A Proof-Of-Concept Demonstration of Visual Teach and Repeat on a Quadrocopter Using an Altitude Sensor and a Monocular Camera

Andreas Pfrunder
ETH Zurich
Zurich, Switzerland
andrepfr@student.ethz.ch

Angela P. Schoellig, Timothy D. Barfoot
University of Toronto Institute for Aerospace Studies
Toronto, Ontario, Canada
schoellig@utias.utoronto.ca, tim.barfoot@utoronto.ca

***Abstract*—This paper applies an existing vision-based navigation algorithm to a micro aerial vehicle (MAV). The algorithm has previously been used for long-range navigation of ground robots based on on-board 3D vision sensors such as a stereo or Kinect cameras. A teach-and-repeat operational strategy enables a robot to autonomously repeat a manually taught route without relying on an external positioning system such as GPS. For MAVs we show that a monocular downward-looking camera combined with an altitude sensor can be used as 3D vision sensor replacing other resource-expensive 3D vision solutions. The paper also includes a simple path-tracking controller that uses feedback from the visual and inertial sensors to guide the vehicle along a straight and level path. Preliminary experimental results demonstrate reliable, accurate and fully autonomous flight of an 8-m-long (straight and level) route, which was taught with the quadrocopter fixed to a cart. Finally, we present the successful flight of a more complex, 16-m-long route.**

***Keywords*-Vision-Based Flight; Monocular Camera; Quadrocopter;**

Figure 1. Experiments are conducted at the University of Toronto Institute for Aerospace Studies (UTIAS) using the Parrot AR.Drone 2.0 platform, which is equipped with a downward-facing, monocular camera and altitude sensors. The picture also shows the texture of the floor of the VT&R experiments around the quadrocopter.

I. Introduction

In GPS-denied environments, long-range robot navigation is a challenging task. While relative motion estimation systems consisting of a combination of visual, inertial and odometric sensors have become increasingly accurate, the error in the position estimation still grows unbounded if no global corrections are made.

One possibility for long-range robot navigation in GPS-denied environments is to build a manifold map [1] composed of overlapping submaps from on-board vision sensors. This approach has been presented by Furgale and Barfoot [2], where they used a stereo-camera-equipped ground robot with a teach-and-repeat operational strategy for autonomous long-range navigation. In the learning (teach) pass the rover was human-piloted along a desired route while the manifold map was built. During the repeat pass this map is used for localization and navigation. The use of manifold mapping allows repeating a route without requiring a globally consistent map.

In this work, we apply for the first time the visual teach and repeat (VT&R) algorithm to a micro aerial vehicle (MAV). We replace the 3D position sensor with a monocular downward-looking camera and an altitude sensor. As a platform we use a Parrot AR.Drone 2.0 (see Fig. 1), a low-cost off-the-shelf quadrocopter equipped with various sensors, cameras and an on-board stabilization system [3].

Flying vehicles have become increasingly popular in the last ten years due to technical advances in sensors, actuators, materials, processors and battery technology [4]. Potential applications range from surveillance, monitoring to remote sensing (for example, for precision agriculture) and aerial mapping. Compared to ground vehicles, flying vehicles have the advantage to access and investigate areas, which are inaccessible to ground vehicles including challenging terrain. However, one major challenge when aiming for real-world autonomous flight applications is the reliable localization and navigation in areas where GPS is not available or not accurate enough. In these applications, VT&R can be a resource-effective alternative to other localization approaches such as external camera-based localization [5], which is usually a rather expensive solution, or Simultaneous Localization and Mapping (SLAM), which aims to generate a global map and requires extensive computation and data

978-1-4799-4337-1/14 $31.00 © 2014 IEEE
DOI 10.1109/CRV.2014.40

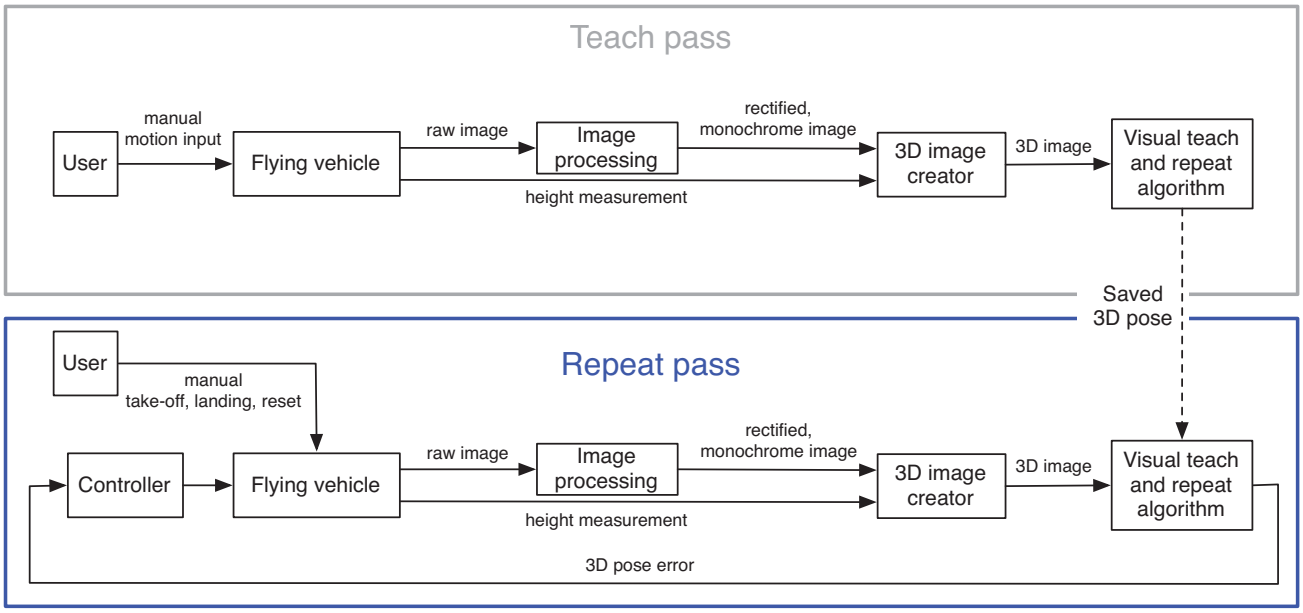


Figure 2. Block diagram of the proposed visual teach and repeat setup for the AR.Drone 2.0. In the teach pass (grey box) we pilot the quadrotor vehicle along a desired path in order to create a map. In this work, we taught the route by mounting the AR.Drone 2.0 to a cart and driving the cart along the desired route. Teaching a route by manually flying is another option. The created map is saved and used later for localization in the repeat pass (blue box), where the quadrocopter autonomously repeats the previously learned route. During repeat, the height is kept constant by the built-in altitude controller.

storage.

In the following sections, we will first provide a literature review on visual navigation (see Section II). In Section III we describe our approach to autonomous, vision-based MAV navigation including a description of the quadrocopter-specific VT&R algorithm (see Section III-A), a list of the assumptions we made (see Section III-B) and finally a description of the position controller guiding the quadrotor vehicle along the desired path during repeat (see Section III-C). In Section IV we describe the hardware and software of this project (see Sections IV-A, IV-B) and present experimental results to show the reliable working system (see Sections IV-C, IV-D). Results are discussed in Section V. We comment on future work and summarize our results in Section VI and VII, respectively.

II. Related Work

The VT&R algorithm used in this project was first presented in [6], where a ground robot equipped with a stereo camera was used to perform VT&R. In this paper, we use a quadrocopter with a monocular, downward-facing camera and an altitude sensor to perform 3D localization. This is the first time VT&R is applied to a flying vehicle where we face new difficulties such as more degrees of freedom and faster dynamics in all three dimensions. The image-processing part of the underlying algorithm remains similar. One major difference is the path-tracking controller that is used to keep the quadrocopter on track during repeat. In [2], a dynamically changing linear velocity depending on the current path difficulty was commanded for forward/backward movements. The path was intercepted by controlling the heading of the ground robot. In our approach, we control both the heading and the sideways velocity to intercept the path. The heading controller aligns the quadrocopter heading with the path direction and the sideways velocity controller intercepts the path using roll movements (see Fig. 4). Thanks to the stereo camera, a real depth image was obtained in [2]. In contrast, we compose a 'fake' 3D image by assigning the same depth obtained from the altitude sensor to all pixels of the monocular image (see Section III-A).

A lot of work has been done in the area of MAV navigation. Several publications focus on autonomous navigation without previous knowledge about the environment nor use of GPS and therefore only relying on on-board sensors. In [7], a miniature laser range finder is used to perform SLAM in indoor environments. Likewise a Kinect camera is used in [8] to create a 3D map of the environment, which allows the MAV to autonomously navigate in 3D space. The same goal is achieved in [9] by using a quadrocopter equipped with a stereo camera. However, the inherent drawback of mapping systems based on lidars, kinects or stereo cameras is high power consumption, high cost, high weight, and therefore limited flight time. In this paper, we use a light-weight, low-cost off-the-shelf quadrocopter with monocular vision instead.

MAVs equipped with monocular cameras are used for

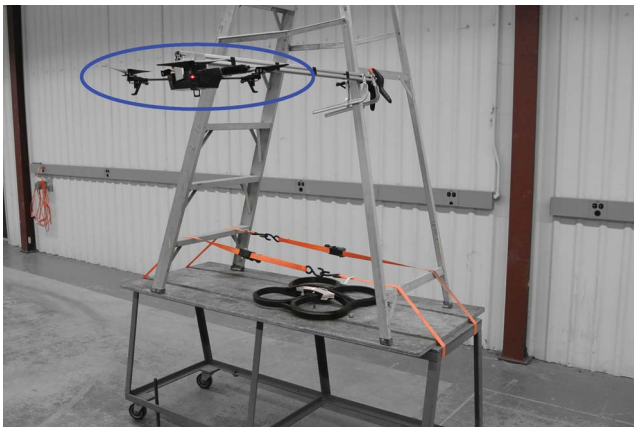


Figure 3. The quadrocopter (highlighted in blue) is mounted on a cart during the teach phase in order to teach a straight path at constant height.

SLAM in [10], [11], [12], [13], [14] and [15]. A different approach is presented in [16], where no map is built but a perspective cue vision algorithm is used to navigate in unknown indoor environments. In this work, we do not try to navigate in a completely unknown environment, but we focus on autonomous repeating of a previously learned/mapped route. Similar to our approach, a map and replay method for a quadrotor vehicle was presented in [17], which uses monocular vision to correct the robot's heading and dead reckoning (process of calculating the current position based on the previous position and advancing this with estimated speed over elapsed time and course) to estimate the distance travelled (cf. [18]). This approach can lead to large localization errors due to the integration of potentially noisy velocity measurements. Our approach is different, as we use the downward-facing camera together with the altitude sensor to create a 3D map of interest points. Localization is performed in 3D space against the 3D map obtained in the teach pass and as a result we obtain a 3D pose error. We use Visual Odometry (VO) to predict the vehicle position and periodic localization against the map to ensure global (topological) accuracy (see Section III-A). Accordingly, we achieve small localization errors, since localization errors are not accumulated over time.

III. Methodology

To achieve autonomous flight, we use a VT&R algorithm for localization and derive a simple path-following controller for straight and level routes.

A. Visual Teach and Repeat (VT&R) with the AR.Drone 2.0

Localization for the flying vehicle is provided by the VT&R algorithm presented in [6], a slightly modified version of [2]. In [6] an appearance-based lidar was used to create a 3D map. In this paper, we solely rely on the vehicle's monocular, downward-facing camera and its altitude sensor (see Fig. 2) for mapping. We use the raw image from the quadrocopter bottom camera to determine the image location of an interest point and the built-in altitude sensor to obtain its depth.

In the first operational phase of the VT&R algorithm, the teach phase, the vehicle is manually piloted along the desired path by an operator. In fixed intervals (defined by travelled distance) a new reference system, a keyframe, is created by extracting the interest points (features) of the current image and storing their 3D positions together with the transformation between the current and the last reference system (obtained from VO). During the second operational phase, the repeat phase, the vehicle re-localizes against the stored keyframes. More precisely, we extract interest points from the live image and compare them to the stored interest points, the map. The live image is not only compared to the nearest keyframe but to a locally consistent map that contains additional information from surrounding keyframes. The entire image processing is based on the Speeded-Up Robust Features (SURF) algorithm [19] and uses the rotation-variant upright descriptor. The strength of the VT&R algorithm is that visual odometry is used in combination with periodic localization updates against the stored map, which ensures global accuracy while repeating long paths.

The proposed VT&R setup for the AR.Drone 2.0 is shown in Fig. 2. We taught the route by mounting the quadrocopter to a cart and driving the cart along the desired route (see Fig. 3). In the future, we intend to teach the quadrocopter by flying manually, for example, with an xbox controller. After image processing the rectified, monochrome image is used together with the altitude measurement to create a 'fake' 3D depth image. During the repeat phase, the VT&R algorithm calculates a 3D pose error. However, the path-tracking controller (see Section III-C) controls the vehicle only in the horizontal plane based on the heading and lateral errors from the VT&R. The vertical direction is controlled separately by Parrot's built-in controller, which uses the measurements from the on-board altitude sensors for feedback. The quadrocopter is kept on track using pitch angle commands, roll angle commands, and turn rate commands around the body's z-axis (see Fig. 4), which are the outputs of our controller. The fact that localization is performed in 3D space but the VT&R algorithm does not calculate a vertical error is a legacy feature from the ground-robot VT&R and should be changed in the future.

B. Assumptions

For this proof-of-concept demonstration, we made several assumptions to limit the project scope. However, extensions to more general scenarios are relatively straight-forward and planned in the future.

We assume that the image plane and therefore the quadrocopter itself is parallel to the ground. Furthermore, we

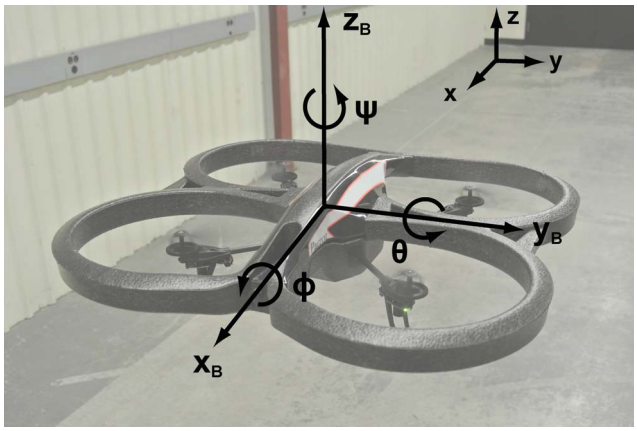


Figure 4. Inertial and body-fixed coordinate systems with roll angle ϕ, pitch angle θ and yaw angle ψ (assuming small angles).

assume that the ground is flat. Those two assumptions together allow us to assign the same depth (obtained from the altitude sensor) to every pixel in the image plane. Obviously, the first assumption is not true during flight but with small pitch and roll angles this is still a reasonable approximation. In addition, we assume that the height of the quadrocopter remains constant during teach and repeat. This is important as path-tracking is currently implemented in a projected plane not considering the height of the vehicle. Furthermore, we assume a straight path in the derivation of the path-tracking controller (see Fig. 5). However, we have shown in experiments that the controller also works for piecewise straight paths.

C. Path Definition and Control

First, we define an inertial and a body-fixed coordinate system to specify the vehicle's attitude and position in three dimensions (see Fig. 4). We use ZYX-Euler angles [20] to specify the attitude of the quadrocopter. Assuming small angles, the Euler angles, roll, pitch and yaw, define rotations around the x_B-, y_B- and z_B-axis, respectively.

We present three decoupled feedback controllers to control pitch, roll and yaw. With the assumption of small angles, the simplified linear quadrocopter dynamics are governed by

$$\ddot{x}(t) = g\theta(t) \quad (1)$$

$$\ddot{y}(t) = -g\phi(t) \quad (2)$$

$$\dot{\psi}(t) = \omega(t), \quad (3)$$

where x and y are the vehicle's position in the inertial coordinate system, g is the gravitational constant, θ, ϕ, ω are the pitch angle, roll angle and angular speed around the z_B-axis, respectively. Note that we neglect the dynamics in z-direction, since we use the built-in altitude controller of the AR.Drone 2.0.

The desired path that is taught by a human operator is assumed to be a straight line in the inertial coordinate system (see Fig. 5) with

$$(\, x_d(t), \quad y_d(t) = y_d, \quad z_d(t) = z_d, \quad \psi(t) = 0\,), \quad (4)$$

where y_d and z_d are constants, $y_d, z_d \in \mathbb{R}$.

For the autonomous repeat, the controller aims to fly the vehicle along the taught path with a constant, predefined velocity, v_d. In our derivations of the path-following controller, we assume that $\theta = \theta_{\text{cmd}}$, $\phi = \phi_{\text{cmd}}$ and $\omega = \omega_{\text{cmd}}$. That is, we assume that its dynamics are much faster than the position controller we design below.

Based on (1) we derive a feedback controller for the vehicle motion in the x-direction, which uses the current forward velocity estimate $v(t)$ of the quadrocopter in the x-direction to calculate the pitch angle to be sent to the vehicle:

$$\theta_{\text{cmd}} = \frac{1}{gT_\theta}(v_d - v(t)). \quad (5)$$

The velocity, $v(t)$, is obtained from the quadrocopter's onboard estimator and the pitch time constant, $T_\theta > 0$, a tuning parameter.

In order to achieve path tracking in the y-direction, we design a controller that responds to lateral errors, $\epsilon_L(t) = y(t) - y_d$, as a second-order system with a time constant $T_\phi > 0$, and a damping ratio $\zeta_\phi > 0$:

$$\phi_{\text{cmd}} = \frac{1}{g}\left[2\zeta_\phi\left(\frac{2\pi}{T_\phi}\right)\dot{\epsilon_L}(t) + \left(\frac{2\pi}{T_\phi}\right)^2 \epsilon_L(t)\right], \quad (6)$$

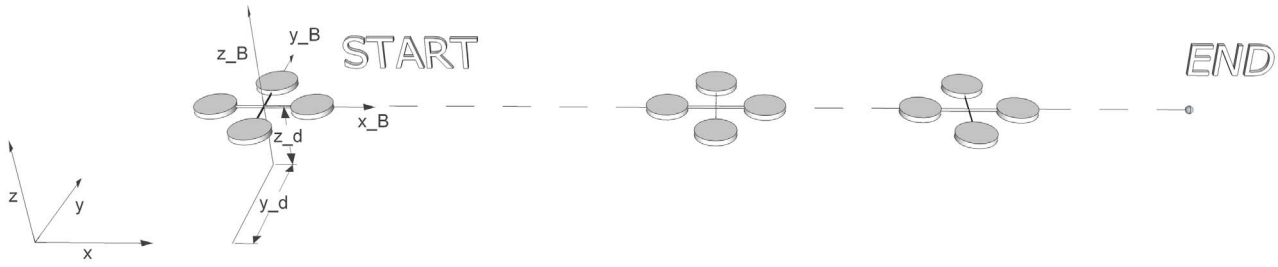


Figure 5. For the controller design we assume that the taught path is a straight line at a constant altitude.

where ϵ_L is obtained from the VT&R algorithm and $\dot{\epsilon_L}$ is the sideways velocity obtained from the on-board velocity estimator of the Parrot AR.Drone. A damping ratio, ζ_ϕ, between 0.7 (underdamped) and 1 (critically damped) is a reasonable choice.

To keep the yaw angle close to zero, we design a proportional controller to minimize the heading error, $\epsilon_H(t) = \psi(t) - \psi_d$, in the repeat pass:

$$\omega_{\text{cmd}} = -\frac{1}{T_\omega}\epsilon_H(t), \tag{7}$$

where the time constant, $T_\omega > 0$, is again a tuning parameter.

IV. Experimental Results

Several experiments were conducted to evaluate the performance of the vision-based localization. Successful autonomous flight was demonstrated along a straight and level path (cf. Fig. 5), which was previously taught with the quadrocopter mounted to a cart (see Fig. 3).

A. Hardware

For our experiments, we used a Parrot AR.Drone 2.0 quadrocopter (see Fig. 1). The vehicle is equipped with two height sensors [21], an ultrasonic sensor and a pressure sensor. An altitude estimation algorithm on-board the vehicle fuses both sensor measurements and uses the resulting estimate for controlling the height of the vehicle. In addition, velocity estimates (based on three-axis accelerometer measurements) are computed on-board the vehicle and sent over wireless to our off-board controller (see Section III-C). Moreover, the quadrocopter has a downward-facing camera with a $64\,^\circ$ wide-angle diagonal lens, a video frequency of 60 frames per second and a resolution of 320x240 pixels (QVGA). During this project we ran the AR.Drone 2.0 on the firmware version 2.3.3.

The off-board image processing and position control ran on a MacBookPro 4.1 with a 2.4 GHz Intel Core 2 Duo processor, 2 GB RAM and NVIDIA GeForce 8600M GT (256 MB). In addition, we used the NVIDIA's Compute Unified Device Architecture (CUDA) toolkit as the SURF part of the VT&R algorithm is directly implemented on the graphics processing unit (GPU).

B. Software

Our algorithm implementation used ROS, an open-source robot operating system [22]. More precisely, we used ROS Electric, installed on a 64-bit 10.04 Ubuntu version. In addition, we used the ROS *ardrone_autonomy* package [23] to interface with the AR.Drone. Small modifications were made to the package including a message that reports the quadrocopter height not only during flight but also when the motors are shut off. This is required to create 3D images during the teach pass when the quadrotor vehicle is mounted to the cart (see Fig. 3). We calibrated the bottom camera in advance, using the ROS *camera_calibration* package [24]. The rectified image was turned into a monochrome image using the ROS *image_proc* package [25].

C. Localization

In the current implementation of the VT&R strategy, we created a new keyframe every 0.2 m travelled and every $3\,^\circ$ of change in yaw. We found from experiments that a successful localization at a height of about 1.75 m was possible if the pitch angle of the quadrocopter was smaller than $8\,^\circ$ and the roll angle smaller than $10\,^\circ$. This is due to the current feature selection and the camera's field of view. However, these limits match well to our small angle assumption in Section III-B.

For experimental validation, we logged the lateral and heading error. In addition, the visual odometry (VO) and the map match failures were stored. VO failures happen when the algorithm fails to track the interest points (features) from one to the next image. As we are currently feeding raw images into the VT&R algorithm at 20 Hz, the flown distance between two images is very small and we nearly never experienced VO failures. Similar to VO failures, map match failures happen when the algorithm is unable to track the features from the current image to the last keyframe. Therefore, map match failures are more likely to happen, as the features need to be tracked over a distance of 0.2 m.

The goal of our first two sets of experiments was to prove that the quadrocopter successfully localizes using the VT&R algorithm with the on-board camera and altitude sensors. We first used the cart (see Fig. 3) to teach an 8-m-long straight route. This exact same path was repeated ten times consecutively with the vehicle still attached to the cart to show that the localization works on a nominal repeat pass. In order to repeat exactly the same route, it was taught and repeated along multiple tables using them as a side boundary. The average result over ten repeats is shown in Table I under 'Nominal'. As expected, the VT&R algorithm reports small lateral and heading errors (small mean and standard deviation), and shows no VO and only few map match failures. These experimental results prove the reliability of the VT&R algorithm in ideal conditions when exactly repeating the taught route.

A second set of five consecutive off-nominal repeats with the vehicle on the cart was performed to show the robustness of the localization. The off-nominal repeats were conducted along multiple tables off-set by about 0.2 m from the nominal path to the right. Therefore, we expect a lateral error of -0.2 m while no change in the average heading error is expected. In Table I, the average result over five repeats is shown under 'Off-nominal'.

D. Autonomous Flight

Ten autonomous repeats of the taught path in Section IV-C were performed using the controller presented in Sec-

Table I
PERFORMANCE OF THE VT&R ALGORITHM IN EXPERIMENTS.
Under 'Nominal', we show the error obtained by exactly repeating the taught route with the quadrotor vehicle fixed to the cart. Under 'Off-nominal', we show experimental data for the case when repeating the route with an off-set of -0.2 m, still using the cart. Under 'Flying', data is shown of the quadrocopter repeating the route flying fully autonomously.

Repeat	Mean [m]	Std. dev. [m]
Nominal	-0.0041	0.0186
Off-nominal	-0.1908	0.0266
Flying	-0.0197	0.1031

(a) Lateral error.

Repeat	Mean [deg]	Std. dev. [deg]
Nominal	0.8660	0.4903
Off-nominal	0.3151	0.5618
Flying	0.9944	1.6901

(b) Heading error.

Repeat	VO [%]	Map match [%]
Nominal	0	1.6882
Off-nominal	0.1325	1.3100
Flying	0	20.8692

(c) Localization failure.

tion III-C with: $T_\theta = 0.7\,\text{s}$, $v_d = -0.3\,\text{m/s}$, $\zeta_\phi = 1$, $T_\phi = 5\,\text{s}$ and $T_\omega = 1\,\text{s}$.

Not surprisingly, the errors during autonomous flight were slightly larger, and we experienced more map failures compared to the on-the-cart repeats of Section III-C. Nevertheless, all ten consecutive repeats were successful, meaning the quadrocopter always reached the end of the path, which proves the successful working of the vision-based localization combined with the path-following controller. The results are shown in Table I under 'Flying'.

Finally, the VT&R algorithm was also tested on a 16-m-long path with two 45° angle bends (see Fig. 7b). Similar to the previous experiments, we taught the route with the vehicle attached to the cart and repeated it flying autonomously. We repeated the route multiple times using the same desired velocity v_d as before. All repeats were successful. We then increased the desired velocity v_d of the quadrocopter during repeat by a factor of four to 1.2 m/s. All repeats at the higher speed were still successful. We created a short movie of the performance (see Fig. 7). It is found at:

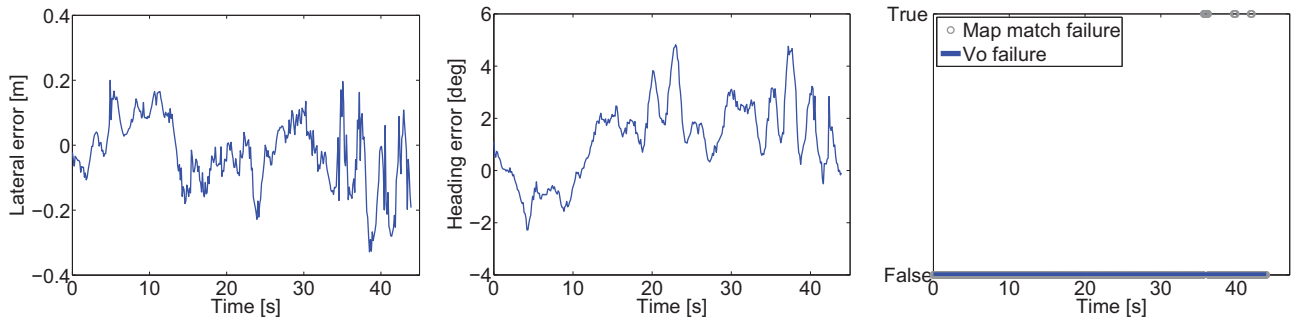
http://youtu.be/BRDvK4xD8ZY

V. DISCUSSION

As stated in Section IV, teaching was done with the quadrocopter mounted to a cart and not flying manually (as typically intended by the VT&R algorithm). The reason was that we made two assumptions in our proof-of-concept implementation that were hard to achieve via manual control: first, we assumed that the quadrocopter was parallel to the ground in Section III-B and second, we assumed a straight path in Section III-C.

In Table Ia and b we show the average lateral and heading error for nominal, off-nominal and flying repeats. For the nominal and the off-nominal repeat the standard deviations mainly resulted from the localization algorithm and are very small, which indicates that the localization works as intended. For the flying repeat, the standard deviations increased slightly. The increase reflects the controller's tracking performance. A typical plot of the lateral and heading error estimates is shown in Fig. 6a and b.

The map failures that we experienced during testing are shown in Table Ic. VO failures are very rare. We experience occasional map match failures in the nominal and the off-nominal repeat pass. This is not unusual; similar behaviour has been described in [2]. In the flying repeat pass, map match failures happened approximately 21% of the time; at those times the vehicle was not able to localize against the map obtained during the teach phase but instead relied on VO only for pose estimation. The highest experienced map match failure was 76%. Nevertheless, all ten flying repeats were successful, which highlights the importance of VO in



(a) Lateral error. (b) Heading error. (c) Localization failure.

Figure 6. Performance plots, obtained during a typical flight repeat pass. The 8-m-long straight route was taught with the quadrocopter mounted to a cart (see Fig. 3) and afterwards repeated autonomously.

the localization algorithm [2].

Experiments showed that the altitude at which the VT&R algorithm is performed was crucial. Teaching at a high altitude increases the camera's field of view and results in fewer map match failures. Depending on the height used during the teach pass, different sizes of interest points are used. The higher we fly, the bigger the feature must be in order to be detected by the SURF algorithm. The resolution of the bottom camera is, as mentioned in Section IV-A, very low, and therefore small features are not visible at high altitudes. To summarize, if enough big interest points are available, it is beneficial to fly high. The characteristics of the floor play an important role as well. Highly repetitive or untextured ground is to be avoided.

Since we created a 'fake' depth image by assigning the same altitude to all pixels in the image plane, the localization algorithm cannot deal well with surface height changes. A possible solution for high-altitude flights is to exclusively rely on the pressure sensor for altitude estimation, which has a reasonable accuracy for higher altitudes. Hence, ground height changes would not affect the 'fake' depth image.

VI. Future Work

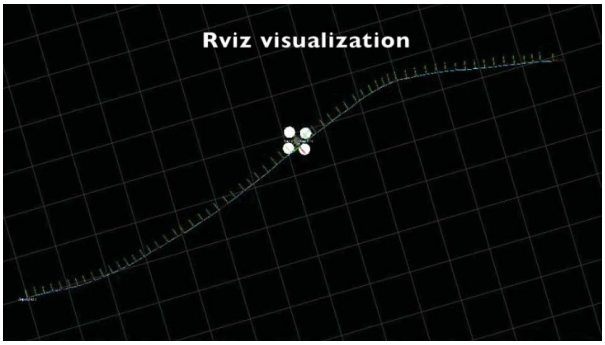
This paper described preliminary results obtained from implementing, for the first time, a VT&R algorithm [6] on a quadrocopter. In Section III-B we summarized several assumptions of the current approach. Future work will aim to address those limitations. The goal is to enable route teaching through a human pilot (and not only on the cart) and to allow arbitrary flight paths (and not straight lines at a constant altitude). To do so, the following steps are necessary:

- accounting for roll and pitch when assigning the depth to each pixel;
- extending the controller to be able to track arbitrary paths and to include a vertical controller;
- deriving a feasible desired velocity profile (instead of a constant velocity), which takes vehicle constraints and constraints derived from the VT&R localization algorithm into account;
- increasing the image processing speed, which would enable higher flight speeds;
- extensive testing under various conditions (including lighting changes).

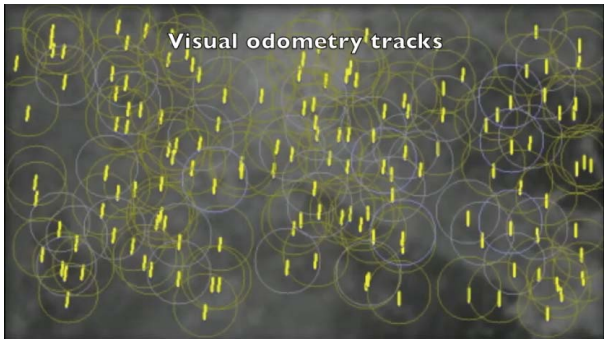
VII. Conclusion

We successfully showed that the visual teach and repeat (VT&R) algorithm in [6], which has previously been used for autonomous, long-range navigation of ground vehicles, can be applied to flying vehicles with the 3D sensor being replaced by a monocular, downward-facing camera and an altitude sensor. The use of a manifold map of overlapping submaps allows long-range navigation along a previously explored path without the use of GPS or a globally consistent

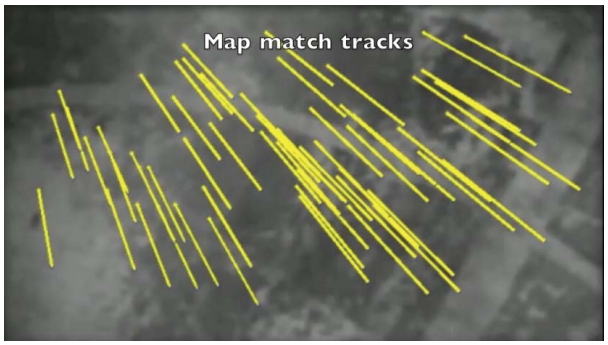
(a) External video.



(b) Path and quadrocopter visualization.



(c) Visual odometry tracks.



(d) Map match tracks.

Figure 7. Snapshots of the 16-m-long autonomous flight, which is shown in the movie found at http://youtu.be/BRDvK4xD8ZY.

map. A series of preliminary experiments were conducted and showed accurate localization based on the VT&R algorithm and successful vision-based autonomous flight along a straight, level path with the designed path-tracking controller. The path was taught with the quadrocopter fixed to a cart to guarantee a straight path at constant altitude. We demonstrated the capability of the overall system by autonomously repeating a longer and more complex route at 1.2 m/s, see Fig. 7 and the associated video.

ACKNOWLEDGMENT

The authors would like to thank Prof. Roland Siegwart from the Autonomous Systems Lab (ASL) for supervising the original work of this paper on behalf of ETH Zurich. We also extend our thanks to Valentin Peretroukhin, Tristan Laidlow and Sandro Camichel for fruitful discussion and for helping with the video. This work was supported by the Natural Sciences and Engineering Research Council (NSERC) through the NSERC Canadian Field Robotics Network (NCFRN).

REFERENCES

[1] A. Howard, "Multi-robot mapping using manifold representations," in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, vol. 4, 2004, pp. 4198–4203.

[2] P. T. Furgale and T. D. Barfoot, "Visual teach and repeat for long-range rover autonomy," *Journal of Field Robotics*, vol. 27, no. 5, pp. 534–560, 2010.

[3] Parrot, "AR.Drone 2.0," accessed June 17, 2013. [Online]. Available: http://ardrone2.parrot.com

[4] N. Michael, D. Mellinger, Q. Lindsey, and V. Kumar, "The GRASP multiple micro-UAV testbed," *IEEE Robotics Automation Magazine*, vol. 17, no. 3, pp. 56–65, 2010.

[5] S. Lupashin, M. Hehn, M. W. Mueller, A. P. Schoellig, M. Sherback, and R. D'Andrea, "A platform for aerial robotics research and demonstration: The Flying Machine Arena," *Mechatronics*, vol. 24, no. 1, pp. 41–54, 2014.

[6] C. McManus, P. Furgale, B. Stenning, and T. Barfoot, "Visual teach and repeat using appearance-based lidar," in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, 2012, pp. 389–396.

[7] S. Grzonka, G. Grisetti, and W. Burgard, "Towards a navigation system for autonomous indoor flying," in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, 2009, pp. 2878–2883.

[8] A. S. Huang, A. Bachrach, P. Henry, M. Krainin, D. Maturana, D. Fox, and N. Roy, "Visual odometry and mapping for autonomous flight using an RGB-D camera," in *Proceedings of the IEEE International Symposium of Robotics Research (ISRR)*, 2011.

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

[10] M. Blösch, S. Weiss, D. Scaramuzza, and R. Siegwart, "Vision based MAV navigation in unknown and unstructured environments," in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, 2010, pp. 21–28.

[11] M. Achtelik, S. Weiss, and R. Siegwart, "Onboard IMU and monocular vision based control for MAVs in unknown in- and outdoor environments," in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, 2011, pp. 3056–3063.

[12] J. Engel, J. Sturm, and D. Cremers, "Accurate figure flying with a quadrocopter using onboard visual and inertial sensing," in *Proceedings of the Workshop on Visual Control of Mobile Robots (ViCoMoR) at the IEEE/RJS International Conference on Intelligent Robot Systems (IROS)*, 2012.

[13] A. Visser, N. Dijkshoorn, M. van der Veen, and R. Jurriaans, "Closing the gap between simulation and reality in the sensor and motion models of an autonomous AR.Drone," in *Proceedings of the International Micro Air Vehicle Conference and Flight Competition, IMAV*, 2011.

[14] K. Celik, S.-J. Chung, M. Clausman, and A. Somani, "Monocular vision SLAM for indoor aerial vehicles," in *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2009, pp. 1566–1573.

[15] Z. Huijuan and H. Qiong, "Fast image matching based-on improved SURF algorithm," in *Proceedings of the International Conference on Electronics, Communications and Control (ICECC)*, 2011, pp. 1460–1463.

[16] C. Bills, J. Chen, and A. Saxena, "Autonomous MAV flight in indoor environments using single image perspective cues," in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, 2011, pp. 5776–5783.

[17] T. Krajník, V. Vonásek, D. Fišer, and J. Faigl, "AR-Drone as a platform for robotic research and education," in *Research and Education in Robotics - EUROBOT*, ser. Communications in Computer and Information Science. Springer Berlin Heidelberg, 2011, vol. 161, pp. 172–186.

[18] T. Krajník, J. Faigl, V. Vonásek, K. Košnar, M. Kulich, and L. Přeučil, "Simple yet stable bearing-only navigation," *Journal of Field Robotics*, vol. 27, no. 5, pp. 511–533, 2010.

[19] H. Bay, A. Ess, T. Tuytelaars, and L. V. Gool, "Speeded-up robust features (SURF)," *Computer Vision and Image Understanding*, vol. 110, no. 3, pp. 346–359, 2008.

[20] P. H. Zipfel, *Modeling and simulation of aerospace vehicle dynamics*, 2nd ed., 2007.

[21] Parrot, "AR.Drone 2.0 Specifications," accessed May 27, 2013. [Online]. Available: http://ardrone2.parrot.com/ar-drone-2/specifications

[22] M. Quigley, K. Conley, B. Gerkey, J. Faust, T. Foote, J. Leibs, R. Wheeler, and A. Y. Ng, "ROS: an open-source robot operating system," in *Proceedings of the Workshop on Open Source Software at the the IEEE International Conference on Robotics and Automation (ICRA)*, vol. 3, no. 3.2, 2009.

[23] M. Monajjemi, "ardrone_autonomy," accessed June 18, 2013. [Online]. Available: https://github.com/AutonomyLab/ardrone_autonomy

[24] J. Bowman, "camera_calibration package," accessed June 18, 2013. [Online]. Available: http://www.ros.org/wiki/camera_calibration

[25] J. L. Patrick Mihelich, Kurt Konolige, "image_proc package," accessed June 18, 2013. [Online]. Available: http://www.ros.org/wiki/image_proc