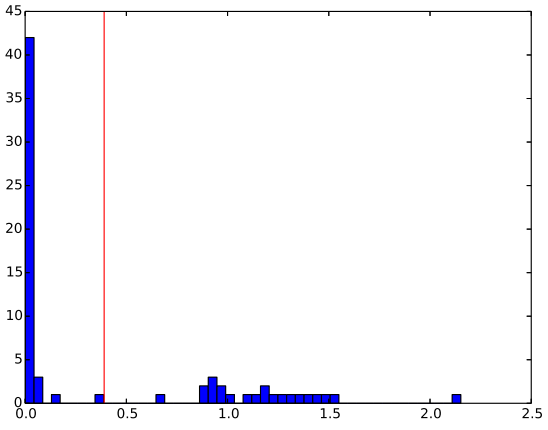

Fig. 6: Translation error histogram using CC and P3P, with the mean marked as a red line

B. Ferns

Finally, as an alternative to the PTAM relocalization method, it was proposed to use machine learning techniques,

more concretely *ferns*. In [12] it is shown that *ferns* can be used to distinguish up to around 200 different classes. We applied the method to this dataset where there are 1730 classes. In figure 7 it can be seen that almost 50 of the 69 frames where correctly relocalized. Which is more than using Multi Relocalizer with the three-point *Real Pose Finder*. The classifier was trained with 100 *ferns* of 12 tests each.

It should be noticed that not all points need to be well classified, as long as more than 3 points are correctly identified then the posterior RANSAC will find and use the once that agree.

The three-point with CC method and this one, can correctly relocalize the same number of frames. Probably there is one part of the dataset that is more ambiguous and difficult to recognize and both algorithms struggle with it.

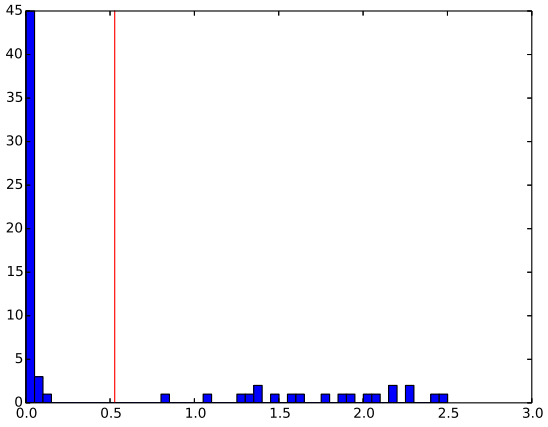

Fig. 7: Translation error histogram using *ferns* with 12 tests, with the mean marked as a red line

VI. Execution time

In figure 8 the mean execution time of relocalization is shown. There, it can be seen that the ESM and the P3P methods are the faster while the method using *ferns* classifier is slower and is sensitive to the number of classes. Also, while ESM and P3P use optimized third party libraries, the *ferns* based classifier has been integrally implemented by us, maybe not achieving the best performance. The used training time for the classifier can be seen in figure 9.

VII. Conclusions

This work has addressed an important part was missing in SVO, a good relocalization method which should recover the 6 DoF pose from only a map and a new frame. Different methods have been studied and implemented. First, as a starting point, the method from PTAM has been implemented which is based on image alignment.

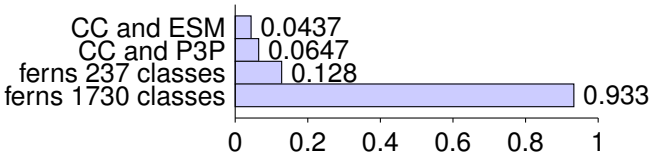

Fig. 8: Single relocalization execution time

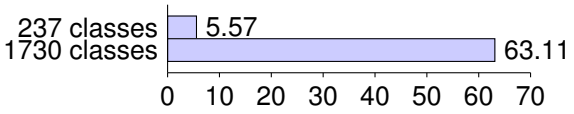

Fig. 9: *ferns* classifier training time

Then, an alternative method based on the previous has been proposed. This method is based on space geometry and on the descriptor extraction and matching framework and uses the three-point algorithm to find the camera pose by relating some world points and pixel coordinates. This method produces very accurate results when matches are found between images and in general is a great improvement over the base line, being more robust and accurate.

Finally a new approach is proposed. The central idea is to use machine learning techniques to characterise the appearance of some known points in space, to later be able to retrieve their position from the pixels of an image. *Ferns* are very similar to Random Forests but simpler and easier to implement, while being able to encode the same of information. A classifier based on *ferns* have been implemented and the same time as integrated in a relocalizer.

This method has been found to give good results even in larger areas where more than 1700 classes need to be classified even though a simplified version of the training was used.

References

[1] Simon Baker and Iain Matthews. Lucas-Kanade 20 Years On: A Unifying Framework. *International Journal of Computer Vision*, 56(3):221–255, February 2004.

[2] S Benhimane and E Malis. Homography-based 2D visual servoing. *Robotics and Automation, 2006. ICRA*, 2006.

[3] Martin A Fischler and Robert C Bolles. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM*, 24(6):381–395, 1981.

[4] Christian Forster, Matia Pizzoli, and Davide Scaramuzza. SVO: Fast Semi-Direct Monocular Visual Odometry. *Proc. IEEE Intl. Conf. on Robotics*, 2014.

[5] Richard Hartley and Andrew Zisserman. *Multiple view geometry in computer vision*. Cambridge university press, 2003.

[6] Georg Klein and David Murray. Parallel Tracking and Mapping for Small AR Workspaces. *IEEE and ACM International Symposium on Mixed and Augmented Reality*, November 2007.

[7] Georg Klein and David Murray. Improving the agility of keyframe-based SLAM. *Computer VisionECCV*, 2008.

[8] Laurent Kneip and Paul Furgale. Opengv: A unified and generalized approach to real-time calibrated geometric vision.

[9] Steven Lovegrove. *Parametric dense visual SLAM*. PhD thesis, Imperial College London, 2012.

[10] David G Lowe. Distinctive image features from scale-invariant keypoints. *International journal of computer vision*, 60(2):91–110, 2004.

[11] D Nister and H Stewenius. Scalable recognition with a vocabulary tree. *Vision and Pattern Recognition*, 2006.

[12] M Ozuysal and M Calonder. Fast keypoint recognition using random ferns. *Pattern Analysis*, 2010.

[13] Jamie Shotton, Ben Glocker, Christopher Zach, Shahram Izadi, Antonio Criminisi, and Andrew Fitzgibbon. Scene Coordinate Regression Forests for Camera Relocalization in RGB-D Images. In *Computer Vision and Pattern Recognition (CVPR), IEEE Conference on*, pages 2930–2937, 2013.

[14] Josef Sivic and Andrew Zisserman. Video Google: A text retrieval approach to object matching in videos. *Computer Vision, 2003. Proceedings.*, (Iccv):1470–1477 vol.2, 2003.

[15] Brian Williams, Georg Klein, and Ian Reid. Real-time SLAM relocalisation. *Vision, ICCV. IEEE 11th*, pages 1–8, 2007.