Available online at www.sciencedirect.com

ScienceDirect

Procedia Computer Science 39 (2014) 28 – 34

Procedia Computer Science

6th International conference on Intelligent Human Computer Interaction, IHCI 2014

On depth usage for a lightened visual *SLAM* in small environments

Maxime Boucher, Fakhreddine Ababsa, Malik Mallem

IBISC lab, 40 rue du Pelvoux, Courcouronnes 91020, France

Abstract

Historically popular, the well established monocular-SLAM is however subject to some limitations. The advent of cheap depth sensors allowed to circumvent some of these. Related methods frequently focus heavily on depth data. However these sensors have their own weaknesses. In some cases it is more appropriate to use both intensity and depth informations equally. We first conduct a few experiments in optimal conditions to determine how to use good quality information in our monocular based SLAM. From this we propose a lightweight *SLAM* designed for small constrained environments.

Peer-review under responsibility of the Scientific Committee of IHCI 2014
Keywords: SLAM, 3d Reconstruction, tracking

1. Introduction

SLAM is the ability for an autonomous mobile device to create a model of the unknown environment it moves in while localizing itself in the model. Applications range from robotics to augmented reality, including autonomous vehicles. Several kind of sensors are available to accomplish this task. When using a single camera, one speaks of monocular *SLAM*. In the calibrated case and for general moves, the camera pose has 6 degrees of freedom. In a point cloud representation of the environment, each point position have 3 degrees of freedom. Monocular *SLAM* is difficult since it requires estimating the camera pose and points positions using mesures lying a 2 dimensional space. Many approaches have been formulated to solve the task, and two main classes established. The first one is based on filtering, Kalman filters as in [4] or particular filters as in [13], and incrementally fuses measurements through the update of the camera pose and points positions probability distributions. The second one is based on the adaptation of *Structure From Motion* methods to the incremental nature of *SLAM* carrying bundle adjustments on a temporal, [14], or spatial, [7], subset of cameras. In [20] Strasdat *et al* compared filtered and *Structure From Motion* approaches and concluded the latter are generally more efficients.

Recently low cost structured light based depth sensors became available. These sensors provide informations lying in 3 dimensional space. Several approaches have been devised to take advantage of these, coupled or not to classical

* Corresponding author. Tel.: +33169477515 ; fax: +33169477599 .
E-mail address: maxime.boucher@ibisc.univ-evry.fr

Peer-review under responsibility of the Scientific Committee of IHCI 2014
doi:10.1016/j.procs.2014.11.006

RGB cameras. In[10] Henry *et al* realised a *SLAM* combining an *ICP* approach,[2], and a bundle adjustment approach in a single cost function. Interestingly, their experiments showed their hybrid cost function provides better results than any of the methods taken singly. In[5] Fioraio and Konolige realised an *ICP* based *SLAM*. They observed better results when taking visual intensity informations into account. In[15] Newcombe *et al* only use depth informations and fuse these in a dense environment map at each acquisition step. Their method provides the advantage to achieve loop closures naturally if the system's drift is moderate. However if drift is too large it isn't corrigible. In[19] Scherer *et al* fuse visual and depth informations in a hybrid cost function modyfing the monocular *SLAM* of[7].

2. Visual *SLAM*

2.1. Our approach

Notation: The term camera pose defines the orientation of the camera's axes and the position of its focal point. Using the term camera we usually refer to the camera pose at a given time and the associated image captured at this time. Using the term point we usually refer to a map point belonging to $\mathbb{R}^3$. The camera trajectory and the recorded video stream then define a set of cameras. We note C_i the i-th camera pose and I_i its associated image and P^j the j-th map's point's position. Then we note m_i^j the measure in the image plane of the j-th point observed by the ith camera and z_i^j is its measured depth.

We took inspiration from Mouragnon *et al*[14] to design our visual *SLAM*. The approach is based on the definition of keyframes, or keycameras. Keyframes are a subset of cameras ensuring that two consecutive ones have at least a certain amount of shared information. In a classic fashion the map is a cloud of points lying in $\mathbb{R}^3$. We suppose the camera calibrated, we thus know intrinsic parameters of the camera and the distorsion affecting image formation. We first describe the operating of the *SLAM* initialized, then we skim the initialization.

At each image acquisition we define a new camera. Its pose is initialized according to the last camera's one. Then we undistort the image and apply an interest points detection and description method. To reduce ambiguities we prefer the term measures over interest points. Measures are matched against those of the last keyframe and filtered through a *RANSAC* procedure, from Fischer and Bolles[6], applied to the *5-points* algorithm from Nister *et al*,[16], for essential matrix estimation. The filtered matchings allow to link *inliers* measures from the new camera to map's points. Then the camera pose can be adjusted through bundle adjustment, see[3] or[21], with its parameters defined as the only optimizable ones through the process.

Shared information between the current camera and the last keyframe is evaluated as the number of inlier matchings. Shall this number fall below a treshold M or the number of observed map's points fall below a treshold M', with $M' < M$, the preceding camera is promoted to keyframe.

The new keyframe pose is precisely estimated, hence it is possible to seek to link some map's points to unmatched camera measures. Indeed, interest points' detectors and descriptors aren't perfectly accurate and some points observed in, or matched between, images I_{i-2} and I_i may not be detected, or matched, in image I_{i-1}. To do so, we browse the points observed in *keyframes* preceding the last one, within the limit of $l = 1000$ points, and reproject them in the new keyframe image plane. Descriptors of these points are then matched against those of unmatched measures. Matching results are accepted if the distance between the measure and the reprojected point is below $5.99 \cdot \sigma^2$ measure units, with σ being the standard error of measure detection. The treshold's value is taken from Hartley and Zisserman's[9], chapter 4. We observed in some cases this step is crucial for the algorithm success, especially when the camera field of view is narrow. In a broader sense, this provides a tighter constraint of the problem.

Following this a local bundle adjustment is performed. In this *bundle adjustment*, following[14], the last $n = 3$ keyframes see their pose parameters defined as optimizable while those of the $N = 7$ preceding keyframes are fixed, but contribute to error evaluation. The locality of bundle adjustment is necessary to guarantee reasonnable execution times. Fixed parameters are important as they permit to constrain the problem. Finally, *inliers* between the new keyframe and the preceding one are triangulated if no map point is linked to them.

Initialisation is realised as follow: the very first frame is the first keyframe, its orientation is the identity and position the origin, $\mathbf{0}$. Cameras are matched against it until the number of *inliers* falls below M'. Then the third keyframe is defined as the preceding camera. The second keyframe is chosen between the first and third. Inlier measures are triangulated and a bundle adjustment is performed with fixed parameters for the first keyframe pose.

2.2. The need for depth

This system provides good results as long as a map of good quality is available. As a matter of fact, triangulation needs to happen frequently so that the system can be better constrained, particularly to reduce noise measurement impact, and because measures can only be matched as long as the point of view they're seen under hasn't too much changed. However, the accuracy of triangulation grows with the distance between cameras' positions, thus an accurate triangulation requires to happen more occasionally. These two constraints are opposites, and the first one being the strongest, there are some cases where the second constraint needs to be relaxed. It is the case when the camera performs a purely rotational move or if the translation part is too small for its effect to be larger than interest points detection's noise. Then point's triangulation is very likely to be extremely inaccurate.

Pure rotational moves are detectables and it is possible to forbid keyframe creation or point triangulation to prevent deviant estimations. This at best allows to avoid some failure cases, hoping the number of matchings will stay sufficiently high to constrain the system or that the camera will come back to some already mapped parts of the environment, as in [17] from Pirchheim *et al*.

The best way to solve this problem is probably to make use of some depth informations. There are at least two ways to obtain depth. The first one is to use a second camera, the second one is to use a *Kinect*-like depth camera. The latter provides direct depth measurement but have a limited range, some areas in depth image may contain no information and some environments, particularly exteriors, subject to infra-red radiation can't be mapped. The former needs to perform an additional matching step, thus requires more computational power. In both cases, depth informations being obtained on the triangulation principle measures are not perfectly accurate. To limit the computational needs of our system we choose to use a depth camera *Xtion Pro Live* from *ASUS*.

3. Using depth

Unlike the majority of approaches using depth camera as the main sensor to perform *SLAM*, observing the limits in terms of use case for the depth sensor, we choose to keep the RGB camera at the heart of our solution. Depth informations are thus used to replace the triangulation step when available. However, triangulation is only an initialisation for points' positions, which are afterwards refined through bundle adjustment. We thus wondered what use should be made of depth measures through bundle adjustment. We distinguish several cases. Differences between these cases lie in the way points are treated though bundle adjustment and the dimensionality of the space in which error evaluation occurs.

In bundle adjustments camera pose parameters are treated the same way as they are in our monocular approach. But errors, also called residuals, can be evaluated in three different ways. We note σ_i and σ_d the standard deviations for image and depth measures. σ_i is considered to be about 1 pixel and we follow Scherer *et al* results to estimate σ_d.

- *2D*:
  Depth information is not used. Residuals are the usual 2*D* reprojection errors

$$f(\{C,P\}) = \sum_i \sum_j h\left((p(C_i,P^j) - m_j^i)\right)$$

  Where $p()$ is the projection function of P^j in C_i's image plane and $h()$ a robustified cost function. The role of it is to limit the influence of outlier measures. We use the *Huber* cost function. This function grows quadratically below a predefined threshold, and linearly above. As errors lie in a 2-dimensional space we set the threshold as $5.99 \cdot \sigma_i^2$, again from [9].
- *3D*:
  Residuals are evaluated in 3*D* space, and we consider $z_i^j \cdot m_i^j$ to be a 3*D* measure of the *j*th point position according to camera *i*.

$$f(\{C,P\}) = \sum_i \sum_j \frac{1}{{\sigma_i}^2 \cdot {z_i^j}^2} \cdot h\left((t(C_i,P^j) - z_i^j \cdot m_i^j)\right)$$

Function $t(.)$ replace the point P^j in the reference of camera C_i. We recall that for a random variable x with a gaussian distribution $\mathcal{N}(\mu,\sigma^2)$, the resulting random variable from multiplication with a scalar α has the gaussian distribution $\mathcal{N}(\alpha\cdot\mu,\alpha^2\cdot\sigma^2)$. Thus we set the *Huber* function threshold as $min\left(7.81\cdot\sigma_d^2\ ,\ 7.81\cdot(z_i^j\cdot\sigma_m)^2\right)$. As noted by Scherer *et al* in [19], the drawback of this cost function is that the standard deviations of measures z_i^j and m_i^j are not of the same order.

- *2+1D*:
  This is the solution of Scherer *et al* in [19]. Image and depth residuals are evaluated separately

$$f(\{C,P\}) = \sum_i\sum_j \frac{1}{\sigma_i^2}\cdot h\left((p(C_i,P^j)-m_j^i)\right) + \frac{1}{\sigma_d^2}\cdot h\left((t(C_i,P^j))_{[3]}-z_i^j\right)$$

  It is easy in this case to take the different standard deviations of measures into account. For depth errors, the *Huber* function threshold is set to $3.84\cdot\sigma_d^2$.

Points can be treated in two ways:

- *opt*: points can be optimized in bundle adjustment.
- *fix*: points are not optimizable, thus are anchored to the $3D$ position they have been initialized to.

These 3 residuals evaluation methods and 2 treatments define 6 strategies. We use the *ceres* library to optimize cost functions. We consider a pinhole camera of 640×480 resolution with the following intrinsic parameters:

$$K = \begin{pmatrix} 546.04 & 0 & 316.66 \\ 0 & 546.05 & 234.71 \\ 0 & 0 & 1 \end{pmatrix}$$

4. Evaluations in small constrained environments

First we evaluated the 6 strategies in a small constrained environments with close objects so that depth measures can be considered of good quality. We recorded 6 RGB-D streams, comprising between 3000 and 16000 images. We tried to perform canonical moves with the camera. These canonical moves are either pure translations along a single axis of the camera, X, Y, or Z, or near pure rotations around a single axis X, Y or Z. We name these sequences Seq_X, Seq_Y, Seq_Z and Seq_θ, Seq_ϕ, Seq_ψ. For each sequence we performed 3 round trips. For Seq_X the maximal amplitude of a round trip is about 100cm and scene's depth is about 50cm, for Seq_Y 60cm and 50cm, for Seq_Z 100cm and depth belongs to the $[50,150]$cm interval. For rotational sequences movements amplitudes vary between 0 and $\frac{\pi}{2}$ radians. Depths belong to the $[50,100]$cm interval for Seq_θ, $[50,150]$cm for Seq_ϕ, and $[40,50]$cm for Seq_ψ. Figure 1 displays the first experimental scene and the result of a *SLAM* application.

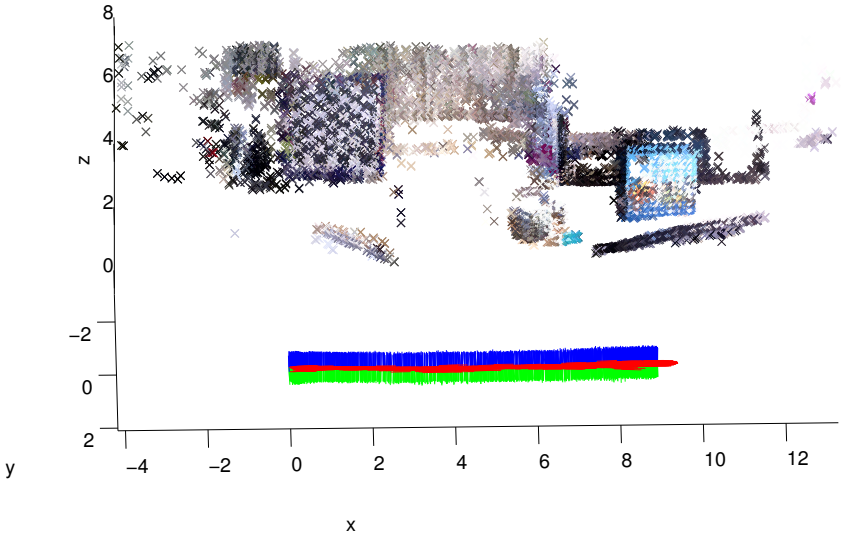
For each sequence the last camera pose is almost the same as the first. We applied the 6 strategies to the 6 sequences. We use the *OpenSURF* implementation of the *SURF*,[8], detector and descriptor. In table 1 we show position and orientation errors between the last and first camera pose. Each cell contains:

- The norm of the distance between cameras origins.
- The norm of the 3-vector made of the angular differences between axes. For readability these are expressed in degrees.

Unsurprisingly, initializations of points positions being of good quality, we observe all 3 cost functions perform well, the $2+1D$ appearing to be most accurate.

We find it much more interesting to observe that fixing points positions gives better estimations than not. Thus, it appears in such small constrained environments to be possible without lost of accuracy to lighten bundle adjustment costs. Indeed, when removing points from the set of parameters to optimize, the size of the matrix inversed during bundle adjustment becomes significantly smaller since there are usually much more points parameters than camera parameters. For all 6 sequences we measured a speed up of a factor between 3 and 4 during bundle adjustments.

(a)



(b)

Fig. 1: (a) three images from Seq_X. (b) reconstructions and frames' positions obtained with fixed points and the $3D$ cost function. Axes X, Y and Z are colored in red, green and blue.

5. Application: a relatively lightweight *SLAM*

From these observations we modified our general *SLAM*, refered to as ***Gen***, to adapt it to these kind of small constrained environments. Since in that case fixing map points is at least as efficient as optimizing them, performing local bundle adjustments doesn't provide much more improvements than using the most efficient cost function for error minimization during pose estimation. We can thus remove the local bundle adjustment step from the algorithm. We refer to this lightened version as ***Light***.

On the road to elaborate a more efficient *SLAM*, several other improvements can be made. As we earlier indicated, in the previous experiments we made use of *SURF* detector-descriptor. While it provides highly accurate point detection and good repeatability it is quite computationally expensive. On our machine *SURF* interest points detection and description takes ususally between 150ms and 250ms. Though more erroneous, lighter weight alternatives exist. For interest point detection we choose to use *Harris* corners and *Censure* points,[1], named *STAR* in the *OpenCV* library. For interest point description we observed binary descriptors *ORB*,[18], and *BRISK*,[12], provide similar accuracy, thus we can use one or the other. In table 2, we use the acronym *HCB* to refer to the combination of *Harris* corners detector, *Censure* points detector and *BRISK* descriptor.

		2D		3D		2+1D	
		opt	fix	opt	fix	opt	fix
Seq_X	cm	6.41	4.98	3.46	**2.06**	4.93	3.11
	°	10.18	8.82	7.57	**6.52**	9.24	7.46
Seq_Y	cm	1.31	0.51	0.76	0.39	0.61	**0.34**
	°	3.58	3.94	3.62	**3.51**	3.68	3.65
Seq_Z	cm	1.99	1.77	3.35	3.02	2.58	**1.59**
	°	1.17	1.08	1.80	1.58	1.48	**0.99**
Seq_θ	cm	0.38	**0.15**	1.09	0.89	0.53	0.47
	°	3.48	**3.14**	3.74	3.36	3.20	3.21
Seq_ϕ	cm	1.72	**0.87**	2.71	3.50	1.36	1.15
	°	2.98	3.62	2.66	**2.59**	3.75	3.48
Seq_ψ	cm	4.50	0.41	0.58	0.30	0.63	**0.18**
	°	4.43	1.31	0.68	**0.66**	1.17	0.84
Mean	cm	2.72	1.45	1.99	1.69	1.77	1.14
	°	4.3	3.65	3.35	3.04	3.75	3.27

Table 1: Pose errors for the last camera in position (cm) and orientation (degrees) for the six sequences for each strategy. For each one the best result is outlined in bold red.

While those descriptors can be fastly matched, they exhibit a significant quantity of *outliers*. Thus the drawback is the 5points-*RANSAC* based filtering may need a significant number of iterations to reach consensus, reducing the speed up gained during the matching step.

This can be circumvented using an inertial measurements unit, IMU. We use an *XSens* IMU which internally uses an extended Kalman filter to estimate its orientation. While orientations show drift accumulation over time, we observed it growing sufficiently slowly between two keyframes to provide good estimations of the camera relative rotation, provided the IMU's axes and the camera's axes are aligned. Alignement is easily achieved via a quick calibration of the hybrid sensor. Knowing the relative rotation of camera greatly simplifies the filtering step, since we only need to estimate the translation part of the essential matrix. We thus need to estimate only 2 parameters. The computational impact during *RANSAC* filtering of *outliers* matchings is then greatly reduced.The ***Light*** version applied with this scheme is refered as ***Light_imu***. The reader might be interested to know Kneip *et al* made the same observations in [11].

We recorded a sequence similar to the Seq_X one with inertial measures. This one is however about only one round trip, its whole length is thus about 2m. We applied the light versions of our *SLAM* on the sequence. In table 2 we compare the precedently devised RGB-D *SLAM* using the $2+1D$ cost function, with the lightweight version.

	Framerate	Errors	
		Position	Orientation
Gen SURF	3.2 fps	1.56cm	2.11°
Light SURF	3.3 fps	1.25cm	2.88°
Gen HCB	4.1 fps	2.78cm	1.71°
Light HCB	6.5 fps	1.51cm	4.17°
Light_imu HCB	8.9 fps	0.78cm	3.86°

Table 2: Comparison between our general RGB-D *SLAM* and some lightweight alternatives designed for small constrained environments.

Using *SURF* interest points we don't see important speed-up. The time needed to compute these being significantly more important than those of the other parts of the process.

However processing times are drastically reduced using the *HCB* points. We observe speed up of 1.58 and 2.17. We also observe the camera pose estimation doesn't suffer from these improvements.

6. Conclusion

We compared six ways of using depth informations in visual *SLAM*: we compared three cost functions and found one being more efficient than the others. We observed in some cases it may be more interesting not to optimize map points and from this devised some modifications to lighten our *SLAM*. Real results proved the efficiency of these modifications.

References

1. M. Agrawal, K. Konolige, and M. R. Blas. Censure: Center surround extremas for realtime feature detection and matching. In *Computer Vision–ECCV 2008*, pages 102–115. Springer, 2008.
2. P. J. Besl and N. D. McKay. Method for registration of 3-d shapes. In *Robotics-DL tentative*, pages 586–606. International Society for Optics and Photonics, 1992.
3. D. N. C. Engels, H. Stewenius. Bundle adjustment rules. In *PCV*, 2006.
4. A. J. Davison. Real-time simultaneous localisation and mapping with a single camera. In *ICCV*, pages 1403–1410. IEEE Computer Society, 2003.
5. N. Fioraio and K. Konolige. Realtime visual and point cloud slam. In *Proc. of the RGB-D Workshop on Advanced Reasoning with Depth Cameras at Robotics: Science and Systems Conf.(RSS)*, volume 27, 2011.
6. M. A. Fischler and R. C. Bolles. Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography. *Commun. ACM*, 24(6):381–395, 1981.
7. D. M. G. Klein. Parallel tracking and mapping for small a.r. workspaces. *International Symposium on Mixed and Augmented Reality*, 2007.
8. T. T. H. Bay, A. Ess and L. V. Gool. Speeded-up robust features (surf). In *ECCV*, 2006.
9. R. Hartley and A. Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, 2004.
10. P. Henry, M. Krainin, E. Herbst, X. Ren, and D. Fox. Rgb-d mapping: Using depth cameras for dense 3d modeling of indoor environments. In *the 12th International Symposium on Experimental Robotics (ISER)*, volume 20, pages 22–25, 2010.
11. L. Kneip, M. Chli, and R. Siegwart. Robust real-time visual odometry with a single camera and an imu. In J. Hoey, S. J. McKenna, and E. Trucco, editors, *BMVC*, pages 1–11. BMVA Press, 2011.
12. S. Leutenegger, M. Chli, and R. Y. Siegwart. Brisk: Binary robust invariant scalable keypoints. In *Computer Vision (ICCV), 2011 IEEE International Conference on*, pages 2548–2555. IEEE, 2011.
13. M. Montemerlo, S. Thrun, D. Koller, and B. Wegbreit. Fastslam: A factored solution to the simultaneous localization and mapping problem. *AAAI*, 2002.
14. E. Mouragnon, M. Lhuillier, M. Dhome, F. Dekeyser, and P. Sayd. Real time localization and 3d reconstruction. In *CVPR (1)*, pages 363–370. IEEE Computer Society, 2006.
15. R. A. Newcombe, S. Izadi, O. Hilliges, D. Molyneaux, D. Kim, A. J. Davison, P. Kohli, J. Shotton, S. Hodges, and A. W. Fitzgibbon. Kinectfusion: Real-time dense surface mapping and tracking. In *ISMAR*, pages 127–136, 2011.
16. D. Nister. An efficient solution to the five-point relative pose problem. *PAMI*, 2004.
17. C. Pirchheim, D. Schmalstieg, and G. Reitmayr. Handling pure camera rotation in keyframe-based slam. In *Mixed and Augmented Reality (ISMAR), 2013 IEEE International Symposium on*, pages 229–238. IEEE, 2013.
18. E. Rublee, V. Rabaud, K. Konolige, and G. Bradski. Orb: an efficient alternative to sift or surf. In *Computer Vision (ICCV), 2011 IEEE International Conference on*, pages 2564–2571. IEEE, 2011.
19. S. A. Scherer, D. Dube, and A. Zell. Using depth in visual simultaneous localisation and mapping. In *Robotics and Automation (ICRA), 2012 IEEE International Conference on*, pages 5216–5221. IEEE, 2012.
20. H. Strasdat, J. M. M. Montiel, and A. J. Davison. Real-time monocular slam: Why filter? In *ICRA*, pages 2657–2664. IEEE, 2010.
21. B. Triggs, P. F. McLauchlan, R. I. Hartley, and A. W. Fitzgibbon. Bundle adjustment - a modern synthesis. In B. Triggs, A. Zisserman, and R. Szeliski, editors, *Workshop on Vision Algorithms*, volume 1883 of *Lecture Notes in Computer Science*, pages 298–372. Springer, 1999.