Micro Air Vehicle Localization and Position Tracking from Textured 3D Cadastral Models

András L. Majdik, Damiano Verda, Yves Albers-Schoenberg, Davide Scaramuzza

***Abstract*— In this paper, we address the problem of localizing a camera-equipped Micro Aerial Vehicle (MAV) flying in urban streets at low altitudes. An appearance-based global positioning system to localize MAVs with respect to the surrounding buildings is introduced. We rely on an air-ground image matching algorithm to search the airborne image of the MAV within a ground-level Street View image database and to detect image matching points. Based on the image matching points, we infer the global position of the MAV by back-projecting the corresponding image points onto a cadastral 3D city model. Furthermore, we describe an algorithm to track the position of the flying vehicle over several frames and to correct the accumulated drift of the visual odometry, whenever a good match is detected between the airborne MAV and the street-level images. The proposed approach is tested on a dataset captured with a small quadroctopter flying in the streets of Zurich.**

SUPPLEMENTARY MATERIAL

Please note the dataset used in this work along with a video demonstration are both available on our webpage: `rpg.ifi.uzh.ch`

I. INTRODUCTION

Our motivation is to create vision-driven localization methods for Micro Aerial Vehicles (MAV) flying in urban environments, where the satellite GPS signal is often shadowed by the presence of the buildings, or not available. Accurate localization is indispensable to safely operate small-sized aerial service-robots to perform everyday tasks, e.g., goods delivery,[1] inspection and monitoring,[2] first-response and telepresence in case of accidents.

In this paper, we tackle the problem of localizing MAVs in urban streets, with respect to the surrounding buildings. We propose the use of textured 3D city models to solve the localization problem of a camera equipped MAV. A graphical illustration of the problem addressed in this work is shown in Fig. 1.

A. Majdik, D. Scaramuzza, and Y. Albers-Schoenberg are with the Robotics and Perception Group, University of Zurich, Switzerland—`http://rpg.ifi.uzh.ch`. A. Majdik is also affiliated with the University of Szeged, Hungary. D. Verda is with the University of Genoa, Italy. Contact: `majdik@ifi.uzh.ch`, `davide.scaramuzza@ieee.org`, `yvesal@ethz.ch`, `damiano.verda@unige.it`.

This research was supported by Scientic Exchange Programme SCIEX-NMS-CH (no. 12.097), the European Union and the European Social Fund through project FuturICT.hu (grant no.: TAMOP-4.2.2.C-11/1/KONV-2012-0013), the Swiss National Science Foundation through project number 200021-143607 (Swarm of Flying Cameras), and the National Centre of Competence in Research Robotics.

[1]As envisaged by Matternet: `http://matternet.us`.

[2]As envisaged by Skycatch: `http://skycatch.com`.

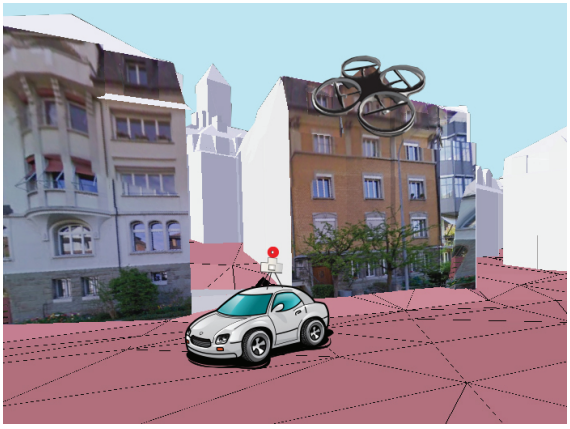

Fig. 1: Illustration of the problem addressed in this work. The absolute position of the aerial vehicle is computed by matching airborne MAV images with ground-level Street View images that have previously been back-projected onto the cadastral 3D city model.

In our previous work [1], we described an algorithm to search airborne MAV images within a geo-registered, street-level image database. Namely, we localized airborne images in a topological map, where each node of the map is represented by a Street View image. [3]

In this paper, we advance our earlier work [1], by back-projecting the geo-referenced images onto the 3D cadastral model of the city to obtain the depth the scene. Therefore, the algorithm described in this paper, computes the 3D position of the corresponding image points—between the airborne MAV images and ground-level Street View images—in order to compute the real-world 3D position of the flying vehicle within a metric map.

World models, maps of the environment, street-network layouts have been used for a long time to localize vehicles performing planar motion in urban environments. Recently, several research works have addressed the localization of ground vehicles using publicly available maps [2], [3], road networks [4] or satellite images [5]. However, the algorithms described in those works are not suitable for the localization of flying vehicles, because of the large viewpoint differences. With the advance of the mapping technology, more and more detailed, textured 3D city models are becoming publicly available [6], which can be exploited for vision-based localization of MAVs.

In the recent years, numerous research papers have addressed visual appearance-based place recognition by ground vehicles in Simultaneous Localization and Mapping (SLAM) systems [7], [8]. Such image-matching algorithms tend to

[3]In our work we used Google Street View images: `http://google.com/streetview`.

perform poorly in the case of flying robots [1], since in this case—besides the challenges present in ground visual search algorithms, such as illumination, lens distortions, over-season variation of the vegetation, and scene changes between the query and the database images—extreme changes in viewpoint and scale can be found between the aerial MAV images and the ground-level images. To illustrate the challenges of matching airborne images with ground level ones, in Fig. 2 we show a few samples of the airborne images and their associate Street View images from the dataset used in this work. As can be observed, due to the different fields of view of the ground cameras and aerial vehicles and their different distance to the buildings' facades, the aerial image is often a small subsection of the ground-level image, which mainly consists of highly-repetitive and self-similar structures (e.g., windows). All these peculiarities make the air-ground matching problem extremely difficult to solve for state-of-the-art feature-based image-search techniques.

In our work we depart from conventional appearance-based localization algorithms by applying our *air-ground image matching* algorithm to match airborne MAV images with ground-level Street View images. In this paper, we advance our previous topological localization [1] by computing and tracking the position of the MAV in the 3D space using cadastral 3D city models. To the best of our knowledge, this is the first work to present an in-depth analysis to solve the problem of localizing an MAV and track its position using textured 3D cadastral city models.

To summarize, this paper advances the state-of-the-art with the following contributions:

- We present a solution to the localization problem of MAVs flying at low altitudes (up to 10 meters) in urban streets. The dataset used in this work is publicly available on our website.
- We present a new appearance-based global positioning system to detect the position of MAVs with respect to the surrounding buildings. The proposed algorithm matches airborne MAV images with geo-tagged Street View images and exploits cadastral 3D city models in order to compute the absolute position of the flying vehicle.
- We describe an algorithm to track the vehicle position and correct the accumulated drift induced by the on-board state estimator.

The remainder of the paper is organized as follows. Section II presents the visual localization system. Section III describes the position tracking algorithm. Section IV presents the experimental results.

II. APPEARANCE-BASED GLOBAL POSITIONING SYSTEM

In this section, we briefly summarize and show the limitations of our previous work [1] concerning the appearance-based, global localization of MAVs in topological maps. Next, we present the cadastral 3D model used in this work and we show the back-projection of the Street View images onto the city model.

Fig. 2: Comparison between ground-level Street View (top row) and airborne MAV (row 2 and 3) images used in this work. Note the significant changes—in terms of viewpoint, over-season variation, and scene between the database ((a); respectively (d)) and query images ((b), (c); respectively (e),(f))—that obstruct their visual recognition.

A. Overview and evaluation of the Air-ground Image Matching algorithm

Conventional image-matching algorithms tend to fail in matching airborne MAV images with street-level images because of the extreme changes in viewpoint and scale that can be found between them. In [1], we proposed an algorithm that can successfully match air-ground images, with a significant difference in viewpoint and scale. To avoid the caveats of other image-search algorithms in case of severe viewpoint changes between the query and the database images, in [1] we generate virtual views of the scene, which exploit the air-ground geometry of the system. Next, similarly to [9], we extract SIFT [10] image points in the original image, and also, on the generated virtual views. Further on, we use a robust algorithm to select the inlier feature points based on virtual line descriptors [11]. Also, we present techniques to speed up the image search within the Street View image database by adopting a histogram-voting scheme and by computing the algorithm in parallel threads on different cores.

Our appearance-only-based *Air-ground matching* algorithm is described more in detail in [1]. It can successfully recognize more than 45% of the airborne MAV images within a database of street-level images with 100% precision—namely without detecting any false positive matches—using only the visual similarity between them. In other words, almost every second airborne image captured by the MAV, flying at different altitudes (up to 15 meters), and often traveling close to the buildings, is successfully recognized along a 2km path.[4]

The precision of the air-ground matching algorithm and

[4]For further details please watch: `http://youtu.be/CDdUKESUeLc`.

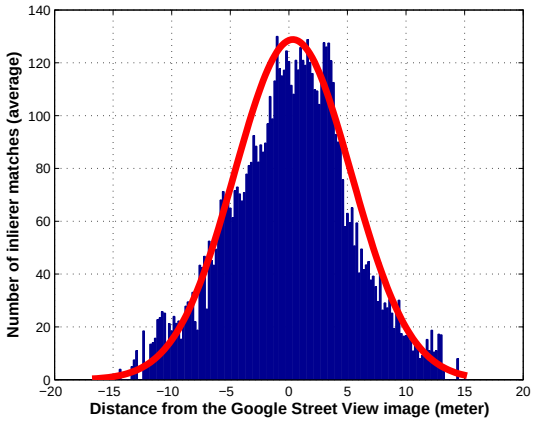

Fig. 3: Number of inlier feature points matched between the MAV and ground images versus the distance to the closest Street View image.

the uncertainty of the position determination depends on the correctly-matched number of features. Fig. 3 summarizes the mean number of inliers matched between the airborne and the ground images versus the distance to the closest Street View image. The results show a Gaussian distribution with standard deviation $\sigma = 5$ meters. This means that, if the MAV is in the vicinity of 5 meters from a Street View image along the path, our algorithm can detect around 60 correct correspondences. The 3D coordinates of the matched image feature points can be computed by back-projecting them onto the 3D city model.

The main goal of this work is to present a proof-of-concept of the system, rather than the real-time, efficient implementation. Though, for the sake of completeness, we present in Fig. 4 the effective processing time of the Air-ground image matching algorithm, using a commercially available laptop with an 8 core—2.40 GHz clock—architecture. The Air-ground matching algorithm is computed in five major steps: (1) virtual view generation and feature extraction; (2) approximate nearest-neighbor search within the full Street View database; (3) putative correspondences selection; (4) approximate nearest-neighbor search among the features extracted from the aerial MAV image with respect to the selected ground level image; (5) acceptance of good matches (kVLD inlier detection). In Fig. 4 we used more than 400 airborne MAV images. All the images were searched within the entire Street View images that could be found along the 2km trajectory. Notice that the longest computation time is the approximate nearest-neighbor search in the entire Street View database for the feature descriptors found in the MAV image. However, for position tracking, this step is completely neglected (Section III) since, in this case, the MAV image is compared only with the neighboring Street View images (usually up to 4 or 8, computed in parallel on different cores, depending on the road configuration). Finally, notice that the histogram voting (Fig. 4) takes only 0.01 seconds. On average, steps (1), (4), and (5) are computed in 3.2 seconds. Therefore, if the MAV flies roughly with a speed of 2 m/s, its position would be updated every 6.5 meters (section IV).

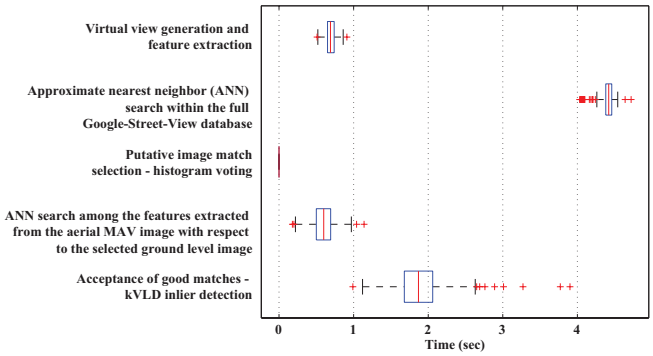

Fig. 4: Analysis of the processing time of the Air-ground image matching algorithm. To compute this figure, we used more than 400 airborne MAV images, and all the images were searched within the entire Street View image database, that could be found along the 2km trajectory.

B. Textured 3D cadastral models

The 3D cadastral model of Zurich used in this work was acquired from the city administration and claims to have an average lateral position error of $\sigma_{\mathrm{l}} = \pm 10$ cm and an average error in height of $\sigma_{\mathrm{h}} = \pm 50$ cm. The city model is referenced in the Swiss Coordinate System *CH1903* [12]. Note that this model does not contain any textures.

The geo-location information of the Google Street View dataset is not exact. The geo-tags of the Street View images provide only approximate information about where the images were recorded by the vehicle. Indeed, according to [13], where 1,400 Street View images were used to perform the analysis, the average error of the camera positions is 3.7 meters and the average error of the camera orientation is 1.9 degrees. In the same work, an algorithm was proposed to improve the precision of the Street View image poses. There, cadastral 3D city-models were used, in combination with image-segmentation techniques, to detect the outline of the buildings. Finally, the pose was computed by an iterative optimization, namely by minimizing the offset between the segmented outline and the reprojected one. The resulting corrected Street View image positions have a standard deviation of 0.1184 meters and the orientation of the cameras have standard deviation of 0.476 degrees.

In our work, we apply the algorithm from [13] on the dataset used in this work to correct the Street View image poses. Then, we back-project each pixel onto the 3D cadastral model. One sample of the resulting *textured 3D model* is shown in Fig. 5. This step is crucial to compute the scale of the monocular visual odometry (Section III-A) and to localize the MAV images with respect to the street level ones, thus, reducing the uncertainty of the position tracking algorithm. In the next section, we give more details about the integration of textured 3D models into our pipeline.

C. Global MAV camera pose estimation

By back-projecting the Street View images, onto the 3D city model as illustrated in Fig. 5, the absolute MAV camera pose and orientation can be estimated given a set of known 3D-2D correspondence points. Several approaches have been proposed in the literature to estimate the external camera parameters based on 3D-2D correspondences. In

Fig. 5: (a) perspective view of the cadastal 3D city model; (b) the ground-level Street View image overlaid on the model; (c) the back-projected texture onto the cadastral 3D city model; (d) estimated MAV camera positions matched with one Street View image; (e) the synthesized view from one estimated camera position corresponding to actual MAV image (f); (g)-(i) show another example from our dataset, where (g) is an aerial view of the estimated camera position (h), which is marked with the blue camera in front of the textured 3D model, (h) is the synthesized view from the estimated camera position corresponding to actual MAV image (i).

[14], the *perspective-n-point (PnP) problem* was introduced and different solutions were described to retrieve the absolute camera pose given *n correspondences*. The authors in [15] addressed the PnP problem for the minimal case where n equals 3 and introduced a novel parametrization to compute the absolute camera position and orientation. Given that the output of our Air-ground matching algorithm [1] may still contain outliers and that the model-generated 3D coordinates may depart from the real 3D coordinates, we apply the P3P algorithm together with a RANSAC scheme [14] to discard the outliers. Finally, using the inlier points we compute the MAV camera position by applying EPnP [16]. We refine the resulting camera pose estimate using the Levenberg-Marquardt [17] optimization, which minimizes the reprojection error given by the sum of the squared distances between the observed image points and the reprojected 3D points.

Fig. 5 (a-c) show how the Street View images are back-projected onto the 3D city model. Moreover, Fig 5 (d) shows the estimated camera positions and orientations in the 3D city model for a series of consecutive MAV images. As we do not have any ground-truth (i.e. we do not know the true MAV camera position) we visually evaluate the accurateness of the position estimate by rendering-out the estimated MAV camera view and comparing it to the actual MAV image for a given position as shown in 5(e-f). Fig. 5(g-i) again show another example of estimated camera position (g), the synthesized camera view (h), and the actual MAV image (i). By comparing the actual MAV images to the rendered-out views (Fig. 5(e-f) and Fig. 5(h-i)), we can see that—even though there are differences in terms of orientation and position—there is a significant visual overlap between the two images meaning that the estimated camera position is relatively close to the true camera position. Similar results were derived for the remaining MAV-Street View image pairs of the recorded dataset. Based on this visual verification, we can conclude that the absolute MAV camera positions are correctly computed. These global-localization updates will be used in the next section to correct the accumulated drift in the trajectory of the MAV.

III. POSITION TRACKING

The goal of this section is to track the state of the MAV over several images. The vehicle state at time k is composed by the position vector and the orientation of the airborne image with respect to the global reference system.

To simplify the proposed algorithm, we neglect the roll and pitch, since we assume that the MAV flies in near-hovering conditions. Consequently, we consider the reduced state vector $q_k \in \mathbb{R}^4$

$$q_k = (p_k, \theta_k), \tag{1}$$

where $p_k \in \mathbb{R}^3$ denotes the position and $\theta_k \in \mathbb{R}$ the yaw angle.

We adopt a Bayesian approach [18] to track and update the position of the MAV. We compute the posterior probability density function (PDF) of the state in two steps. To compute the prediction update of the Bayesian filter, we use visual odometry. To compute the measurement update, we integrate the global position as soon as this is made available by the algorithm described in the previous section.

The system model f describes the evolution of the state over time. The measurement model h relates the current measurement $z_k \in \mathbb{R}^4$ to the state. Both are expressed in a probabilistic form:

$$q_{k|k-1} = f(q_{k-1|k-1}, u_{k-1}), \tag{2}$$

$$z_k = h(q_{k|k-1}), \tag{3}$$

where $u_{k-1} \in \mathbb{R}^4$ denotes the output of the visual odometry algorithm at time $k-1$, $q_{k|k-1}$ denotes the prediction estimate of q at time k, and $q_{k-1|k-1}$ denotes the updated estimate of q at time $k-1$.

A. Visual odometry

Visual Odometry (VO) is the problem of incrementally estimating the ego-motion of a vehicle using its on-board camera(s) [19]. We use the VO algorithm in [20] to incrementally estimate the state of the MAV.

B. Uncertainty estimation and propagation of the VO

At time k, VO takes two consecutive images I_k, I_{k-1} as input and returns an incremental motion estimate with respect to the camera reference system. We define this estimate as $\delta^*_{k,k-1} \in \mathbb{R}^4$

$$\delta^*_{k,k-1} = (\Delta s^*_k, \Delta\theta_k), \tag{4}$$

where $\Delta s^*_k \in \mathbb{R}^3$ denotes the translational component of the motion and $\theta_k \in \mathbb{R}$ the yaw increment. Δs^*_k is valid up to a scale factor, thus the metric translation $\Delta s_k \in \mathbb{R}^3$ of the MAV at time k with respect to the camera reference frame is equal to

$$\Delta s_k = \lambda \Delta s^*_k. \tag{5}$$

We define $\delta_{k,k-1} \in \mathbb{R}^4$ as

$$\delta_{k,k-1} = (\Delta s_k, \Delta\theta_k), \tag{6}$$

where $\lambda \in \mathbb{R}$ represents the scale factor. We describe the procedure to estimate λ in Section III-E.

We estimate the covariance matrix $\Sigma_{\delta_{k,k-1}} \in \mathbb{R}^{4x4}$ using Monte Carlo technique [21]. The VO at every step of the algorithm provides an incremental estimate $\delta_{k,k-1}$, together with a set of corresponding image points between image I_k and I_{k-1}. We randomly sample five couples from the corresponding point set multiple times (1000 in our experiments). Each time, we use the selected samples as an input to the 5-point algorithm [22] to obtain the estimate $\{\delta_i\}$. All these estimates form $\mathcal{D} = \{\delta_i\}$. Finally, we calculate the uncertainty $\Sigma_{\delta_{k,k-1}}$ of the VO by computing the sample covariance from the data.

The error of the VO is propagated throughout consecutive camera positions as follows. At time k the state $q_{k|k-1}$ depends on $q_{k-1|k-1}$ and $\delta_{k,k-1}$

$$q_{k|k-1} = f(q_{k-1|k-1}, \delta_{k,k-1}), \tag{7}$$

We compute its associated covariance $\Sigma_{q_{k|k-1}} \in \mathbb{R}^{4x4}$ by the error-propagation law:

$$\Sigma_{q_{k|k-1}} = \bigtriangledown f_{q_{k-1|k-1}} \Sigma_{q_{k-1|k-1}} \bigtriangledown f^T_{q_{k-1|k-1}} + \bigtriangledown f_{\delta_{k,k-1}} \Sigma_{\delta_{k,k-1}} \bigtriangledown f^T_{\delta_{k,k-1}}, \tag{8}$$

assuming that $q_{k-1|k-1}$ and $\delta_{k,k-1}$ are uncorrelated. We compute the Jacobian matrices numerically. The rows of the Jacobian matrices $\bigtriangledown ({}^i f_{q_{k-1|k-1}})$, $\bigtriangledown ({}^i f_{\delta_{k,k-1}}) \in \mathbb{R}^{1x4}$ (i = 1,2,3,4) are computed as

$$\bigtriangledown({}^i f_{q_{k-1|k-1}}) = \left[\frac{\partial({}^i f)}{\partial({}^1 q_{k-1|k-1})} \quad \frac{\partial({}^i f)}{\partial({}^2 q_{k-1|k-1})} \quad \frac{\partial({}^i f)}{\partial({}^3 q_{k-1|k-1})} \quad \frac{\partial({}^i f)}{\partial({}^4 q_{k-1|k-1})}\right],$$
$$\bigtriangledown({}^i f_{\delta_{k,k-1}}) = \left[\frac{\partial({}^i f)}{\partial({}^1 \delta_{k,k-1})} \quad \frac{\partial({}^i f)}{\partial({}^2 \delta_{k,k-1})} \quad \frac{\partial({}^i f)}{\partial({}^3 \delta_{k,k-1})} \quad \frac{\partial({}^i f)}{\partial({}^4 \delta_{k,k-1})}\right], \tag{9}$$

where ${}^i q_{k-1|k-1}$ and ${}^i \delta_{k,k-1}$ denote the i-th component of $q_{k-1|k-1}$ respectively $\delta_{k,k-1}$. The function ${}^i f$ relates the updated state estimate $q_{k-1|k-1}$ and the VO output $\delta_{k,k-1}$ to the i-th component of the predicted state ${}^i q_{k|k-1}$.

In conclusion, the state covariance matrix $\Sigma_{q_{k|k-1}}$ defines an uncertainty space (with a confidence level of 3σ). If the measurement z_k that we compute by means of the appearance-based global positioning system is not included in this uncertainty space, we do not update the state and we rely on the VO estimate.

C. Uncertainty estimation of the appearance-based global localization

Our goal is to update the state of the MAV $q_{k|k-1}$, whenever an appearance-based global position measurement $z_k \in \mathbb{R}^4$ is available. We define z_k as

$$z_k = (p^S_k, \theta^S_k), \tag{10}$$

where $p^S_k \in \mathbb{R}^3$ denotes the position and $\theta^S_k \in \mathbb{R}$ the yaw in the global reference system.

The appearance-based global positioning system (Section II) provides the index $j \in \mathbb{N}$ of the Street View image corresponding to the current MAV image, together with two sets of $n \in \mathbb{N}$ 2D corresponding image points between the two images. Furthermore, it provides also the 3D coordinates of the corresponding image points in the global reference system. We define the set of 3D coordinates as $X^S := \{x^S_i\}$ ($\{x^S_i\} \in \mathbb{R}^3 \ \forall\ i = 1, \cdots, n$) and the set of 2D coordinates as $\mathcal{M}^D = \{m^D_i\}$ ($\{m^D_i\}, \in \mathbb{R}^2 \ \forall\ i = 1, \cdots, n$).

If the MAV image matches with a Street View image, it cannot be farther than 15 meters from that Street View camera (c.f. Fig. 3), according to our experiments. We illustrate the uncertainty bound of the MAV in a bird-eye view in

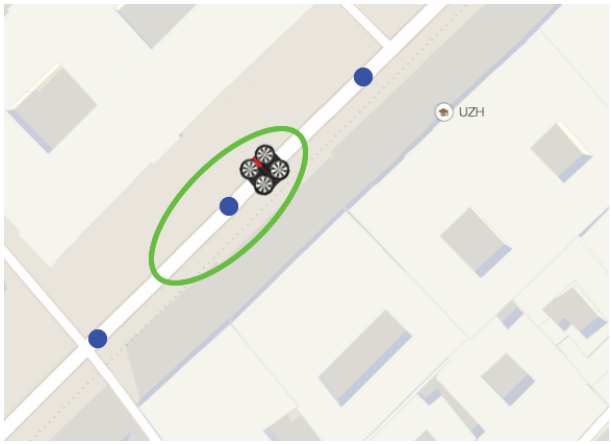

Fig. 6: Blue dots represent Street View cameras. If the MAV current image matches with the central Street View one, the MAV must lie in an area of 15 meters around the corresponding Street View camera. We display this area with a green ellipse.

Fig. 6 with the green ellipse, where the blue dots represent Street View camera positions. To reduce the the uncertainty associated to z_k, we use the two sets of corresponding image points.

We compute z_k such that the reprojection error of X^S with respect to $\mathcal{M}^D$ is minimized, that is

$$z_k = \underset{z}{\operatorname{argmin}}(\sum_{i=1}^{n} ||m_i^D - \pi(x_i^S, z)||), \qquad (11)$$

where π denotes the j-th Street View camera projection model.

The reprojected points coordinates $\pi(x_i^S, z)$ are noisy, because of the uncertainty of the Street View camera poses and of the 3D model data. The $\mathcal{M}^D$, $\mathcal{X}^S$ sets may contain outliers. We choose then P3P-RANSAC (Section II-C) to compute z_k, selecting the solution with the highest consensus (maximum number of inliers, minimum reprojection error).

Similarly to Section III-B, we estimate the covariance matrix $\Sigma_{z_k} \in \mathbb{R}^{4x4}$ using Monte Carlo technique. We randomly sample five couples of corresponding points from $\mathcal{M}^D$, $\mathcal{X}^S$ multiple times (1000 in the experiments). Each time, we use the selected samples as an input to the P3P algorithm, to obtain the measurement $\{z_i\}$. As we can see in Fig. 3, a match with images gathered by Street View cameras farther than twenty meters is not plausible. We use this criterion to accept or discard $\{z_i\}$ measurements. All the plausible estimates form the set $\mathcal{Z} = \{z_i\}$. We estimate Σ_{z_k} by computing the sample covariance from the data.

D. Fusion

We aim to reduce the uncertainty associated to the state by fusing the prediction estimate with the measurement, whenever an appearance-based global position measurement is available. The outputs of this fusion step are the updated estimate $q_{k|k}$ and its covariance $\Sigma_{q_{k|k}} \in \mathbb{R}^{4x4}$. We compute them according to Kalman filter equations [23]:

$$q_{k|k} = q_{k|k-1} + \Sigma_{q_{k|k-1}}(\Sigma_{q_{k|k-1}} + \Sigma_{z_k})^{-1}(z_k - q_{k|k-1}) \quad (12)$$

$$\Sigma_{q_{k|k}} = \Sigma_{q_{k|k-1}} - \Sigma_{q_{k|k-1}}(\Sigma_{q_{k|k-1}} + \Sigma_{z_k})^{-1}\Sigma_{q_{k|k-1}} \quad (13)$$

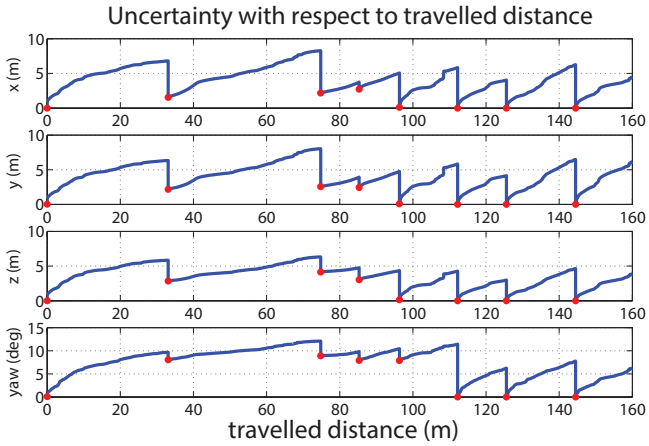

Fig. 7: Uncertainty along the x, y, z-axis and yaw, with respect to the traveled distance.

E. Initialization

In order to initialize our system we use the global localization algorithm, namely we use (11) to compute the initial state $q_{0|0}$ and the Monte Carlo procedure described in III-C to estimate its covariance $\Sigma_{q_{0|0}}$. In the initialization step, we also estimate the absolute scale factor λ for Visual Odometry. After the initial position we need another position of the MAV that is globally localized by our appearance-based approach. Finally, we compute λ by comparing the metric distance traveled computed by the two global localization estimates, with the unscaled motion estimate returned by the VO.

IV. EXPERIMENTS AND RESULTS

A. The experimental dataset

We collected a dataset in downtown Zurich, Switzerland using a commercially available Parrot AR.Drone 2. The flying vehicle was manually piloted along a 150 m trajectory, collecting images throughout the environment at different altitudes. The MAV camera was always side-looking, facing the buildings. We considered a street in which we could receive also the GPS signal. Although we do not have accurate ground truth data to compare with (GPS has very limited accuracy in urban environments due to multipath), we can evaluate our algorithm visually. We also plot the obtained trajectories in the cadastral 3D city model, where the surrounding buildings can give an intuitive measure about the MAV positions. The memory requirements of the appearance-based global localization system are moderate, since only the image features extracted offline from the Street View images and their real world coordinates obtained form the cadastral 3D city model are stored in order to localize the MAV. The rendered views showing the textured model together with the MAV trajectories are for a better evaluation of the proposed solutions.

B. Results

Even though, we do not have a ground-truth path of the MAV to compare with, we can still evaluate visually the performance of our system. Furthermore, we display our

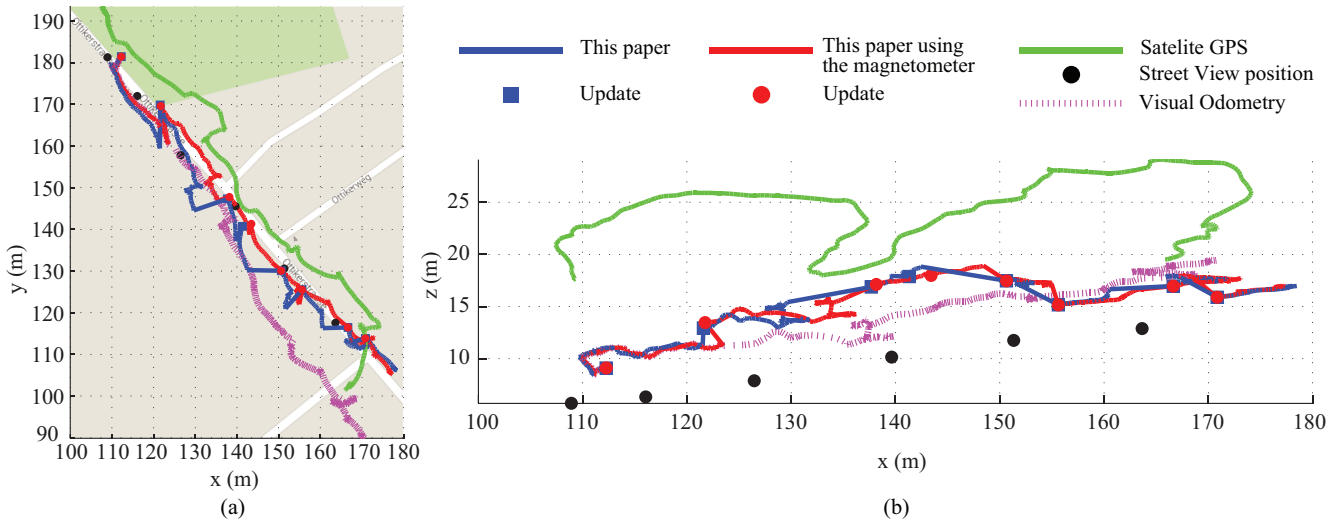

Fig. 8: Comparison between the estimated trajectories: (a) bird eye-view perspective of the results overlayed on Google Maps; (b) side view of (a); black circles represent the Street View camera positions, note that the terrain is ascendant, consequently, we measure the altitude above the ground; we display the trajectory measured using the on-board GPS with green; the path estimate obtained with the system described in this paper is with blue, the squares identify the state updates; we show with red the enhanced version of our path estimate, computed using the on-board magnetometer data to estimate the yaw of the MAV, red circles identify the state updates; finally, the magenta displays the estimate given by pure Visual Odometry.

result within the very accurate 3D city model, which can give a good basis to evaluate the result.

We show the results using a bird-eye-view perspective, overlayed on Google Maps in Fig. 8 (a). Fig. 8 (b) shows a side-view of the same figure, where the flying altitude can be observed. We display the trajectory measured by the on-board GPS in green. Note that the GPS measurement shows a very high altitude compared to the real one, which never exceeds 6 meters relative to the ground. Note that, although the terrain is uphill, the Street View images are recorded at the same altitude. We show the path estimated by the algorithm described in this paper with blue. The blue squares mark the positions where an update from the appearance-based global localization system was integrated into the Bayesian tracking to correct the trajectory. Although in these states the position was accurately detected, the orientation of the camera was occasionally not very accurate, resulting in a non-smooth trajectory.

To improve our results, we computed the orientation of the MAV in the update positions (red dots in Fig. 8 (a) and (b)), using the on-board magnetometer data to correct the yaw angle of the vehicle. Thereby, the best trajectory was obtained in comparison with the actual one. We show the uncertainty of the position and yaw in Fig. 7. Along the considered path, the average uncertainty was 4.2 m in x, 4.3 m in y, and 3.6 m in z. After each state update, highlighted by red dots in Fig. 7, the uncertainty drops lower than 5 m. The yaw uncertainty never exceeds 12.2 degrees, and it is about 7.5 degrees in average. When the Visual Odometry (magenta in Fig. 8 (a) and (b)) is not updated by the appearance-based global-positioning system, a very big error is accumulated in the trajectory of the MAV. Conversely, when the proposed appearance-based localization is used, the trajectory of the MAV is corrected as it shown in 8 (a) for the x, y coordinate, respectively 8 (b) for the z coordinate.

In Fig. 9 (a)-(c), we display the results using the cadastral 3D model, in order to evaluate the trajectories with respect to the surrounding buildings. The Visual Odometry estimate is shown in black, the GPS in green, and our estimate in blue. The altitude estimate error of the GPS is even more notable, while the VO estimate penetrates the buildings. The estimated trajectory with the proposed algorithm is the most plausible, is the most similar to the actual one.

Finally, the rendered view of the textured 3D model—which the MAV perceives at the end of the trajectory—is visually more similar to the real one (Fig. 9 (d)), in case it is estimated by the presented algorithm (Fig. 9 (e)), in comparison with the rendered view computed based on the GPS measurement (Fig. 9 (f)).

V. CONCLUSIONS

This paper presented a solution to localize MAVs in urban environments with respect to the surrounding buildings using a single on-board camera and priorly geo-tagged street-level images together with a cadastral 3D city model. A reliable alternative to satellite-based global positioning was introduced, which is purely based on appearance. Moreover, it was shown that the performance of such a system can be increased by making use of additional on-board sensors such as the magnetometer. The presented appearance-based positioning described in this paper can be of great importance to safely operate—to takeoff, land, and navigate—small-sized, autonomous aerial vehicles in urban environments equipped with vision cameras.

ACKNOWLEDGEMENTS

The authors are grateful to Aparna Taneja and Luca Ballan for providing corrected, accurate Google Street View image positions for the data used in this work.

REFERENCES

[1] A. Majdik, Y. Albers-Schoenberg, and D. Scaramuzza, "Mav urban localization from google street view data," in *IEEE International Conference on Intelligent Robots and Systems*, 2013, pp. 3979 – 3986.

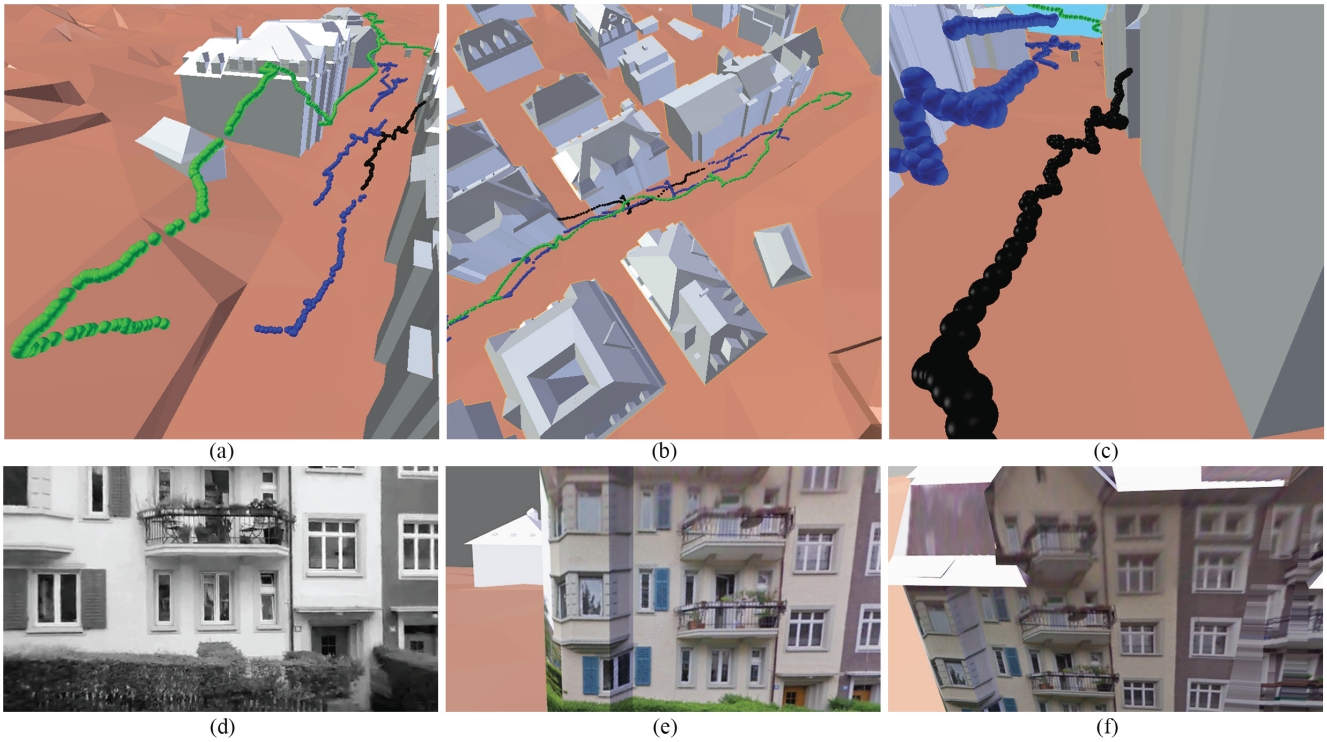

Fig. 9: Comparison between path estimates shown within the cadastral 3D city model: Top row: we display the Visual Odometry estimate in black, GPS in green, our estimate in blue; (a) altitude evaluation: in the experiment, the MAV flew close to the middle of the street and it never flew over the height of 6 m (above the ground), from this point of view, our path estimate (blue) is more accurate than the GPS one (green); (b) perspective view of the path estimates; (c) trajectory zoom: the pure VO trajectory penetrates one of the surrounding buildings, using the proposed method, we are able to reduce the drift of the VO; Bottom row: we show a visual comparison of the: (d) actual view; (e) rendered view of the textured 3D model corresponding to (d), which the MAV perceives according to our estimate; (f) rendered view of the textured 3D model corresponding to (d) which the MAV perceives according to the GPS measurement; to conclude, the algorithm presented in this paper outperform the other techniques to estimate the trajectory of the MAV flying at low altitudes in urban environment.

[2] M. A. Brubaker, A. Geiger, and R. Urtasun, "Lost! leveraging the crowd for probabilistic visual self-localization," in *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2013.

[3] B. L. G. Floros, B. van der Zander, "Openstreetslam: Global vehicle localization using openstreetmaps," in *International Conference on Robotics and Automation*, 2013.

[4] M. Hentschel and B. Wagner, "Autonomous robot navigation based on openstreetmap geodata," in *Intelligent Transportation Systems (ITSC), 2010 13th International IEEE Conference on*, 2010, pp. 1645–1650.

[5] R. Kuemmerle, B. Steder, C. Dornhege, A. Kleiner, G. Grisetti, and W. Burgard, "Large scale graph-based SLAM using aerial images as prior information," *Journal of Autonomous Robots*, vol. 30, no. 1, pp. 25–39, 2011.

[6] D. Anguelov, C. Dulong, D. Filip, C. Frueh, S. Lafon, R. Lyon, A. Ogale, L. Vincent, and J. Weaver, "Google street view: Capturing the world at street level," *Computer*, vol. 43, no. 6, pp. 32–38, 2010.

[7] M. Cummins and P. Newman, "Appearance-only slam at large scale with fab-map 2.0," *International Journal of Robotics Research*, vol. 30, no. 9, pp. 1100–1123, Aug. 2011.

[8] M. Milford, "Vision-based place recognition: how low can you go?" *International Journal of Robotics Research*, vol. 32, no. 7, pp. 766–789, 2013.

[9] J.-M. Morel and G. Yu, "Asift: A new framework for fully affine invariant image comparison," *SIAM J. Imaging Sciences*, vol. 2, no. 2, pp. 438–469, 2009.

[10] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *I. J. of Computer Vision*, vol. 60, no. 2, pp. 91–110, 2004.

[11] Z. Liu and R. Marlet, "Virtual line descriptor and semi-local graph matching method for reliable feature correspondence," in *in British Machine Vision Conference*, 2012, pp. 16.1–16.11.

[12] "Formulas and constants for the calculation of the swiss conformal cylindrical projection and for the transformation between coordinate systems," Federal Department of Defence, Civil Protection and Sport DDPS, Tech. Rep., 2008.

[13] A. Taneja, L. Ballan, and M. Pollefeys, "Registration of spherical panoramic images with cadastral 3d models," in *3D Imaging, Modeling, Processing, Visualization and Transmission (3DIMPVT), 2012 Second International Conference on*. IEEE, 2012, pp. 479–486.

[14] 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.

[15] L. Kneip, D. Scaramuzza, and R. Siegwart, "A novel parametrization of the perspective-three-point problem for a direct computation of absolute camera position and orientation," in *Proc. of The IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, Colorado Springs, USA, June 2011.

[16] F.Moreno-Noguer, V.Lepetit, and P.Fua, "Accurate non-iterative o(n) solution to the pnp problem," in *IEEE International Conference on Computer Vision*, Rio de Janeiro, Brazil, October 2007.

[17] R. Hartley and A. Zisserman, *Multiple View Geometry in Computer Vision*, 2nd ed. New York, NY, USA: Cambridge University Press, 2003.

[18] S. Thrun, W. Burgard, D. Fox, *et al.*, *Probabilistic robotics*. MIT press Cambridge, 2005, vol. 1.

[19] D. Scaramuzza and F. Fraundorfer, "Visual odometry [tutorial]," *Robotics & Automation Magazine, IEEE*, vol. 18, no. 4, pp. 80–92, 2011.

[20] C. Wu, S. Agarwal, B. Curless, and S. M. Seitz, "Multicore bundle adjustment," in *In IEEE Conference on Computer Vision and Pattern Recognition (CVPR*. IEEE, 2011, pp. 3057–3064.

[21] S. Thrun, D. Fox, W. Burgard, and F. Dellaert, "Robust monte carlo localization for mobile robots," *Artificial intelligence*, vol. 128, no. 1, pp. 99–141, 2001.

[22] D. Nistér, "An efficient solution to the five-point relative pose problem," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 26, no. 6, pp. 756–770, 2004.

[23] R. E. Kalman *et al.*, "A new approach to linear filtering and prediction problems," *Journal of basic Engineering*, vol. 82, no. 1, pp. 35–45, 1960.