2013 IEEE/RSJ International Conference on
Intelligent Robots and Systems (IROS)
November 3-7, 2013. Tokyo, Japan

FollowMe: Person Following and Gesture Recognition with a Quadrocopter

Tayyab Naseer*, Jürgen Sturm†, and Daniel Cremers†

*Department of Computer Science, University of Freiburg, Germany
†Department of Computer Science, Technical University of Munich, Germany

***Abstract*— In this paper, we present an approach that allows a quadrocopter to follow a person and to recognize simple gestures using an onboard depth camera. This enables novel applications such as hands-free filming and picture taking. The problem of tracking a person with an onboard camera however is highly challenging due to the self-motion of the platform. To overcome this problem, we stabilize the depth image by warping it to a virtual-static camera, using the estimated pose of the quadrocopter obtained from vision and inertial sensors using an Extended Kalman filter. We show that such a stabilized depth video is well suited to use with existing person trackers such as the OpenNI tracker. Using this approach, the quadrocopter not only obtains the position and orientation of the tracked person, but also the full body pose – which can then for example be used to recognize hand gestures to control the quadrocopter's behaviour. We implemented a small set of example commands ("follow me", "take picture", "land"), and generate corresponding motion commands. We demonstrate the practical performance of our approach in an extensive set of experiments with a quadrocopter. Although our current system is limited to indoor environments and small motions due to the restrictions of the used depth sensor, it indicates that there is large potential for such applications in the near future.**

I. Introduction

Imagine you had a flying camera that was able to follow you and recognize simple commands given by hand gestures: Such a device would be very useful, for example take a video sport activities such as climbing or skiing – easily allowing videos and pictures to be taken from a whole new perspective without the need for a full-scale helicopter or a pilot who remotely controls a small-scale UAV. The ability to recognize a set of control commands in the form of hand gestures completely eliminates the need for a remote control even to transmit high-level control commands such as "land" or "take a picture now". While people following by itself is a useful application, quadrocopters are also envisioned to serve as flexible helpers for autonomous mapping, in search and rescue tasks, and even for the transportation of small goods. In all of these tasks, a natural user interface that allows a person to interact with the quadrocopter is highly relevant.

Although today's quadrocopter systems have several limitations with respect to their payload, computational capabilities, battery time, and safety, it is clear that with progressing miniaturization the realization of such devices will come soon within reach. A pioneering example is the Joggobot [11] that is able to autonomously follow an athlete wearing a special t-shirt equipped with optical markers. The goal of this paper is to eliminate this requirement and to enable a similar application without the use of artificial markers. Therefore, we investigate in this paper how to detect a person and track the full body pose from a moving camera, how to extract simple gesture commands from this pose, and to make the quadrocopter react accordingly.

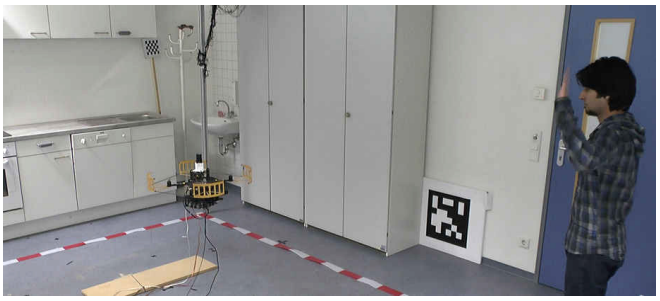


Fig. 1. Our approach enables a quadrocopter to follow a person and to understand simple gesture commands such as "follow me", "take picture", and "land".

While people detection and tracking has been studied intensively in computer vision and robotics, many existing solutions [14, 2, 30] assume a static sensor which simplifies background subtraction and tracking significantly. A large body of work exists on pedestrian detection for driving assistance systems and autonomous cars [8, 24], where often only the position and the movement direction of the persons is estimated. In contrast, our application requires people detection and tracking from a freely moving flying platform. Furthermore, we need the full body pose for gesture recognition. To the best of our knowledge, we are the first to tackle this problem for the application on a quadrocopter.

In our current system, we use an AscTec Pelican quadrocopter that is equipped with an Asus Xtion Pro Live depth camera. The core idea behind our approach is to stabilize the depth image in software by simulating one or more static cameras using image warping. We found that the warped depth images from our simulated camera are steady enough to be used with existing pose trackers such as the one from OpenNI [25]. The camera pose estimation runs on board. Although we currently run all computationally intensive tasks offboard on a ground station, as only a CPU is needed, it is feasible and straight-forward to run the same software onboard. In each time step, we obtain from the tracker the full body posture of the tracked person. This allows us to follow this person at a certain distance, and to recognize simple

978-1-4673-6357-0/13/$31.00 ©2013 IEEE

hand gestures such as a raised left or right arm. We map these gestures to commands such as "follow me", "hold position", "take picture", and "land". Overall, our system demonstrates that person following and gesture recognition is feasible with a quadrocopter. A video illustrating our approach is included in the supplemental material.

II. Related Work

People detection and tracking has a long history in computer vision and robotics (see [21, 26] for two recent surveys). Various approaches have been proposed to detect persons on camera images [16, 4]. Popular techniques include the sliding window approach, boosting [31], histograms of gradients [6], and chamfer matching. It has been shown that a higher detection accuracy can be achieved when additionally motion is considered [32]. Excellent prior work also exists for people detection from a moving vehicle [9, 8, 7], which is relevant for driving assistance systems and the autonomous car of the future. After a person has been detected, the next problem is to estimate his body posture [10, 29, 1, 15]. This can be either estimated top-down, for example, using a particle filter and an appearance model, or bottom-up, by using a part detector or pixel-wise classifier.

Next to camera images, also range data from laser scanners has been used in particular by the robotics community, both in 2D [2, 28] and 3D [24, 3]. Low-cost depth cameras such as the Microsoft Kinect stimulated the development of novel approaches to people detection and tracking on depth images [14, 30, 19, 35]. Shotton et al. [14] learn deep decision trees to detect and track up to 20 body parts of the human body. By combining color and depth information, the detection performance can further be improved [30, 19]. However, most approaches still assume a static camera, which simplifies image segmentation. This is in particular true for both the Microsoft Kinect tracker[1] and the OpenNI tracker[2]. In preliminary experiments, we found that both trackers fail consistently when used with a moving camera, that is they detect a person where there is none, or fail to detect a person present in the image. Some examples are shown in Fig. 2 and Fig. 3.

Early work on people following with quadrocopters therefore relies on visual markers. Gräther and Müller recently presented a jogging companion that motivates a person during jogging [11]. Another interesting application is to use a quadrocopter to provide external visual imagery, which allows for example athletes to gain a better understanding of their motions [13]. Lichtenstern et al. [17] presented a system where they control a team of flying robots using gestures, captured by a hovering quadrocopter with a depth camera. This is different from our work as the quadrocopter is only hovering, so no substantial movement of the camera is expected. Another recent approach recognizes gestures by using optical flow to cancel the egomotion [22]. Using this approach, they track the face and the hands but do not obtain the full body pose.

[1] http://microsoft.com/en-us/kinectforwindows/develop/overview

[2] http://openni.org

Fig. 2. OpenNI user segmentaion fails for non-static cameras. When a person is detected, it is assigned a particular color whereas non-colored depth map implies no person detection.

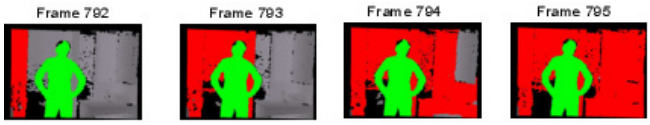


Fig. 3. False positives with non-stationary camera. It shows that the OpenNI tracker detects the background as a human.

The key idea behind our work is to stabilize the depth image as a pre-processing step so that existing trackers which assume a static camera become directly applicable – in the presence of translational camera movement, this however is only possible with the availability of depth-information [23, 12, 20]. Full 3D video stabilization methods require an intermediate 3D reconstruction of the scene to allow for the simulation of novel viewpoint [5]. By using a depth camera such as the Microsoft Kinect, one already obtains a suitable depth image that can be used for motion estimation and novel view synthesis [18]. A similar approach was recently described to stabilize the stereo images of a walking humanoid robot [27].

In this work, we enable a quadrocopter to detect and track the user with an onboard depth camera. In contrast to previous work, we do not require artificial markers on the person. Furthermore, we implemented a gesture recognition that allows the user to give simple commands.

III. Approach

Our approach consists of five main components, as illustrated in Fig. 4. We employ an AscTec Pelican quadrocopter that we equipped with an Asus Xtion Pro Live sensor for people detection and a PointGrey FireFly camera for localization. The PointGrey FireFly camera is used to track the quadrocopter's movement using ceiling markers, based on the ARToolkitPlus library [33]. We employ an Extended Kalman filter to optimally fuse these visual pose estimates with inertial measurements from an onboard IMU. Based on the estimated pose estimates, the depth image from the Asus sensor is stabilized, i.e. warped into a virtual, static camera, and fed into the OpenNI tracker. We then use the full body pose to generate the next waypoint and to recognize a small set of gestures, and a LQR controller to steer the quadrocopter to the desired location, using the open-source implementation of ETH [34].

A. Quadrocopter localization

As localization is not the focus of this paper, we employ marker-based localization using the ArtoolkitPlus library. We

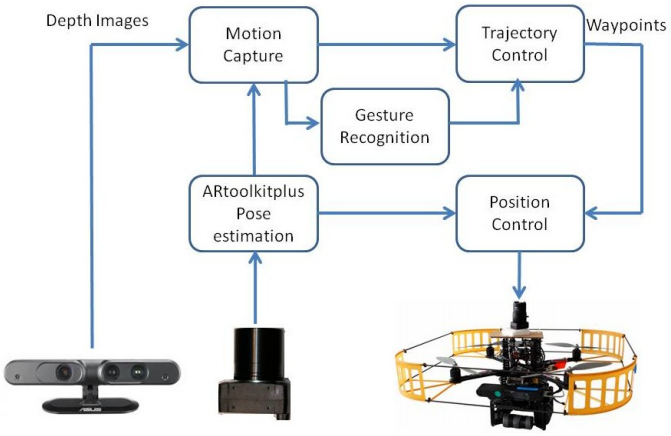


Fig. 4. System overview. We use a depth camera on a flying platform to detect a person. We stabilize the depth image based on the current pose estimate of the quadrocopter.

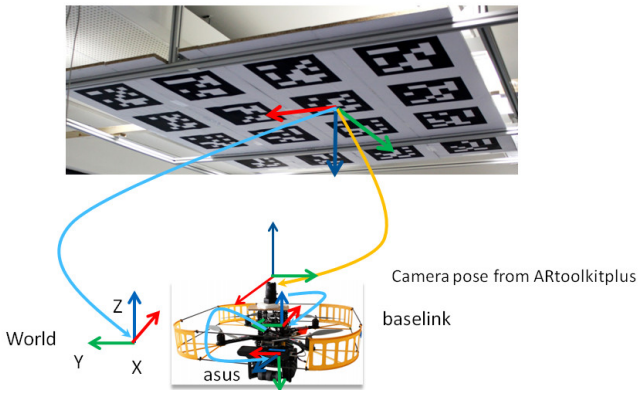


Fig. 5. Coordinate frames used in our approach. Static transformations are shown in blue whereas camera pose is estimated in realtime and is shown in yellow.

use the multi-marker mode, which provides us with accurate and robust pose estimates and allows us to cover a large part of our lab. In particular, we mounted a matrix of 4×4 markers on the ceiling over an area of 3×3m. Mounting the markers on the ceiling instead of on the floor has the advantage that we can perform autonomous take-off and landing behaviors. Each marker has a side length of 45cm. At the typical flight height of 0.6m, we found that the quadrocopter observes on average 4 markers at the same time. Our setup consisting of the quadrocopter, the ceiling markers, and the relevant coordinate systems are illustrated in Fig. 5.

We defined the positions and orientations of all markers in a map, and the ArtoolkitPlus library provides us in each time step with the camera pose with respect to this map. In the remainder of this work, we represent a 3D pose in terms of its transformation matrix, i.e.,

$$T = \begin{pmatrix} R & \mathbf{t} \\ \mathbf{0} & 1 \end{pmatrix} \in \mathbb{R}^{4\times 4}, \tag{1}$$

where $R \in \mathbb{R}^{3\times 3}$ is the rotation matrix and $\mathbf{t} \in \mathbb{R}^3$ is the translation vector. In each time step, we obtain the pose of the markers relative to the camera, which we denote by T^{marker}_{camera}. Furthermore, we defined the world coordinate system that we located in the middle of our quadrocopter lab on the floor, and determined manually the transformation T^{marker}_{world} between this coordinate system and the marker map, and the transformation $T^{baselink}_{camera}$ from the camera to the base link of the robot. As the monocular camera is pointing upwards to the ceiling while the Asus sensor is pointing horizontally forward towards the user, we constructed a special calibration stick that we could hold in front of both cameras at the same time. The size of the calibration stick was chosen so that we could first determine the relative pose $T^{checkerboard2}_{checkerboard1}$ between both checkerboards by holding the stick in front of a single camera, and then use this calibration to estimate the transformation between the two cameras T^{asus}_{camera}.

Transformation matrices can be easily concatenated and inverted. In the following, we will briefly introduce the notation that we use throughout the remainder of this paper: Given a 3D point $\mathbf{p}_w = (x_w, y_w, z_w)^\top \in \mathbb{R}^3$ in world coordinates, we can compute the corresponding position in camera coordinates $\mathbf{p}_c = (x_c, y_c, z_c)^\top \in \mathbb{R}^3$ as

$$\begin{pmatrix} x_c \\ y_c \\ z_c \\ 1 \end{pmatrix} = \begin{pmatrix} R & \mathbf{t} \\ \mathbf{0} & 1 \end{pmatrix} \begin{pmatrix} x_w \\ y_w \\ z_w \\ 1 \end{pmatrix} \tag{2}$$

where R and $\mathbf{t}$ are the rotation and translation of the corresponding transformation T^{world}_{camera}.

We assume a pinhole camera model, that defines the relationship between a 3D point $\mathbf{p} = (x, y, z)^\top \in \mathbb{R}^3$ and a 2D pixel $\mathbf{x} = (i, j)^\top \in \mathbb{R}^2$ as follows,

$$\pi(x, y, z) = \left(\frac{f_x x}{z} + c_x, \frac{f_y y}{z} + c_y \right)^\top = (i, j)^\top. \tag{3}$$

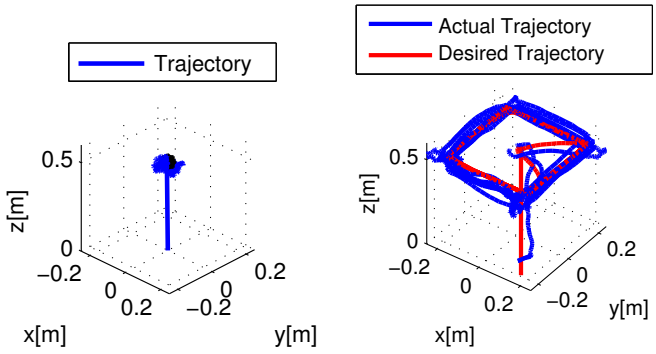
Here, f_x, f_y, c_x, c_y refer to the focal length and the optical center of the camera, respectively. In reverse, given the depth z of a pixel (i, j), we can reconstruct the 3D point using

$$\rho(i, j, z) = \left(\frac{(i - c_x)z}{f_x}, \frac{(j - c_y)z}{f_y}, z \right)^\top = (x, y, z)^\top. \tag{4}$$

B. Position control

The quadrocopter estimates its current pose from visual pose estimates and IMU readings using an extended Kalman filter [34]. This state estimate is then used for position control.

In a set of preliminary experiments, we found that we can achieve accurate and stable position control. In a first experiment, we sent a hovering command to the quadrocopter at point $\mathbf{p}_{goal} = (0, 0, 0.5)$. We measured an average root mean square error (RMSE) around the given set point of 2.78cm along the x-axis, 2.22cm along the y-axis, and 0.73cm along the z-axis. The maximum run-away was 7.4cm along y-axis over a period of 5 minutes. In a second experiment, we sent a simple trajectory in form of a rectangle with 40cm side length to the quadrocopter. The results are visualized in Fig. 6. As can be seen from this plot, we achieve accurate



(a) Quadrocopter hovering (b) Square Trajectory

Fig. 6. Position control of the quadrocopter

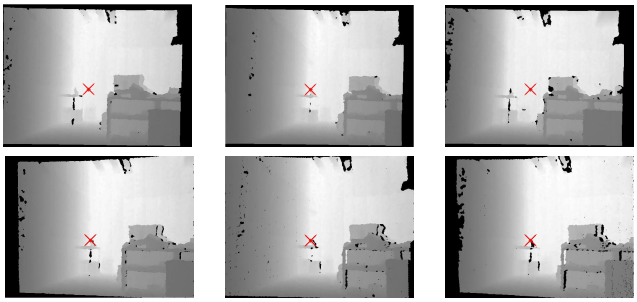


Fig. 7. Visualization of the raw and the stabilized depth images with camera motion along a square trajectory (top and bottom row respectively). The red dot always marks the same (absolute) pixel in the image. As can be seen, the red pixel jumps significantly in the raw images, while it remains stable in the stabilized ones.

position control with this setup.

C. Camera stabilization

As we could confirm in our preliminary experiments, the OpenNI tracker only performs well with a stationary camera (see Fig. 2 and 3), which is due to the fact that internally the tracker performs background subtraction to segment the image into moving foreground (persons) and static background. The OpenNI tracker has an initialization time of a few seconds in which it learns the background model, which is then used for pixel-wise, binary classification. On a flying platform however, the constraint of a stationary background is clearly violated and this approach therefore fails.

We therefore propose to simulate a virtual static camera based on the pose estimate of the quadrocopter. To this aim, we memorize the pose T_{world}^{static} of a reference frame that we use as the pose of the static-virtual camera. As reference frame, we choose the current camera pose at time step t and keep it as long as person tracking is successful. In principle, it would be sufficient to initialize a new static camera only after tracking has been lost, as long as enough previous frames are kept in a buffer to allow immediate (re-)initialization of the people tracker. Alternatively, several static cameras can be maintained at the same time and used for redundant tracking. In our approach, we do not deal with the problem of re-initialization, but assume that the person is staying within the frustrum of the first static camera.

To create a virtual image in a static camera frame, we need to render the depth image from the perspective of a fixed, alternative viewpoint – as the OpenNI tracker solely works on depth information, only the synthesized depth image is required. This is done by first un-projecting the pixels of the current camera image to reconstruct the respective 3D point, which is then transformed into the coordinate frame of the virtual, static camera and re-projected onto the image plane. Consider a depth image I_k taken by a current camera at time k with pose T_{world}^{k}. The reconstructed 3D point corresponding to a pixel (i,j) with depth $I_k(i,j)$ is given by

$$\mathbf{p}_k(i,j) = \rho(i,j,I_k(i,j)). \tag{5}$$

Given the transformation between the current and the static camera

$$T_{static}^{k} = (T_{world}^{static})^{-1} T_{world}^{k}, \tag{6}$$

we can transform the 3D point into the frame of the static camera as follows,

$$\mathbf{p}_{static}(i,j) = T_{static}^{k} \mathbf{p}_k(i,j). \tag{7}$$

We project this point onto the image plane of the static camera, i.e.,

$$(i',j')^\top = \pi(\mathbf{p}_{static}(i,j)), \tag{8}$$

and assign the computed depth at the corresponding location in the warped depth image,

$$I_{static}(\lfloor i'+0.5 \rfloor, \lfloor j'+0.5 \rfloor) \leftarrow z_{static}. \tag{9}$$

When two points are projected onto the same pixel we assume the further-away one to be occluded and remove it. The above procedure is applied to all image pixels, i.e. (5) to (9), to obtain a warped depth image I_{static} in the frame of the virtual, static camera.

Forward warping, as described above, is known to produce holes in the warped image, due to the discrete sampling of pixels. To remedy this, we apply median filtering over a small 5×5 patch to close holes and regularize the result, i.e.,

$$I_{static}(i,j) \leftarrow \underset{(k,l)\in S}{\text{median}} I_{static}(i+k, j+l). \tag{10}$$

Note that this procedure requires only very few additions and multiplications for each pixel, and therefore can be computed very efficiently. Fig. 7 shows the result of warping when the camera was following a square trajectory. A particular point has been marked and labeled across the frames for reference. Each frame shown here has been captured after four seconds to allow significant camera motion.

D. Cascaded control

Figure 8 shows the flow diagram of our cascaded control architecture used in our approach. The low level processor (LLP) provides the attitude control for the platform. The high level processor (HLP) runs the extended Kalman filter and position/velocity control at 1KHz, and accepts velocity commands and waypoints. The error controller is a LQR

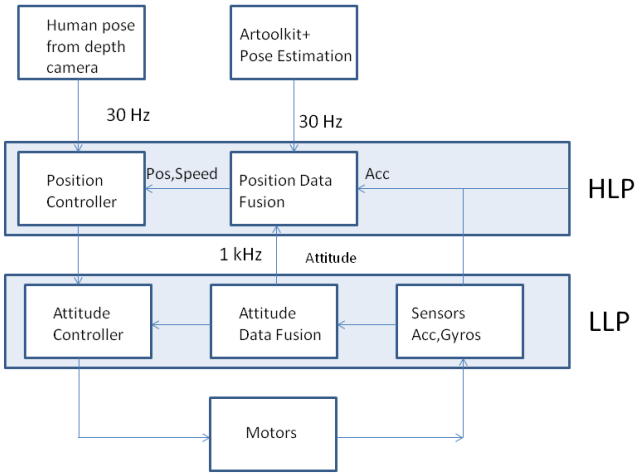


Fig. 8. Scheme of our cascaded control architecture used in our approach.

controller for each axis. The feed forward model allows the quadrocopter to quickly reach the waypoint and reduces overshoot [34]. It is possible to specify the approach speed and accuracy with which the quadrocopter should reach the goal location. The onboard PC running ROS supplies the HLP with the visual pose estimates and velocity commands/waypoints at 30Hz.

E. Robust person following

As explained above, we first stabilize the incoming depth stream using our approach with a static-virtual camera. Subsequently, we feed the stabilized depth stream into the OpenNI tracker to obtain the 3D joint positions $\mathbf{p}^{joint}_{static}$ of all 15 body parts, where $joint \in \{torso, lefthand, \ldots\}$. Note that we obtain the resulting joint positions in the frame of the static-virtual camera, from where we transform them into the world frame for following the person. For example, we obtain the joint position of the torso in the world frame as:

$$\mathbf{p}^{torso}_{world} = T^{static}_{world}\mathbf{p}^{torso}_{static} \tag{11}$$

When the quadrocopter is in person following mode, we aim to keep the relative pose. We save the initial relative pose when the person following mode is activated, i.e.,

$$\Delta_{desired} = \mathbf{p}^{torso}_{world} - \mathbf{p}^{baselink}_{world} \tag{12}$$

where *baselink* refers to the center of the quadrocopter. From then on, we continuously generate waypoints such that this relative pose is maintained, i.e., we compute

$$\mathbf{p}^{waypoint}_{world} := \mathbf{p}^{torso}_{world} - \Delta_{desired} \tag{13}$$

that we send to the position controller. We propose the following algorithm 1 for robust person tracking and following.

F. Gestured controlled flight

One benefit of our approach is that we obtain the full body pose of the person in the field of view of the camera. This allows us to specify simple gesture commands that the

function PERSON TRACKING FROM MOVING CAMERA
  input: Current camera pose T_1
  Initialize virtual static camera pose at $T_{static} := T_1$
  for each new depth image I_k and camera pose T_k
    Calculate transform T^k_{static} between current and static camera
    Generate depth image I_{static} according to (5)–(9)
    Feed stabilized depth image to OpenNI tracker
    if gesture recognized
      Take picture or land
    else
      Update waypoint and follow person
    end if
end function

Algorithm 1: Robust person tracking on stabilized images.

quadrocopter can recognize. In particular, we implemented gestures such as raising the left or right arm. When both arms are low, the quadrocopter follows the person as described in the previous section. To detect an arm gesture, we check for a certain vertical distance between the hand and the corresponding hip joint. For example, to recognize whether the right hand is raised, we check whether

$$z^{righthand}_{world} - z^{righthip}_{world} \geq 50cm. \tag{14}$$

If this condition is met, the quadrocopter saves an image of the person to disk (see Fig. 11 for a few snapshots). When the user raises the left hand, we land the quadrocopter at its current position by sending a motion command with a constant negative velocity to the HLP.

IV. RESULTS

In this section, we present an evaluation of our approach. In particular, the goal of our experiments is to demonstrate that (1) our approach to video stabilization using the visual pose estimate is valid, (2) person can be detected and the full body pose can be tracked, (3) various gestures can be recognized reliably, and (4) the overall system enables person following with the quadrocopter and, also, allows the user to give simple commands such as “take picture”, and “land”.

A. Video stabilization

Figure 7 shows the raw video and the stabilized video from the quadrocopter while it was following a rectangular trajectory (as shown in Fig. 6). As can be seen, the changes in the pose of the quadrocopter lead to significant changes of the depth image, while the stabilized depth image remains static. The larger the change in viewpoint however, the sparser the synthesized depth image becomes due to self-occlusions and a changing camera frustrum. As a result, in principle the quadrocopter needs to re-initialize the static camera after substantial motion.

B. Person detection and pose recognition

We feed the stabilized depth image to the OpenNI tracker. Figure 9 shows an example of full body pose tracking from a moving camera. We found that the OpenNI tracker usually requires up to 80 frames to initialize the background model, and another 40 frames to initialize person tracking.

Therefore, we plan to implement a ring buffer of 120 frames to ensure that the static camera can be seamlessly re-initialized at a different location without loosing track of the followed person. To evaluate the performance of the tracker

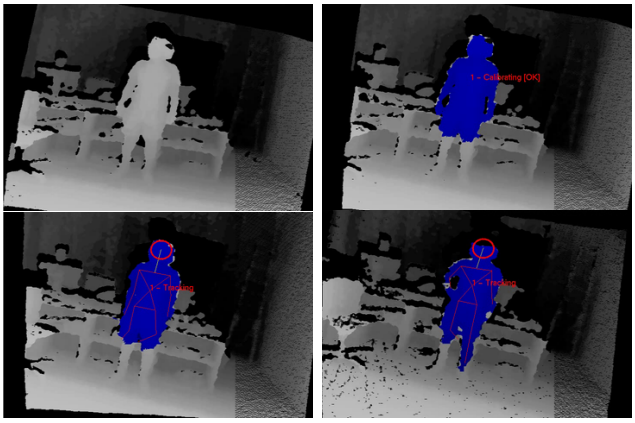


Fig. 9. Illustration of robust person tracking with stabilized depth video. The OpenNI tracker successfully initializes tracking, although the real depth camera undergoes a strong motion.

on a moving platform, we commanded the quadrocopter to keep position and to fly on a rectangular path. We asked different persons to stand at a distances of 2m in front of the quadrocopter, and evaluated the tracking performance on raw video and stabilized video. Table I gives the result over a total of 1000 frames: As can be seen, the average performance with stabilized depth video is substantially higher than without stabilization.

TABLE I

TRACKING PERFORMANCE WITH AND WITHOUT VIDEO STABILIZATION FOR DIFFERENT MOTIONS OF THE QUADROCOPTER. THE PERCENTAGES GIVE THE RATIO OF FRAMES WITH SUCCESSFUL POSE TRACKING.

flight path	depth video	performance
hovering	raw video	30%
	stabilized video	95%
rectangular	raw video	16%
path	stabilized video	86%

C. Person following

We evaluated our approach of person following in a scenario where the person moved along the x-axis i.e forward

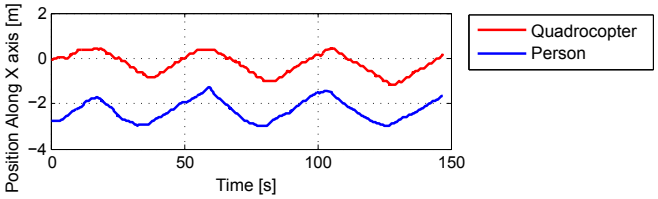


Fig. 10. Robust person following over a timespan of three minutes.

and backwards. Figure 10 shows the results, which clearly show that quadrocopter accurately followed the person's movement throughout the whole flight.

TABLE II

EVALUATION OF THE GESTURE RECOGNITION PERFORMANCE OVER FIVE DIFFERENT SUBJECTS. THE OVERALL PERFORMANCE WAS 92.5%.

test subject	raise left hand	raise right hand
person 1	100%	90%
person 2	90%	100%
person 3	100%	90%
person 4	80%	100%
person 5	90%	80%
overall performance	92.5%	92.5%

D. Gesture recognition

Based on the full body pose obtained from the stabilized depth video, we recognize various arm gestures such as "raise left hand" and "raise right hand". To evaluate the performance of recognition, we asked 5 different subjects to perform these gestures while standing in front of the hovering quadrocopter. Each subject was asked to repeat each gesture 10 times. The results are given in Tab. II. While the recognition rates differ per person, the overall recognition rate of 92.5% indicates that our gesture recognition system allows a user to give a small set of commands to a flying quadrocopter without requiring a remote control.

E. Full system

To evaluate the overall performance of the overall system, we asked three subjects to perform a simple routine with the quadrocopter. As soon as the quadrocopter detects a person, it switches to person following mode. Then the person walks a small distance (around 1m) into any desired direction. The quadrocopter copies this behaviour. Subsequently, the person raises his right hand to trigger a picture on the quadrocopter. By raising the left hand, the quadrocopter lands. A subset of the images taken by the quadrocopter during the right hand gesture are shown in Fig. 11.

Our experiments clearly show that depth video stabilization is a suitable approach to enable person detection and tracking with a flying quadrocopter. Furthermore, the full body pose allows us to recognize simple gestures, which enables a user to give various control commands to a quadrocopter. Finally, the demonstration of the full system indicates that such applications bear a large potential for the future. However, we believe that significant progress in miniaturization of all components is required before such a solution becomes attractive for the consumer.

V. CONCLUSION

In this paper, we demonstrated that full-pose person tracking and simple gesture recognition is feasible using an autonomous quadrocopter equipped with an on-board depth camera. This is achieved by stabilizing the camera images, warping the depth image according to the estimated pose of

Fig. 11. Recognition of the right hand gesture triggers the caption of an image.

the quadrocopter. As a result, existing person trackers such as the OpenNI framework, which in their original form require the camera to be static, become applicable and provide the full body pose. This allowed us to implement gesture control including to follow a person, taking a picture, and landing when given a hand sign. While our current approach is limited to indoor environments, and relies on an external ground station, we plan to extend it for outdoor use on a fully autonomous quadrocopter that solely relies on monocular or stereo cameras. We believe that using quadrocopters as flying companions bears a large potential for practical application in the future.

REFERENCES

[1] A. Agarwal and B. Triggs. Recovering 3D human pose from monocular images. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 28(1), 2006.

[2] K. Arras, O. Martinez, and W. Burgard. Using boosted features for detection of people in 2D range scans. In *Proc. of the IEEE Intl. Conf. on Robotics and Automation (ICRA)*, 2007.

[3] M. Bajracharya, B. Moghaddam, A. Howard, S. Brennan, and L. Matthies. A fast stereo-based system for detecting and tracking pedestrians from a moving vehicle. *Int. J. Rob. Res.*, 28(11-12):1466–1485, November 2009.

[4] M. Breitenstein, F. Reichlin, B. Leibe, E. Koller-Meier, and L. Van Gool. Robust tracking-by-detection using a detector confidence particle filter. In *IEEE International Conference on Computer Vision (ICCV)*, pages 1515–1522. IEEE, 2009.

[5] C. Buehler, B. Michael, and M. Leonard. Non-metric image-based rendering for video stabilization. In *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2001.

[6] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In *Proc. of the IEEE Int. Conf. on Computer Vision and Pattern Recognition (CVPR)*, volume 1, pages 886–893, 2005.

[7] P. Dollr, C. Wojek, B. Schiele, and P. Perona. Pedestrian detection: A benchmark. In *Proc. of the IEEE Int. Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2009.

[8] A. Ess, B. Leibe, K. Schindler, and L. Van Gool. A mobile vision system for robust multi-person tracking. In *Proc. of the IEEE Int. Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2008.

[9] D.M. Gavrila. Pedestrian detection from a moving vehicle. In *European Conference on Computer Vision (ECCV)*, 2000.

[10] D.M. Gavrila and L.S. Davis. 3d model-based tracking of humans in action: a multi-view approach. In *Proc. of the IEEE Computer Vision and Pattern Recognition Conference*, 1996.

[11] E. Graether and F. Mueller. Joggobot: a flying robot as jogging companion. In *Extended Abstracts on Human Factors in Computing Systems, CHI '12*, pages 1063–1066, New York, NY, USA. ACM.

[12] M. Grundmann, V. Kwatra, and I. Essa. Auto-directed video stabilization with robust l1 optimal camera paths. In *Proc. of the IEEE Int. Conf. on Computer Vision and Pattern Recognition (CVPR)*, pages 225–232. IEEE, June 2011.

[13] K. Higuchi, T. Shimada, and J. Rekimoto. Flying sports assistant: external visual imagery representation for sports training. In *Proc. of the 2nd Augmented Human International Conference*, pages 7:1–7:4, New York, NY, USA, 2011. ACM.

[14] J.Shotton, A.Fitzgibbon, M.Cook, T.Sharp, M.Finocchio, R.Moore, A.Kipman, and Andrew Blake. Real-time human pose recognition in parts from single depth images. In *Proc. of the IEEE Int. Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2011.

[15] R. Kehl and L. Van Gool. Markerless tracking of complex human motions from multiple views. *Computer Vision and Image Understanding*, 104(2), November 2006.

[16] B. Leibe, E. Seemann, and B. Schiele. Pedestrian detection in crowded scenes. In *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 878–885, Washington, DC, USA, 2005. IEEE Computer Society.

[17] Michael Lichtenstern, Martin Frassl, Bernhard Perun, and Michael Angermann. A prototyping environment for interaction between a human and a robotic multi-agent system. In *Proc. of the 7th ACM/IEEE International Conference on Human-Robot Interaction (HRI 2012)*, Boston, Massachusetts, USA, March 2012.

[18] S. Liu, Y. Wang, L. Yuan, J. Bu, P. Tan, and J. Sun. Video stabilization with a depth camera. In *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 89–95, 2012.

[19] M. Luber, L. Spinello, and K. Arras. People tracking in RGB-D data with on-line boosted target models. In *IEEE Int. Conf. on Intelligent Robots and Systems (IROS)*, pages 3844–3849. IEEE, 2011.

[20] Y. Matsushita, E. Ofek, W. Ge, X. Tang, and H. Shum. Full-frame video stabilization with motion inpainting. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 28(7):1150–1163, July 2006.

[21] T. Moeslund, A. Hilton, and V. Krüger. A survey of advances in vision-based human motion capture and analysis. *Comput. Vis. Image Underst.*, 104(2), 2006.

[22] Valiallah Mani Monajjemi, Jens Wawerla, Richard T. Vaughan, and Greg Mori. Hri in the sky: Creating and commanding teams of uavs with a vision-mediated gestural interface. In *Proc. of the IEEE Int. Conf. on Intelligent Robots and Systems (IROS'13)*, 2013.

[23] C. Morimoto and R. Chellappa. Evaluation of image stabilization algorithms. In *Proc. of the IEEE Int. Conf. on on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 1998.

[24] L. Navarro-Serment, C. Mertz, and M. Hebert. Pedestrian detection and tracking using three-dimensional LADAR data. *I. J. Robotic Res.*, 29(12), October 2010.

[25] OpenNI. http://openni.org. Last accessed on March 14, 2013.

[26] R. Poppe. Vision-based human motion analysis: An overview. *Comput. Vis. Image Underst.*, 108(1–2), 2007.

[27] Y. Ryu, H. Roh, M. Chung, J. Heo, and J. Oh. 3d video stabilization for a humanoid robot using point feature trajectory smoothing. In *Humanoids*, pages 81–86. IEEE, 2011.

[28] D. Schulz, W. Burgard, D. Fox, and A. Cremers. People tracking with a mobile robot using sample-based joint probabilistic data association filters. *I. J. Robotic Res.*, 22(2), 2003.

[29] C. Sminchisescu and B. Triggs. Covariance-scaled sampling for monocular 3D body tracking. In *Proc. of the IEEE Int. Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2001.

[30] L. Spinello and K. O. Arras. People detection in rgb-d data. In *Proc. of the IEEE Int. Conf. on Intelligent Robots and Systems (IROS)*, 2011.

[31] P. Viola and M. Jones. Rapid object detection using a boosted cascade of simple features. In *Proc. of the IEEE Int. Conf. on Computer Vision and Pattern Recognition (CVPR)*, pages 511–518, 2001.

[32] P. Viola, M. Jones, and D. Snow. Detecting pedestrians using patterns of motion and appearance. *Int. J. Comput. Vision*, 63(2), 2005.

[33] D. Wagner and D. D. Schmalstieg. Artoolkitplus for pose tracking on mobile devices, 2007.

[34] S. Weiss, M. Achtelik, S. Lynen, M. Chli, and R. Siegwart. Real-time onboard visual-inertial state estimation and self-calibration of mavs in unknown environments. In *Proc. of the IEEE Int. Conf. on Robotics and Automation (ICRA)*, 2012.

[35] L. Zhang, J. Sturm, D. Cremers, and D. Lee. Real-time human motion tracking using multiple depth cameras. In *Proc. of the IEEE Int. Conf. on Intelligent Robot Systems (IROS)*, 2012.