Feature Localization Refinement for Improved Visual Odometry Accuracy

Aldo Cumani

Abstract— This work aims at improving the accuracy in the estimation of the path of a mobile platform from onboard passive stereo vision (so-called Visual Odometry). Our algorithm estimates motion steps by robust bundle adjustment of matched feature points, independently extracted from two pairs of stereo images. It is shown that, when using a fast Hessian-based feature detector/descriptor developed by us, a simple and computationally inexpensive algorithm can be devised to refine the image localization of features. Tests on real data confirm that this refinement actually yields a non negligible improvement in path estimation accuracy.

Keywords— Robot localization, Stereo vision, Visual odometry, Feature descriptors, Feature detectors

I. Introduction

THE research in sensor-based autonomous navigation has received considerable attention in recent years, particularly in connection with unmanned planetary exploration tasks. A rover vehicle on a planet surface (Mars, Moon) must be capable of truly autonomous navigation in a mostly unknown environment, as its continuous teleoperation from an Earth station is clearly out of question. In this context, accurate localization of the rover, i.e. accurate determination of its path, is generally a highly desirable feature. Pure dead reckoning (wheel odometry) is known to yield quite poor estimates (due e.g. to wheel slippage); also, wheel odometry alone can at most yield a 2D path estimate, so it is not even sufficient in principle when navigating on an unknown non-planar surface, and must be complemented by other independent inputs.

On the other hand, path estimation from onboard vision sensors (so-called *Visual Odometry* [1], [2], [3]) is able to yield accurate results while being intrinsically 3D. However, the following two points must be carefully considered.

First, any path estimate from onboard visual measurements is necessarily *incremental*, i.e. resulting from the sum of smaller independent motion estimates. Indeed, one-step visual estimation of the motion of the rover from a point A to a point B requires that a sufficient number of identifiable *landmarks* be visible from both A and B; moreover, such landmarks cannot be too far away if one wants a reliable estimate of the displacement of the rover from A to B (though distant landmarks may be useful for estimating the heading change). However, incremental algorithm typically implies error accumulation: it is therefore of utmost importance that each individual step be as accurate as possible, to keep error growing at a minimum.

This work has been partly supported by Regione Piemonte in the framework of the STEPS Project, led by Thales Alenia Space Italia

A. Cumani is with the Istituto Nazionale di Ricerca Metrologica, Torino, Italy a.cumani@inrim.it

Second, 3D estimation from monocular vision is subject to an unavoidable scale uncertainty, which may only be overcome using independent information, such as e.g. the actual path length measured independently, or the actual size of some recognizable object in the observed scene. On the other hand, a calibrated stereo rig is able to yield metric 3D structure and motion without needing other information.

In this context, our group at INRIM has developed a visual odometry algorithm [3], [4], [5], [6], [7] which relies on tracking pointwise image features extracted from the images of an onboard binocular stereo head. At intervals along the rover trajectory, its motion is estimated from the tracked features in the four images (two before and two after the motion). Several kinds of point features have been tested to this end. A *déjà vu* mechanism for exploiting possible loop-closure (cyclic paths) has also been devised, by periodically storing features and pose estimates, and comparing currently observed features to stored ones when the rover presumes to be near a saved position.

This work focusses on a particular feature detector/descriptor method (CVL) developed by us [6]. In particular, it is shown that this kind of descriptor can be easily exploited to refine the feature positions computed at the detection step, yielding an overall more accurate estimate of the rover motion. Sec. II summarizes the algorithm and the main sources of inaccuracy, while introducing our CVL feature detector/descriptor and a possible point position refinement method. Sec. III reports and discusses the performance of the proposed algorithm on some real data sets.

II. Visual Odometry

A. Algorithm

Our visual odometry algorithm relies on accumulating relative motions, estimated from corresponding features in the images acquired while the rover is moving. Such estimates are computed at *key frames*, whose spacing is a compromise between larger intervals, desirable both for numerical accuracy and for reducing computations, and the need for a sufficiently large number of features, which naturally tend to be lost because going out of view. The algorithm can be so summarized:

Feature extraction and matching. Point features are extracted at each key frame and left-right matched. Corresponding features are then also matched between consecutive key frames.

Motion estimation. The relative motion of the rover between the current key frame and the preceding one is estimated by robust bundle adjustment of matched feature points.

Our algorithm actually also provides a mechanism (*déjà vu correction*) for exploiting possible loop-closure to correct accumulated errors, by periodically saving features and comparing observed features to the stored ones, but this has already been described elsewhere [8]. The motion estimation method is also described in more detail in [7], and is only summarized here: the relative roto-translation $(\mathbf{r}, \mathbf{t})$ of the rover between two keyframes is found by minimizing a suitable function of the image-plane backprojection errors of matched points in the four images (L_1, R_1, L_2, R_2), i.e. the stereo pair (L_1, R_1) at the first keyframe and the one (L_2, R_2) at the second keyframe. These point matches can be graphically represented as

$$\begin{tikzcd} L_2 \arrow[r, Rightarrow] & R_2 \\ L_1 \arrow[u, Rightarrow] \arrow[r, Rightarrow] & R_1 \end{tikzcd} \qquad (1)$$

where $A \Longrightarrow B$ means that for any matched point in A we have a link to the corresponding point in B. Note that in the actual implementation we only keep bidirectionally valid matches, but the schema (1) makes explicit that we are taking L_1 as reference, and shall be useful later on.

So, assuming that a number N_{12} of point features have been extracted and matched over the four images, let $\mathbf{X}_i = [x_i, y_i, 1, t_i]^\top$ be the unknown 3D projective coordinates of feature $\mathcal{F}_i$, $i = 1 \ldots N_{12}$, in the left camera reference at keyframe 1. Let $\mathbf{u}_{iq} = [u_{iq}, v_{iq}]^\top$ be the 2D coordinates of the projection of $\mathbf{X}_i$ onto image $q \in \{L_1, R_1, L_2, R_2\}$, and $\mathbf{x}_{iq} = [su_{iq}, sv_{iq}, s]^\top$ the same in projective form. Then

$$\begin{array}{rclrcl} \mathbf{x}_{i,L_1} & = & \mathsf{P}_L \mathbf{X}_i & \mathbf{x}_{i,L_2} & = & \mathsf{P}_L \mathsf{M}_{12} \mathbf{X}_i \\ \mathbf{x}_{i,R_1} & = & \mathsf{P}_R \mathsf{M}_S \mathbf{X}_i & \mathbf{x}_{i,R_2} & = & \mathsf{P}_R \mathsf{M}_S \mathsf{M}_{12} \mathbf{X}_i \end{array} \qquad (2)$$

where the intrinsic camera matrices P_L and P_R are known from calibration, as is the 4×4 stereo transform matrix M_S. The only unknown (apart from $\mathbf{X}_i$) is the 4×4 motion matrix M_{12}, which can be parametrized in terms of a 3-rotation and 3-translation represented by two 3-vectors $\mathbf{r}_{12}$ and $\mathbf{t}_{12}$:

$$\mathsf{M}_{12} = \left[\begin{array}{cc} \mathsf{R}_{12} & \mathbf{t}_{12} \\ \mathbf{0}^\top & 1 \end{array} \right] = \left[\begin{array}{cc} e^{[\mathbf{r}_{12}]_\times} & \mathbf{t_{12}} \\ \mathbf{0}^\top & 1 \end{array} \right] \qquad (3)$$

Now, let $\mathbf{u}^*_{iq}$ be the measurements of $\mathbf{u}_{iq}$, i.e. the actually observed image coordinates of feature $\mathcal{F}_i$. We can define a global reprojection error measure J as

$$\begin{array}{rcl} J(\mathbf{p}) & = & \sum_i \sum_q f(\|\mathbf{e}_{iq}(\mathbf{p})\|^2) \\ & = & \sum_i \sum_q f(\|\mathbf{u}_{iq}(\mathbf{p}) - \mathbf{u}^*_{iq}\|^2) \end{array} \qquad (4)$$

parametrized as a function of the $6 + 3N_{12}$ unknowns

$$\mathbf{p} = [\mathbf{r}_{12}^\top, \mathbf{t}_{12}^\top, x_1, y_1, t_1, ...x_{N_{12}}, y_{N_{12}}, t_{N_{12}}]^\top \qquad (5)$$

Then, the first six entries of $\hat{\mathbf{p}} = \arg\min J$ yield the visual odometry estimate $(\hat{\mathbf{r}}_{12}, \hat{\mathbf{t}}_{12})$ of the inter-keyframe motion. This problem can be efficiently solved by a standard sparse Levenberg-Marquardt algorithm (see e.g. [9, A4.3]).

For robustness, a Lorentzian cost function $f(e^2) = \log(1 + e^2/\sigma^2)$ is used, with σ chosen as a function of the expected image-plane error. The residual errors are then compared against a threshold θ (proportional to the Lorentz σ), and those exceeding that threshold are marked as outliers. Bundle adjustment is then run again on the inliers only, using the standard sum-of-squares error function.

B. Possible sources of error

This algorithm is obviously subject to the accumulation of the errors made in each individual step. Such errors essentially come from three sources:

Matching errors: feature points in distinct images can be wrongly matched. This problem can be alleviated by a *robust* estimation method as the one sketched above, able to detect and reject wrong matches (outliers). It is anyway desirable to start with as many good matches as possible.

Insufficient features: detected features must both be numerous and rich in 3D structure, i.e. the corresponding 3D points must neither lie on some singular configuration, nor be too spatially concentrated. The distribution of visual features in the scene is obviously not under control, yet the choice of feature detector may heavily affect the result.

Localization errors: even neglecting mismatches, feature points must be *repeatable*, i.e. they must not disappear in different views, and *robust* against viewpoint changes, i.e. the image plane locations of matched features must be the projections of the same 3D point.

It is clear from the above discussion that a key issue for getting reliable and accurate results is the way image features are detected and represented. This is the subject of the next Section.

C. Feature detectors/descriptors

In general, (point) feature extraction consists of two steps:

Detection, aiming at the localization of visually salient points in the image, and at the determination of the apparent size (scale) of the visual feature.

Description, aiming at representing in a compact form the image behaviour in the vicinity of the detected point. This representation typically consists of a fixed number N of values, that can be interpreted as points in a Euclidean space E^N. This way, the (dis-)similarity of features from different images can be easily computed as the Euclidean distance of their representations.

Good references on state-of-the-art detectors are in [10]. Usually, detectors act by searching the image for the extrema of some local function of the smoothed luminance, such as the Harris operator or the Hessian (though other criteria may be used, e.g. edge line intersections). The size (scale) of the feature is then found at the scale-space extrema of some other operator (e.g. the Laplacian), possibly followed by an iterative refinement aiming at finding the affine shape of the feature, as in [11].

There is as well a vast literature on the subject of feature descriptors; again, a good reference is [12].

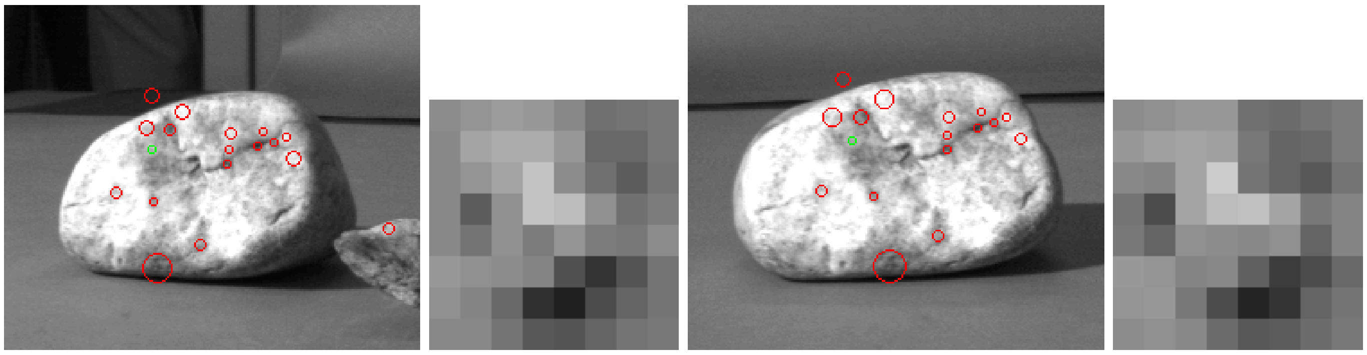

Fig. 1. Two frames from a laboratory sequence, with extracted CVL features. The descriptors for a pair of matched points (indicated by the green dots) are represented as 8×8 patches of graylevel "pixels".

A previous work [13] has discussed the effect on accuracy of various state-of-the-art detectors and descriptors, among which we have as well tested **SURF** features [14] and our homebrew **CVL** features [6]. Speeded-Up Robust Features use the Hessian for detecting interest points both in image space and in scale space. A rotation-invariant (but not affine-invariant) descriptor of size 64 or 128 is then computed by wavelet analysis of the luminance in an image patch, around the detected point. There is also a non-rotationally-invariant version (U-SURF) which has the advantage of faster computation.

D. The CVL feature detector/descriptor

Our own feature detector/descriptor [6] can be seen as a simplified form of SURF: detection is accomplished, as in the latter, by searching for image (x, y) and scale (σ) extrema of the normalized Hessian of image intensity:

$$H(\sigma) = \sigma^2 (I_{xx}(\sigma) I_{yy}(\sigma) - D(\sigma) I_{xy}^2(\sigma)) \tag{6}$$

where $I_{xx}(\sigma)$, $I_{yy}(\sigma)$ and $I_{xy}(\sigma)$ are the second derivatives of the image intensity I filtered with a Gaussian of scale σ, while the factor $D(\sigma)$ takes into account the fact that, for reasons of efficiency, the required derivatives are approximated by using suitably sized *box filters* [14], [6]. Indeed, using the integral image approach by Viola et al. [15], box filters can be implemented quite efficiently with a minimum number of operations, irrespective of filter size; this makes computationally attractive searching the scale space (σ) over e.g. a 4-octave range $\sigma = 1.2 \ldots 26$, with filter mask sizes ranging from 9×9 to 195×195.

A descriptor of size 64 is then computed by mapping a square image area of size 10σ centered at (x, y) to size 8×8, then normalizing these pixel values to zero-mean and unit variance. Again, the use of integral image and box filters allows fast computation of the required pixel averages.

The descriptor so obtained is neither rotation- nor affine-invariant, but it is quite fast to compute (four times faster than SURF) and has proven rather good for the visual odometry application [13]. Fig. 1 shows in pictorial form the descriptors of two matched points in a pair of images from a laboratory sequence, both for the sake of illustration, and because that pictorial description shall be useful in the next Section.

E. Reducing localization errors

A problem with the approach sketched in the previous paragraphs is apparent: image features are localised *independently* in each image by the detection algorithm. This means that if we have, say, two correctly matched points A and B in two images, there is no guarantee that their detected image-plane coordinates (x_A, y_A) and (x_B, y_B) are the projections of *exactly* the same 3D point on the surface of some real object, though, at least in our application, the difference is expected to be small, as the images subjected to the matching procedure are always taken from not too different points of view.

Small does not necessarily mean negligible, however, so one can imagine that better results could be achieved by correcting in some way the estimated image position of, say, feature B before feeding it to the bundle adjustment process. An obvious choice would be to correlate the luminance values in some suitably sized image patches centered around A and B respectively; this however would require both a non negligible amount of computations, and that the original images be still available at this point, not to mention the possible difference in scale of the two features.

On the other hand, a look at Fig. 1 suggests that, when using our CVL features, such a correlation search could be done directly on the descriptor values themselves. Indeed, as explained before, CVL descriptor values are just scaled and normalized intensity values around the feature point, and assuming a correct match also the possible difference in size has already been taken into account by scaling.

Formally, if $D_k^{(P)}$, $k = 0..63$ are the CVL descriptor values for feature point P, given two matched points A and B we define

$$C(x, y) = \sum_{i,j} D_{8(i+y)+j+x}^{(B)} D_{8i+j}^{(A)} \tag{7}$$

for integer x and y, and look for the maximum of C at

$$(dx, dy) = \arg\max C(x, y) \tag{8}$$

Note that C is defined only for integer values of the displacement (x, y), yet we can compute a fractional (dx, dy)

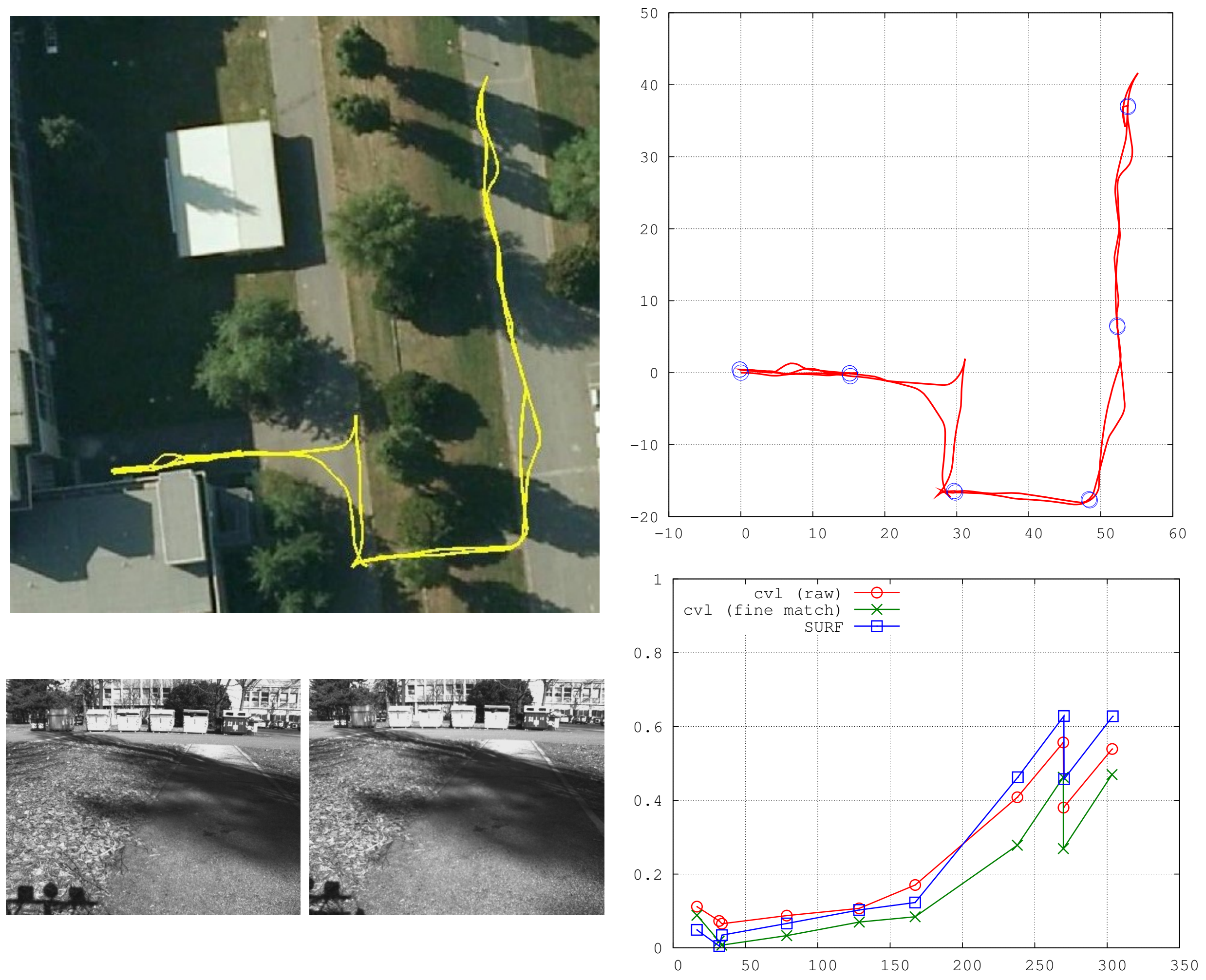

Fig. 2. Results for INRIM data set I. Top left: aerial view, bottom left: a typical pair of images, top right: estimated path, bottom right: return position error vs. path length for SURF features, raw CVL features and CVL with position refinement.

by fitting a second degree function to the 9 values of C on and around the maximum as given by Eq. (8). Moreover, since we assume that our features are correctly matched, we may expect the correction (dx, dy) to be small (less than one descriptor "pixel"), hence $C(x, y)$ only needs to be computed for the 9 values $(x, y) \, \epsilon \, [-1, 0, 1]$, so keeping the additional computations at a minimum.

The resulting (dx, dy) is expressed in descriptor units, so it must be scaled to real image pixels by multiplication by $10\sigma_B/8$, σ_B being the scale of feature point B (remember that, in our implementation, the 8×8 descriptors map an image area of size $10\sigma \times 10\sigma$). The scaled correction is then added to (x_B, y_B).

With reference to the schema (1), we take the points in L_1 as reference, and compute corrections for the matched points in R_1 with respect to L_1, in L_2 with respect to L_1, and in R_2 first with respect to L_2 and then with respect to L_1. The refined point coordinates are then used as measurements $\mathbf{u}^*_{iq}$ in the motion estimation algorithm of Sec. II-A.

III. Results

The visual odometry algorithm has been implemented as an application which can run either in batch mode, reading a previously acquired image sequence, as well as in real time (when using our CVL features), on a Linux laptop onboard a small commercial rover (ActivMedia Pioneer 2AT). Extensive tests have been performed both on synthetic image sequences [13], [16], as well on real sequences, either acquired using our rover, or taken from publicly available data sets.

A. *INRIM data sets*

These data sets were collected in the campus of our Institute, using our Pioneer rover equipped with a self-built stereo head, consisting of a pair of synchronized Basler A312f digital cameras with 6mm lenses, each providing a stream of CCIR-size (720×576) 8-bit graylevel images on a IEEE1394 bus.

Figs. 2, 3 and 4 show the results of running our algorithm, both with and without the point refinement method explained above. For the sake of comparison, also the results with SURF

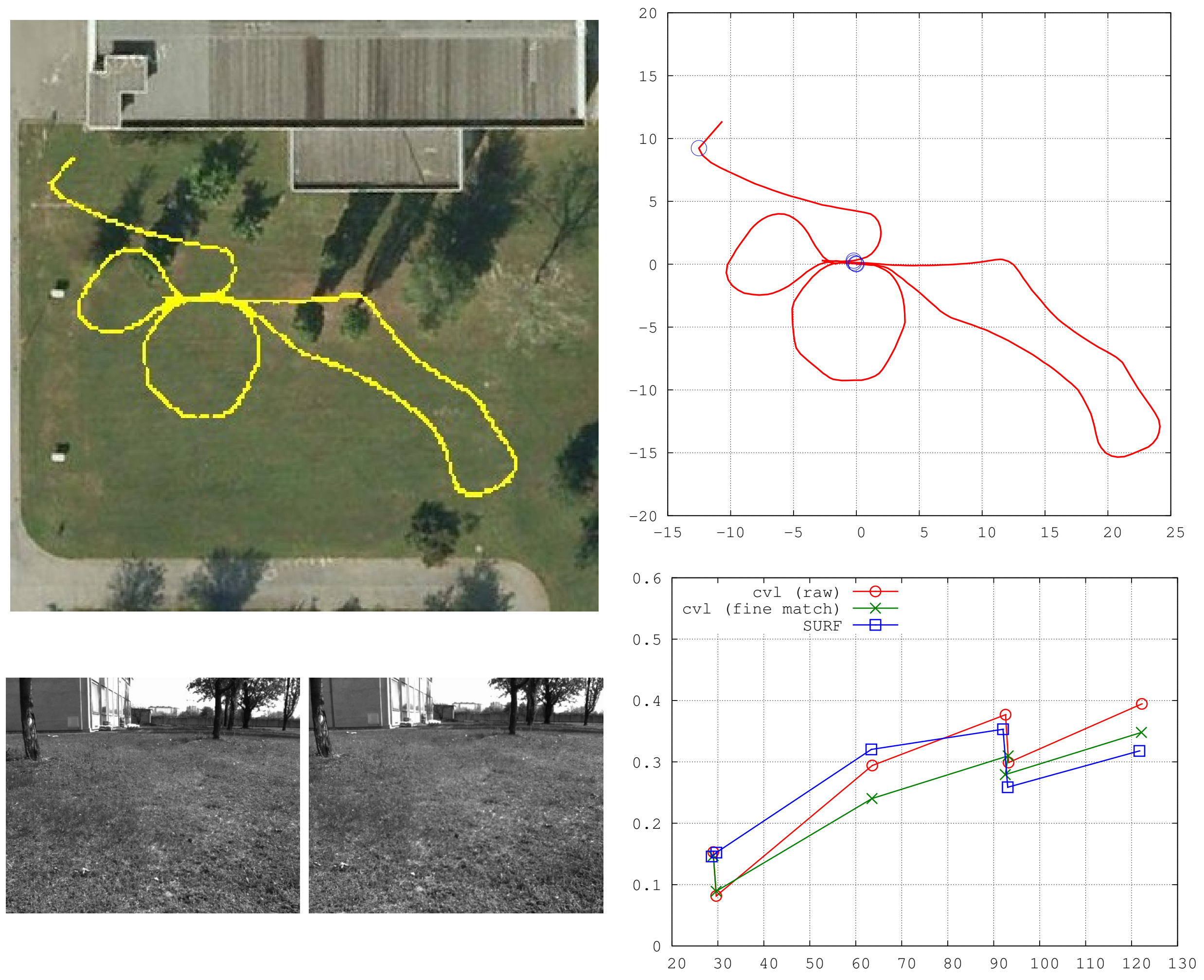

Fig. 3. Results for INRIM data set II. Top left: aerial view, bottom left: a typical pair of images, top right: estimated path, bottom right: return position error vs. path length for SURF features, raw CVL features and CVL with position refinement.

features are shown. These sequences cover a path length of about 350 m each (except set II, which was prematurely stopped due to battery problems.)

Note that we have currently no way to assess the accuracy of the path estimate by independent measurements. In previous experiments [5] we used an arrangement of visual targets placed on the ground at known positions, and took pairs of shots of the rover, at selected epochs, with a pair of (calibrated) compact cameras, so as to be able to evaluate the rover position with respect to the fixed targets by means of triangulation. This arrangement, however, is clearly impractical for rover paths extending over an area of any useful size (i.e. larger than about 10 m diameter). For this reason, in the tests reported here the rover was manually driven to pass multiple times through some places marked on the ground (blue circles in the figures), so the accumulated position error between successive transits could be evaluated (to within around 0.1 m) as a function of path length.

Fig. 2 shows some improved position accuracy when using the proposed refinement method. It should be remarked, however, that this is somewhat a particular case, as on this trajectory the rover was driven to return on its steps by moving backwards. This means that the observed scene on return was the same as on the forward path, which also explains the rather low overall position error.

Figs. 3 and 4 depict a more realistic situation, with the rover wandering around. Again, in both cases the proposed feature refinement method yields some improvement in the odometry accuracy over the use of raw CVL features.

Note that the points in the error graphs have been joined by lines only for the sake of readability, nearby points on the plot are not necessarily near on the path, and the error between successive points is simply unknown.

B. Oxford data set

The proposed algorithm has also been tested on the Oxford New College data set [17], [18], kindly provided to the vision research community by the Oxford Mobile Robotics Group.

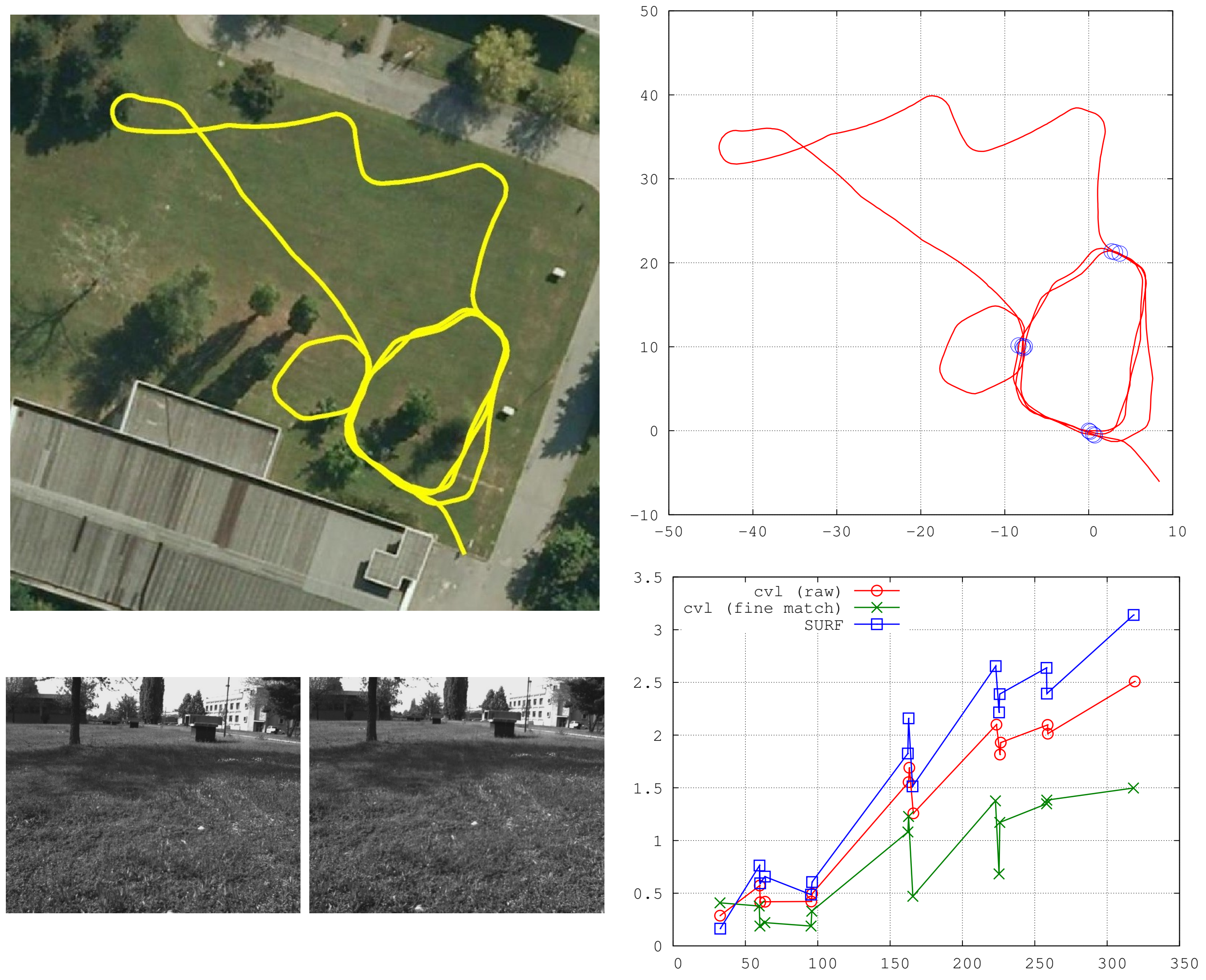

Fig. 4. Results for INRIM data set III. Top left: aerial view, bottom left: a typical pair of images, top right: estimated path, bottom right: return position error vs. path length for SURF features, raw CVL features and CVL with position refinement.

The platform used to collect those data is a two-wheeled vehicle built upon a Segway RMP200 base. A Point Grey Bumblebee camera, placed at about 1 m over ground and tilted about 13° towards ground, provides a 20 Hz stream of greylevel stereo pairs, with a resolution of 512×384 pixels.

This data set was recorded in November, 2008, in New College, Oxford (see Fig. 5) and comprises a total of 52478 stereo pairs, collected in about 47 minutes over a total path length of about 2844 m. The platform was also equipped with other sensors, among which a GPS unit aimed at providing absolute positioning data at more or less regular intervals; unfortunately, these GPS data are only partially useful as ground truth position data, since, as can be seen from Fig. 5 (bottom left, blue graph) their accuracy is at most around 5 m, with occasionally much larger errors.

Another problem with this data set is that the mobile platform, having only two wheels, is able to make very fast turns around its vertical axis. This actually happens two or three times in the part of the path marked as "Epoch B: Parkland" in Fig. 5 (top left). The effect of such sharp turns is to induce serious errors in the estimate of rotation, making the estimated path diverge sensibly from the actual one. Indeed, examining the image sequence, the images grabbed at those points are so blurred that would confuse *any* feature-based visual method.

In spite of that, it can be observed from Fig. 5 (right) that the proposed refinement method is able to yield results less diverging from reality than tose obtained with raw CVL features. The greater accuracy of the estimate is also evident from Fig. 5 (bottom left), where, for the sake of clarity, only the first about 1400 m of CVL estimated path are reported, with and without refinement, superimposed to the GPS track.

IV. Concluding Remarks

In this work we have discussed a visual odometry algorithm, which estimates motion steps by robust bundle adjustment of matched feature points, independently extracted from two pairs of stereo images. We have shown that, when using the fast

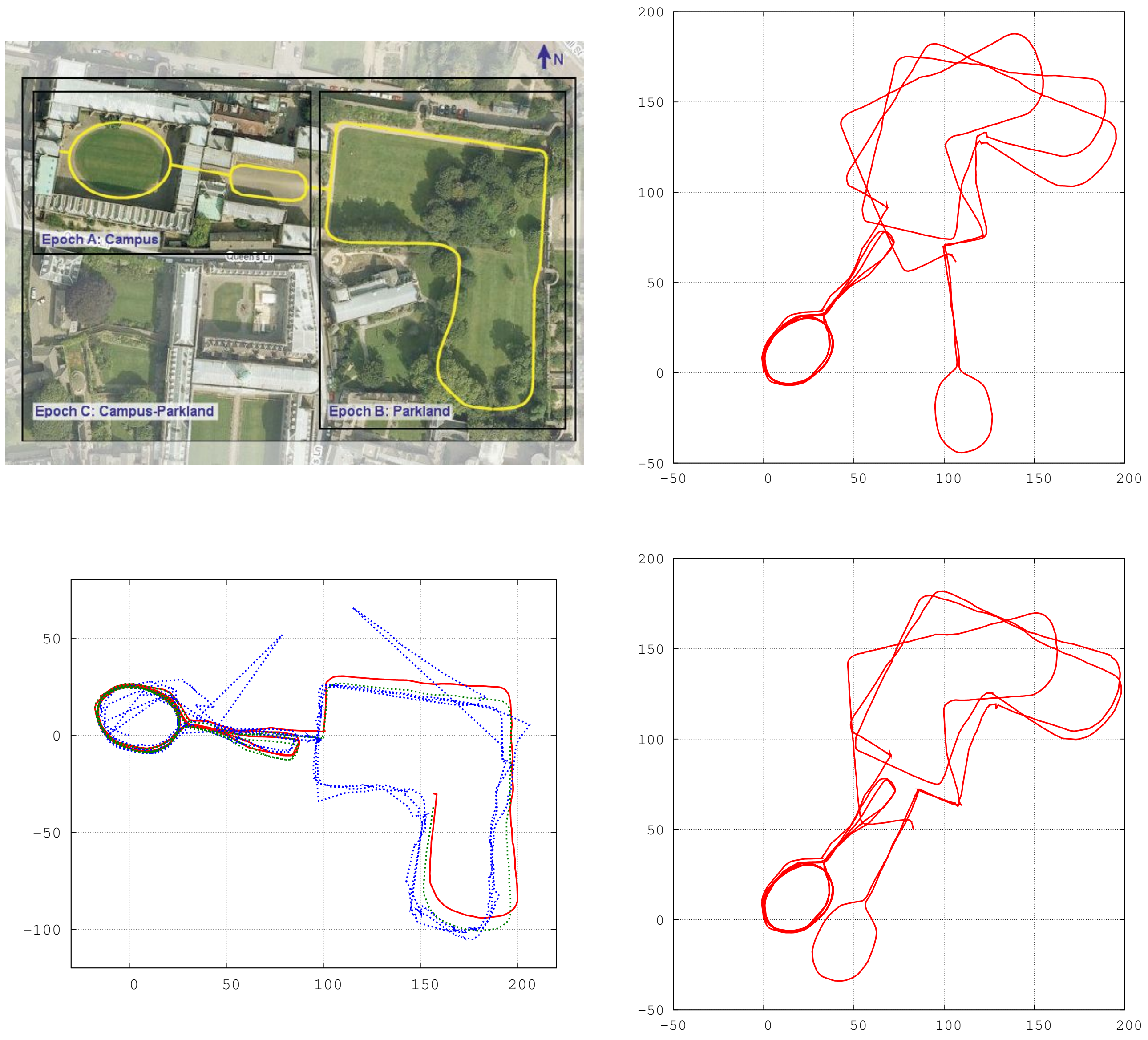

Fig. 5. Results for the New College data set. Top left: aerial view of location (from [17]). Right: estimated path using CVL features, without (top) and with the proposed feature point refinement. Bottom left: part of the CVL estimated path, without (red) and with (green) refinement, superimposed to GPS data.

Hessian-based feature detector/descriptor CVL developed by us, a simple and computationally inexpensive algorithm can be devised to refine the image localization of features, with the purpose of providing more accurate motion estimates. Tests on real data have also been presented.

Some conclusion can be drawn from the results reported in Sec. III. The most important one is that, in an application like our visual odometry algorithm, the simple CVL features we have developed perform rather well, in spite of not having a high degree of geometric invariance (rotation, affine etc.) Indeed, the point of view (PoV) changes little among the four images of a keyframe pair, and scale invariance is enough to take into account the scale change of near features with the rover motion.

Moreover, the point refinement method presented in this work is actually able to yield even more accurate path estimates, at the expense of a very small (order 10%) increase in computations. It should be remarked that CVL features are by themselves quite cheap to compute - about four times faster than SURF, which, as the acronym implies, are already among the fastest ones.

References

[1] D. Nister, O. Naroditsky, and J. Bergen, “Visual odometry,” in *Computer Vision and Pattern Recognition, 2004. CVPR 2004. Proceedings of the 2004 IEEE Computer Society Conference on*, vol. 1, 27 2004, pp. I–652 – I–659 Vol.1.

[2] Y. Cheng, M. Maimone, and L. Matthies, “Visual odometry on the Mars exploration rovers,” in *Proc. IEEE Int. Conf. Systems, Man and Cybernetics*, vol. 1, 2005, pp. 903–910.

[3] A. Cumani and A. Guiducci, “Stereo-based visual odometry for robust rover navigation,” *WSEAS Trans. Circuits and Systems*, vol. 5, no. 10, pp. 1556–1562, 2006.

[4] ——, “Visual odometry for robust rover navigation by binocular stereo,” in *Proc. 6th WSEAS Int. Conf. on Signal, Speech and Image Processing*, 2006.

[5] ——, “Accurate visual odometry for rover navigation,” in *Proc. RAAD08 17th Int. Workshop Robotics in Alpe-Adria-Danube Region*, 2008.

[6] ——, “Fast stereo-based visual odometry for rover navigation,” *WSEAS Trans. Circuits and Systems*, vol. 7, no. 7, pp. 648–657, 2008.

[7] ——, *Proceedings of the European Computing Conference*. BERLIN – DEU: Springer, 2009, vol. 2, ch. Robust and Accurate Visual Odometry by Stereo, pp. 485–494, series: Lecture Notes in Electrical Engineering , Vol. 28.

[8] ——, “Visual odometry with SURFs and déjàvu correction,” in *Proc. 8th Conf. Optical 3D Measurement Techniques*, 2007.

[9] R. Hartley and A. Zisserman, *Multiple View Geometry in Computer Vision*, 2nd ed. Cambridge Univ. Press, 2003.

[10] T. Tuytelaars and K. Mikolajczyk, “Local invariant feature detectors: a survey,” *Found. Trends. Comput. Graph. Vis.*, vol. 3, no. 3, pp. 177–280, 2008.

[11] K. Mikolajczyk and C. Schmid, “Scale & affine invariant interest point detectors,” *International Journal of Computer Vision*, vol. 60, no. 1, pp. 63–86, 2004.

[12] ——, “A performance evaluation of local descriptors,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 27, no. 10, pp. 1615–1630, 2005.

[13] A. Cumani and A. Guiducci, “Accurate determination of the path of an autonomous rover by onboard vision,” in *Proceedings of 5th Brazilian Congress of Metrology*, November 9-13, 2009.

[14] H. Bay, T. Tuytelaars, and L. Van Gool, “SURF: Speeded up robust features,” in *Proceedings 9th European Conference on Computer Vision*, 2006, see also http://www.vision.ee.ethz.ch/~surf/.

[15] P. Viola and M. Jones, “Rapid object detection using a boosted cascade of simple features,” in *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, 2001, pp. 511–518.

[16] A. Cumani and A. Guiducci, “Selecting feature detectors for accurate visual odometry,” *WSEAS Trans. Circuits and Systems*, vol. 8, pp. 822–831, 2009.

[17] M. Smith, I. Baldwin, W. Churchill, R. Paul, and P. Newman, “The new college vision and laser data set,” *International Journal for Robotics Research (IJRR)*, vol. 28, no. 5, pp. 595–599, May 2009.

[18] http://www.robots.ox.ac.uk/NewCollegeData/.

Aldo Cumani graduated in Electrical Engineering in 1972 at the Politecnico di Torino, Turin, Italy. Since then he has been with the Istituto Elettrotecnico Nazionale Galileo Ferraris in Turin, now Istituto Nazionale di Ricerca Metrologica, where he is currently Senior Researcher. His interests are mainly in the fields of image processing and computer vision, with applications to industrial automation and autonomous navigation.