A semi-autonomous UAV platform for indoor remote operation with visual and haptic feedback

Paolo Stegagno, Massimo Basile, Heinrich H Bülthoff, Antonio Franchi

Preprint version, final version at http://ieeexplore.ieee.org/ 2014 IEEE Int. Conf. on Robotics and Automation, Hong Kong, China

A Semi-autonomous UAV Platform for Indoor Remote Operation with Visual and Haptic Feedback

Paolo Stegagno, Massimo Basile, Heinrich H. Bülthoff, and Antonio Franchi

***Abstract*— We present the development of a semi-autonomous quadrotor UAV platform for indoor teleoperation using RGB-D technology as exceroceptive sensor. The platform integrates IMU and Dense Visual Odometry pose estimation in order to stabilize the UAV velocity and track the desired velocity commanded by a remote operator though an haptic interface. While being commanded, the quadrotor autonomously performs a persistent pan-scanning of the surrounding area in order to extend the intrinsically limited field of view. The RGB-D sensor is used also for collision-safe navigation using a probabilistically updated local obstacle map. In the operator visual feedback, pan-scanning movement is real time compensated by an IMU-based adaptive filtering algorithm that lets the operator perform the drive experience in a oscillation-free frame. An additional sensory channel for the operator is provided by the haptic feedback, which is based on the obstacle map and velocity tracking error in order to convey information about the environment and quadrotor state. The effectiveness of the platform is validated by means of experiments performed without the aid of any external positioning system.**

I. Introduction

Micro UAVs constitute the ideal platform for many robotic task, such as exploration, mapping, and surveillance. The unconstrained workspace and versatility allow to use them as flying sensors and actuators to reach and operate on places that are out of the range of more classical ground mobile robots. On the other hand many real-world tasks require (because of their nature or due to governmental regulation) that one or more humans participate to the mission in the quality of either simple supervisors or skilled operators, e.g., in the case of search-and-rescue missions [1],

Recent works have investigated the role of haptic feedback and the fact that it can be successfully used in order to increase the operator situational awareness (see, e.g., [2] and references therein) and therefore to have a positive impact on the human decisions. For this reason haptic shared control of UAVs represents an emerging topic attracting the attention of many research groups in the recent years.

Concerning the single-UAV case, an extensive study has been already done, with special regard on the theoretical point of view. The authors of [2] have studied how to properly design artificial force fields for the haptic cue when bilaterally teleoperating a UAV, while [3] presented the design of an admittance control paradigm from the master side with position feedback. Single-UAV teleoperation control based on the port-Hamiltonian approach has been presented by [4] and extended by [5]. In [6] a strategy to generate the haptic feedback as a virtual force based on both telemetric and optic flow sensors is designed. A novel force feedback user interface for mobile robotic vehicles with dynamics has been shown by [7], and a novel force feedback algorithm that allows the user to feel the texture of the environment has been recently presented by [8]. Finally in [9], [10] the authors have shown how single-robot haptic teleoperation can be seamlessly integrated with complex path planning techniques.

Concerning the, more challenging, multi-UAV case, [11], [12], [13], [14] presented an extensive framework to control a group of UAVs that can be interfaced with multiple operators by means of haptic devices, e.g., to control some generalized velocity of the group formation, and to receive a feedback that is informative of the tracking quality, the swarm status, and properties of the surrounding environment, such as presence of obstacles or wind gusts. In [15], the authors show how that framework can be applied to perform teleoperation over intercontinental distances.

The majority of the works never addressed the problem in a real world scenario, either employing simulation or external motion capture systems. Even though in [8] the obstacles are detected through a laser scanner the state for control purpose is still retrieved by an external camera system. Similarly, in [12] on-board cameras are used to measure the relative bearings, but the velocities were obtained through an external motion capture system. At the best of our knowledge none of the approaches dealing with haptic-teleoperation of UAVs have been experimentally proven on a platform that uses onboard sensors only.

The goal of this paper is therefore to present a UAV platform designed for haptic teleoperation that can be easily operated using velocity control in real unstructured scenarios providing safety against obstacles and relying on onboard sensor only, namely IMU and RGB-D measurements. In fact, this essential sensor equipment, thanks to the presence of a depth camera, is relatively richer with respect to the standard IMU-camera integration setting.

While being commanded, the quadrotor autonomously performs several tasks to improve safety and ease of operation.

First, it extends the intrinsically limited field of view (FOV) of the exteroceptive sensor executing a persistent pan-scanning of the local surrounding area using its yaw degree of freedom. In addition, the camera sensor is used for collision-safe navigation, enacting obstacle avoidance on a local obstacle map attached to the quadrotor body frame, that

P. Stegagno, M. Basile, H. H. Bülthoff and A. Franchi are with the Max Planck Institute for Biological Cybernetics, Department of Human Perception Cognition and Action, Spemannstraße 38, 72076 Tübingen, Germany { paolo.stegagno, mbasile, hhb}@tuebingen.mpg.de.

A. Franchi is also with Centre National de la Recherche Scientifique (CNRS), Laboratoire d'Analyse et d'Architecture des Systèmes (LAAS), 7 Avenue du Colonel Roche, 31077 Toulouse CEDEX 4, France. antonio.franchi@laas.fr

Fig. 1: The quadrotor setup.

is probabilistically updated while pan-scanning, thus fully exploiting the augmented FOV.

The operator will receive two feedbacks from the UAV: visual feedback from the camera and haptic feedback. From the point of view of the operator, the pan-scanning movement is realtime compensated by an IMU-based adaptive filtering algorithm that lets the operator perform the drive experience in a oscillation-free frame that is subject to the operator commands only. Haptic feedback is based on velocity tracking error and the obstacle map, thus creating and additional sensory channel for the operator in order to convey enhanced information about the environment and quadrotor state. The effectiveness of the platform is validated by means of experiments performed without the aid of any external positioning system.

The choice of using an RGB-D camera brings several advantages and also some drawbacks. First, depth measurements are extremely useful because they allow a *metric* estimation of the velocity. Monocular camera methods, as, e.g., the ones based on PTAM [16], do not provide metric information directly and typically need additional sensor fusion with the accelerometer reading, thus requiring a persistently accelerated motion to properly work metrically. In addition, the measurements coming from an RGB-D sensor can be easily used to perform reliable obstacle avoidance. On the other side, RGB-D sensors are usually sensible to natural light, so our system is specifically designed for indoor navigation. To overcome this issue in future, we may take into consideration the possibility to substitute the RGB-D sensor with a stereo camera in the outdoor phases.

The rest of the paper is organized as follows. Section II presents the system setup. Section III presents the generation of the reference velocity and its application in the flight controller, Section IV and Section V address the velocity estimation and the generation of a local obstacle map respectively, Section VI addresses the visual and haptic feedback provided to the human operator. Section VII presents some experimental results and Section VIII concludes the paper.

II. System Setup

The hardware configuration of the system is based on the mechanical frame, actuators, microcontrollers, and inertial measurement unit (IMU) of the MK-Quadro from MikroKopter.Its actuation system consists of four plastic propellers with a diameter of 0.254 m, and a total span and weight of the frame of 0.5 m and 0.12 kg, respectively.

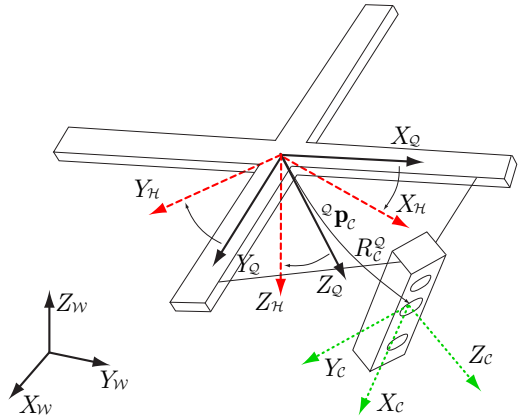

Fig. 2: A representation of all frames involved in the control and estimation of the velocity of the quadrotor.

The propellers are driven by four brushless controllers connected to a control board through a standard I^2C bus. The core of the control board is a 8-bit Atmega1284p microcontroller operating at 20 MHz. The board also includes three 3D LIS344alh accelerometers (0.0039g0 m/s2 resolution and 2g0 m/s2 range) and three ADXRS610 gyros (0.586 deg/s resolution and 300 deg/s range), directly connected to the analog to digital 10-bit converters of the low-level microcontroller and a pressure sensor MPX4115A.

The manufacturer provides the board pre-installed with its own firmware to drive the quadrotor with a remote control, which we have substituted by a different software that has new features and a different interface that allows to control the robot through serial interface. In detail, we have established two xBee serial connections operating at a baud-rate of 115 200 Bd on the two serial ports offered by the board. While one connection is employed to send and receive commands and data to and from the microcontroller, the second connection operates as a one-way channel to retrieve IMU and gyro measurements at high-frequency (500 Hz). The whole system is powered by a 2600 mAh LiPo battery which guarantees an endurance of around 10 min of flight in normal regimes.

In addition, we have retrofitted the MK-Quadro frame with an Asus Xtion RGB-D sensor to obtain exteroceptive measurements of the environment. The RGB-D sensor, from now on referred to simply as 'camera', is rigidly attached to the frame through three 5 mm diameter plastic bars, heading approximatively at 45° on the right of the quadrotor and tilted by approximatively 30° downward, vertically mounted to increase the vertical FOV. At current state of development, since the control board does not provide any USB channel nor the computational power to handle the image stream, the camera is connected to the base station through a wired USB connection system. We are almost ready to integrate in the near future a quad core Odroid-family board able to interfacing with the camera and perform the whole control onboard. The complete system, depicted in Fig. 1, has a weight of approximately 1.000 kg.

Figure 2 represents the relevant frames used in this work. Denote with $\mathcal{W} : \{O_{\mathcal{W}}, X_{\mathcal{W}}, Y_{\mathcal{W}}, Z_{\mathcal{W}}\}$ the inertial (world) frame defined with the North-West-Up (NWU) convention,

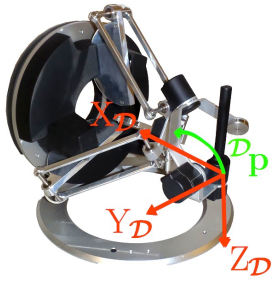

Fig. 3: The haptic device and its frame of reference.

hence with $Z_{\mathcal{W}}$ pointing in the opposite direction of the gravity vector, and with $\mathcal{Q} : \{O_{\mathcal{Q}}, X_{\mathcal{Q}}, Y_{\mathcal{Q}}, Z_{\mathcal{Q}}\}$ a frame attached to a representative point of the quadrotor (ideally its center of mass), which conforms to the North-East-Down (NED) convention as common in the aerospace field. In general, we will denote with ${}^{\mathcal{A}}\mathbf{p}_{\mathcal{B}}$ the position of the origin of a frame $\mathcal{B}$ in another frame $\mathcal{A}$ and with $\mathbf{R}^{\mathcal{A}}_{\mathcal{B}} \in SO(3)$ the rotation matrix expressing the orientation of the frame $\mathcal{B}$ in $\mathcal{A}$. Hence, ${}^{\mathcal{W}}\mathbf{p}_{\mathcal{Q}} \in \mathbb{R}^3$ and $\mathbf{R}^{\mathcal{W}}_{\mathcal{Q}} \in SO(3)$ represent the position and orientation of $\mathcal{Q}$ in $\mathcal{W}$, respectively. Finally, denote with ϕ, θ, ψ respectively the roll, pitch and yaw angles that represent the orientation of the quadrotor in $\mathcal{W}$, i.e., such that $\mathbf{R}^{\mathcal{W}}_{\mathcal{Q}} = \mathbf{R}_x(\pi)\mathbf{R}_z(\psi)\mathbf{R}_y(\theta)\mathbf{R}_x(\phi)$, where $\mathbf{R}_x(\cdot)$, $\mathbf{R}_y(\cdot)$, $\mathbf{R}_z(\cdot)$ represent the canonical rotation matrices about the axes X, Y, and Z respectively. Therefore, e.g., $\mathbf{R}_x(\pi)$ is the rotation matrix matrix transforming the NED coordinates in NWU.

In order to express the human commands we introduce the (NED) horizontal frame $\mathcal{H} : \{O_{\mathcal{H}}, X_{\mathcal{H}}, Y_{\mathcal{H}}, Z_{\mathcal{H}}\}$ such that $O_{\mathcal{H}} \equiv O_{\mathcal{Q}}$ and $Z_{\mathcal{H}} \parallel -Z_{\mathcal{W}}$. Then, the rotation matrix between $\mathcal{H}$ and $\mathcal{Q}$ is $\mathbf{R}^{\mathcal{H}}_{\mathcal{Q}} = \mathbf{R}_y(\theta)\mathbf{R}_x(\phi)$. Finally, we consider the camera frame $\mathcal{C} : \{O_{\mathcal{C}}, X_{\mathcal{C}}, Y_{\mathcal{C}}, Z_{\mathcal{C}}\}$. Since the camera is rigidly attached to the quadrotor, ${}^{\mathcal{Q}}\mathbf{p}_{\mathcal{C}}$ and $\mathbf{R}^{\mathcal{Q}}_{\mathcal{C}}$ are constant extrinsic parameters.

A human operator interfaces with the system and remotely operate the UAV through an omega.6 haptic device, shown in Fig. 3, and a standard 22" screen. The device provides a handle with six degrees of freedom (DOFs), three translational and three rotational, offering complete motion to a 3D rigid body. However, we have limited our system to use only the three translational DOFs.

In order to express all omega.6-related quantities, we define $\mathcal{D} : \{O_{\mathcal{D}}, X_{\mathcal{D}}, Y_{\mathcal{D}}, Z_{\mathcal{D}}\}$ as the inertial NED frame of reference whose origin is located in the steady position of the end effector, placed at the center of its Cartesian workspace. The translational dynamics of the haptic device can be modeled through the Lagrangian equation:

$$M(\mathbf{q})\ddot{\mathbf{q}} + C(\mathbf{q}, \dot{\mathbf{q}})\dot{\mathbf{q}} = \tau + f \tag{1}$$

where $\mathbf{q} = (q_x\, q_y\, q_z)^T \in \mathbb{R}^3$ is the position of the handle in $\mathcal{D}$, $M(\mathbf{q}) \in \mathbb{R}^{3\times3}$ is the positive-definite/symmetric inertia matrix, $C(\mathbf{q}, \dot{\mathbf{q}}) \in \mathbb{R}^{3\times3}$ is the Coriolis matrix, and τ, $f \in \mathbb{R}^3$ are the control and human forces respectively.

III. Reference Velocity Generation and Execution

Apart from the attitude of the quadrotor, represented by ϕ, θ, the position ${}^{\mathcal{W}}\mathbf{p}_{\mathcal{Q}}$ and the yaw ψ are not observable without performing a full SLAM approach, since the robot has only relative measurements of the environment and no absolute positioning capabilities or compass. Hence, it is not possible to regulate the motion on a specific desired position or yaw orientation, and the UAV can only follow a *reference velocity vector* and *reference yaw rate*. Clearly one could integrate the measurements and the motion to obtain an odometry-like estimation of the relative position and yaw from the starting pose. However this method can be employed only for a short time since it would unavoidably drift due to the aforementioned observability issue. Another way would be to implement a computationally expensive full SLAM approach. However, the focus of this work is to develop a basic, solid, and adaptable platform to perform indoor aerial bilateral teleoperation in which the decision on high the level motion strategies are delegated to the human operator.

Denote with ${}^{\mathcal{H}}\mathbf{v}_r$ and $\dot{\psi}_r$ the reference velocity vector and yaw-rate, respectively. The velocity ${}^{\mathcal{H}}\mathbf{v}_r$ is conveniently expressed in the horizontal frame $\mathcal{H}$ in order to abstract from the current attitude of the quadrotor. Note also, that driving the UAV in the frame $\mathcal{H}$ instead of the frame $\mathcal{B}$ is much more convenient for the human operator. These reference quantities are provided to a flight controller (referred to as 'tracker' in the following) that uses the estimated state of the robot (computed as per Sec. IV) to regulate the tracking error to zero. The tracker, described in [14], is a simple PID controller with gravity compensation computing the required thrust, roll and pitch angles that are needed to accelerate as requested and therefore applying the torques that are necessary to track tht desired angles. Note that the integral term is not performed using the position of the UAV, as usual when dealing with localization issues, being that information unobservable. Hence, the integral term is computed simply integrating the error of the velocity. Yaw rate is tracked in a similar fashion using the yaw torque command.

Some steps are performed in order to generate ${}^{\mathcal{H}}\mathbf{v}_r$ and $\dot{\psi}_r$ on the basis of the commands of the human operator and other autonomous actions. In fact, although the robot tries to reproduce the velocity and yaw rate commanded by the operator, it also carries on some tasks autonomously in order to guarantee safety during normal operation and improve the capability of the sensors, as described in the following.

The first step to obtain ${}^{\mathcal{H}}\mathbf{v}_r$ and $\dot{\psi}_r$ is the computation of the desired velocity ${}^{\mathcal{H}}\mathbf{v}_d$ and yaw rate $\dot{\psi}_d$ commanded by the human operator by means of the haptic device. This is done mapping the 3D haptic device position as follows:

$${}^{\mathcal{H}}\mathbf{v}_d = k_v \begin{pmatrix} q_x \cos\alpha \\ q_x \sin\alpha \\ q_z \end{pmatrix} \tag{2}$$

$$\dot{\psi}_d = -k_\psi q_y \tag{3}$$

where k_v and k_ψ are positive gains and α is a parameter expressing the direction of the desired forward motion of the quadrotor on the horizontal plane of $\mathcal{H}$. The effect of (2-3) is that whenever the operator pushes forward the end effector of the haptic device, the robot will move on the horizontal plane the direction α, whenever the operator lifts the end effector the UAV will increase its altitude (and vice versa),

and whenever the operator moves the end effector sideways the UAV will rotate its yaw angle accordingly. No motion command is then allowed on the direction orthogonal to α.

The more meaningful way to chose α would be to select the yaw angle of the camera in $\mathcal{Q}$ as $\alpha = \text{atan2}(\text{r}_{21}, \text{r}_{11})$. where $\mathbf{R}_{\mathcal{C}}^{\mathcal{Q}} = [r_{ij}]_{i=1,\dots,3,\, j=1,\dots,3}$. However, in the following paragraphs we shall show a more sophisticated method related to the pan-scanning motion of the quadrotor aimed at increasing the virtual FOV of its exteroceptive sensor.

The first and most important task performed autonomously by the robot is obstacle avoidance. The RGB-D camera is the only exteroceptive onboard sensor, but is particularly suited for this purpose. Our obstacle avoidance module divides the depth camera images retrieved by the camera into $n \times m$ sectors, and extracts the closest point in each sector. Let be ${}^{\mathcal{C}}\mathbf{p}_{ij}$ the selected point of sector (i, j), expressed in the camera frame. Using extrinsic calibration parameters, i.e., ${}^{\mathcal{Q}}\mathbf{p}_{\mathcal{C}}$ and $\mathbf{R}_{\mathcal{C}}^{\mathcal{Q}}$, together with the roll and pitch angles of the UAV, i.e., ϕ, θ, it is possible to compute the position ${}^{\mathcal{H}}\mathbf{p}_{ij}$ in the horizontal frame $\mathcal{H}$. Then ${}^{\mathcal{H}}\mathbf{p}_{ij}$ will be considered an obstacle, and a repulsive artificial potential will be computed for each (i, j). The weighted sum of all contributions produces an obstacle avoidance velocity term ${}^{\mathcal{H}}\mathbf{v}_o$ that is then summed to the operator desired velocity.

The main drawback of this obstacle avoidance algorithm is the very limited FOV of the sensor. However, we can take advantage of the fact that the execution of the yaw rate is independent from the execution of the UAV Cartesian velocity. Therefore, even though it is not possible to extend the instantaneous FOV of the sensor, we extend it over time by adding a sinusoidal signal $\dot{\psi}_s$ on the commanded yaw rate

$$\dot{\psi}_s = A \sin(\omega t) \tag{4}$$

where A and ω represent parameters opportunely chosen and t represents the time of the experiment. If the yaw dynamics perfectly tracks $\dot{\psi}_s$ then the result in the motion of the UAV is a co-sinusoidal oscillation of the yaw angle ψ with amplitude A/ω and pulsation ω. However, under action of the tracker, the actual closed loop yaw dynamics is not a perfect integrator but a first order linear system which then generates a yaw angle ψ that will oscillate with amplitude $A'_\omega A/\omega$, pulsation ω and phase φ_ω. It is not difficult to compute both A'_ω and φ_ω from the controller parameters or to identify them through a few empirical measurements.

Addition of the term (4) will cause the camera to span a broad angle and the robot to avoid obstacles which would be otherwise ignored due to the limitation of the sensor. Nevertheless, the oscillation causes a serious threaten to the ease of operation because of two side effects.

First, if the UAV is commanded in the pointing direction of the yaw angle of the camera, it would oscillate with it. Hence, the operator trying to drive the UAV on a straight planar line will result in the execution of a sinusoid on the plane. Second, the visualization of the camera stream is now very uncomfortable for the operator, because of the continuous oscillation of the camera frame.

It is worth to mention that the first issue would be avoided

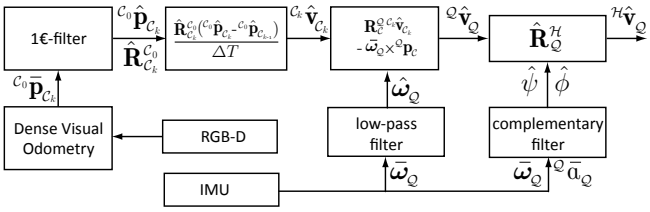

Fig. 4: A block scheme representation of the velocity estimation system.

if the camera was mounted on a pan actuator, rotating independently from the orientation of the UAV. Nevertheless this solution implies new hardware addition, a new actuator and ultimately more weight and less time of flight. Hence, we found it more convenient to exploit the intrinsic characteristic of the quadrotor at its best.

However we have conceived and developed solutions for both the actuation and visual issue. We describe the first in the following and we shall defer the description of our solution for the second issue to Sec. VI-B.

In the considered situation, the best solution to drive the UAV is to have the robot moving always in the direction of the center of the oscillation. In this way, the operator has the opportunity of 'taking a look' to what happens on the two sides of the direction of its motion. This can be easily done by rotating the first two components of (2) by the additional angle $\bar{\psi}_s$ introduced by the oscillation:

$$\bar{\psi}_s = A'_\omega \frac{A}{\omega} \cos(\omega t + \varphi_\omega) \tag{5}$$

Hence, in this situation α can be conveniently chosen as

$$\alpha = \text{atan2}(\text{r}_{21}, \text{r}_{11}) + \bar{\psi}_s \tag{6}$$

Wrapping up, the reference velocity ${}^{\mathcal{H}}\mathbf{v}_r$ for the controller is computed as

$${}^{\mathcal{H}}\mathbf{v}_r = {}^{\mathcal{H}}\mathbf{v}_d + {}^{\mathcal{H}}\mathbf{v}_o = k_v \begin{pmatrix} q_x \cos\alpha \\ q_x \sin\alpha \\ q_z \end{pmatrix} + {}^{\mathcal{H}}\mathbf{v}_o \tag{7}$$

with α given by (6), and the reference yaw rate is computed as

$$\dot{\psi}_r = \dot{\psi}_d + \dot{\psi}_s = -k_\psi q_y + A \sin(\omega t) \tag{8}$$

Other additional autonomous tasks can be added in the proposed scheme. For example we easily integrated autonomous take off and landing subroutines in our system that are not described here for the sake of space.

Finally, in order to take full advantage of the sinusoidal oscillation introduced in the yaw rate of the UAV, a filtering on the measured obstacles ${}^{\mathcal{H}}\mathbf{p}_{ij}$ is needed in the obstacle avoidance module. In fact, an appropriate filter would give the opportunity to maintain an estimate of close obstacles that are not in the FOV of the camera but have been measured in the immediate past. We actually implemented such a filter and we describe it in Sec. V.

IV. State Estimation

The scheme presented in Sec. III needs some quantities that must be retrieved by the UAV onboard sensors. The working principle of our estimation system is summarized in Fig. 4, and uses both measurements of the IMU and of the depth-camera. The first ones are used in a complementary filter to compute estimates $\hat{\phi}$, $\hat{\theta}$ of the roll and pitch angles as described in [17], [18]. In addition, a low-pass filter improves the angular velocity measurements ${}^{\mathcal{Q}}\bar{\omega}_{\mathcal{Q}}$ from the gyros producing an estimate denoted by ${}^{\mathcal{Q}}\hat{\omega}_{\mathcal{Q}}$.

Once the attitude (i.e., roll and pitch) of the quadrotor is known, the images from the depth camera are used to obtain an estimate of the velocity of the quadrotor in the frame $\mathcal{H}$.

At each time-step k the images are used to feed the dvo [1] algorithm [19] which provides the estimates ${}^{\mathcal{C}_0}\bar{\mathbf{p}}_{\mathcal{C}_k}$, $\bar{\mathbf{R}}^{\mathcal{C}_0}_{\mathcal{C}_k}$ of the position ${}^{\mathcal{C}_0}\mathbf{p}_{\mathcal{C}_k}$ and orientation $\mathbf{R}^{\mathcal{C}_0}_{\mathcal{C}_k}$ of the camera frame $\mathcal{C}_k$ at time-step k w.r.t. the camera frame $\mathcal{C}_0$ at time-step 0. Obviously, since dvo performs a visual odometry algorithm, the estimates will eventually diverge from the true value and cannot be used for a long time to obtain absolute position and orientation measurements. Nevertheless, it is possible to extract a noisy but non-drifting measurement of the velocity ${}^{\mathcal{C}_k}\mathbf{v}_{\mathcal{C}_k}$, i.e., the velocity of the origin $O_{\mathcal{C}_k}$ of the frame $\mathcal{C}_k$ expressed in $\mathcal{C}_k$, through the equation:

$$ {}^{\mathcal{C}_k}\mathbf{v}_{\mathcal{C}_k} = \frac{\mathbf{R}^{\mathcal{C}_k}_{\mathcal{C}_0}({}^{\mathcal{C}_0}\mathbf{p}_{\mathcal{C}_k} - {}^{\mathcal{C}_0}\mathbf{p}_{\mathcal{C}_{k-1}})}{\Delta T} \tag{9} $$

where $\mathcal{C}_{k-1}$ denotes the camera frame at time $k-1$ and ΔT is the elapsed time between time-steps $k-1$ and k. However, since (9) corresponds to a first order numerical derivation of the position ${}^{\mathcal{C}_0}\mathbf{p}_{\mathcal{C}}$ it would be considerably affected by noise. For this reason, instead of (9), we use

$$ {}^{\mathcal{C}_k}\hat{\mathbf{v}}_{\mathcal{C}_k} = \frac{\hat{\mathbf{R}}^{\mathcal{C}_k}_{\mathcal{C}_0}({}^{\mathcal{C}_0}\hat{\mathbf{p}}_{\mathcal{C}_k} - {}^{\mathcal{C}_0}\hat{\mathbf{p}}_{\mathcal{C}_{k-1}})}{\Delta T} \tag{10} $$

where ${}^{\mathcal{C}_0}\hat{\mathbf{p}}_{\mathcal{C}_k}$, and $\hat{\mathbf{R}}^{\mathcal{C}_0}_{\mathcal{C}_k}$ are the 1€-filtered [20] versions of ${}^{\mathcal{C}_0}\bar{\mathbf{p}}_{\mathcal{C}_k}$, and $\bar{\mathbf{R}}^{\mathcal{C}_0}_{\mathcal{C}_k}$ respectively.

The velocity ${}^{\mathcal{C}}\mathbf{v}_{\mathcal{C}}$ of $O_{\mathcal{C}}$ in $\mathcal{C}$ can be written as

$$ {}^{\mathcal{C}}\mathbf{v}_{\mathcal{C}} = \mathbf{R}^{\mathcal{C}}_{\mathcal{Q}}\,{}^{\mathcal{Q}}\mathbf{v}_{\mathcal{C}} = \mathbf{R}^{\mathcal{C}}_{\mathcal{Q}}({}^{\mathcal{Q}}\mathbf{v}_{\mathcal{Q}} + {}^{\mathcal{Q}}\omega_{\mathcal{Q}} \times {}^{\mathcal{Q}}\mathbf{p}_{\mathcal{C}}). \tag{11} $$

Therefore we compute an estimate of ${}^{\mathcal{Q}}\mathbf{v}_{\mathcal{Q}}$ at time-step k as

$$ {}^{\mathcal{Q}}\hat{\mathbf{v}}_{\mathcal{Q}} = \mathbf{R}^{\mathcal{Q}\mathcal{C}_k}_{\mathcal{C}}\,{}^{\mathcal{C}_k}\hat{\mathbf{v}}_{\mathcal{C}_k} - {}^{\mathcal{Q}}\hat{\omega}_{\mathcal{Q}} \times {}^{\mathcal{Q}}\mathbf{p}_{\mathcal{C}} \tag{12} $$

Finally, given the estimates $\hat{\phi}$, $\hat{\theta}$, we obtain the sought velocity in the $\mathcal{H}$ frame

$$ {}^{\mathcal{H}}\hat{\mathbf{v}}_{\mathcal{Q}} = \hat{\mathbf{R}}^{\mathcal{H}\mathcal{Q}}_{\mathcal{Q}}\,{}^{\mathcal{Q}}\hat{\mathbf{v}}_{\mathcal{Q}} \tag{13} $$

which is then used in the velocity tracker in order to follow the velocity commanded by the operator and the obstacle avoidance module.

[1] https://github.com/tum-vision/dvo

V. Local Obstacle Maps

In order to further extend the FOV of the camera and increase the safety of the flights, the system builds and propagate in time a local map of the obstacles. The obstacle avoidance will be then performed on the estimates provided by this module. Nevertheless, this module does not perform a SLAM algorithm, since there is no localization purpose in its operation and the produced obstacle map will be limited both in time and space.

The working principle is that of a Bayesian filter. A measurement step initializes and improves the current estimates, while a time update estimate propagate current estimates in time using the state of the UAV.

The main idea is to divide the world surrounding the UAV in cells by a discretization of azimuth and zenith distance angles. Whenever a measurement occurs (i.e., a set of closest points is extracted by a frame from the camera as explained in Sec. III), each measured point is assigned to the correct cell based on its relative azimuth and zenith distance, and overwrites any other point possibly present in that cell. This is to obtain a reactive behavior of the estimates. More copies of the same obstacle point are introduced in the cell, in order to be used as particles.

The time update uses as input the estimated velocity ${}^{\mathcal{H}}\hat{\mathbf{v}}_{\mathcal{Q}}$ and yaw rate $\dot{\psi}$ of the UAV which is used to propagate each estimated obstacle points $\hat{\mathbf{p}}_j$ according to the following:

$$ \hat{\mathbf{p}}_j^{k+1} = \mathbf{R}^T_{k\,k+1}(\hat{\mathbf{p}}_j^k - \mathbf{u}) \tag{14} $$

where $\hat{\mathbf{p}}_j^{k+1}$ is the new estimate, $\mathbf{u} = \Delta T({}^{\mathcal{H}}\mathbf{v}_{\mathcal{Q}}^{k+1} - {}^{\mathcal{H}}\mathbf{v}_{\mathcal{Q}}^{k} + \mathbf{n}_v)/2$ is the estimated linear displacement of the UAV between time k and time $k+1$, $\mathbf{R}_{k\,k+1} = \mathbf{R}_z(\psi_{k\,k+1})$ is the rotation matrix about the vertical axis due to the yaw displacement $\psi_{k\,k+1} = \Delta T(\dot{\psi}^{k+1} - \dot{\psi}^k + \mathbf{n}_\psi)/2$ between time k and time $k+1$, ΔT is the elapsed time between k and $k+1$, and $\mathbf{n}_v$, $\mathbf{n}_\psi$ are two samples randomly extracted on the model of the noise over the estimates of the velocity and the yaw rate, both assumed to be gaussian random variables with zero mean and known covariance.

Examples of obstacle maps can be observed in the middle column of Figs. 9 and 10, which shows the image plane of the onboard camera with reprojection of the the 3D obstacle points (red dots). Note the particles in the right part of the third row, middle column, Fig. 9, which are propagated and not directly measured, since the obstacles are not in the FOV.

VI. Visual/Haptic Feedback for the Operator

A. Haptic Feedback

The main focus of this work is to enable a human operator to remotely control the UAV using only onboard sensors. At this aim, great importance assumes the feedback that the human receives from the UAV and his interface with the system. Hence, we propose a dual haptic and visual feedback in order to enable the operator to safely control the robot.

The haptic feedback algorithm resembles the one proposed in [14] but with a special adaptation to our particular input paradigm and pan-scanning scheme. The force feedback

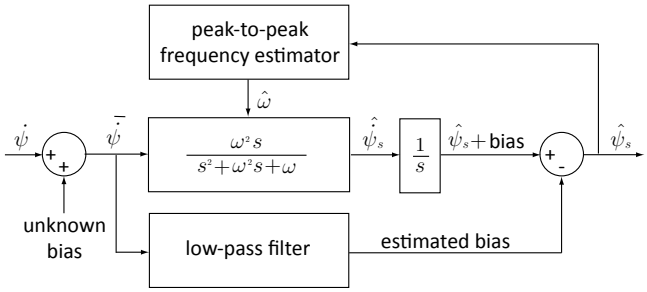

Fig. 5: A block scheme of the system for the estimation of ψ_s.

provided by the device to the operator is governed by:

$$\tau(t) := -B\dot{\mathbf{q}} - K_f\mathbf{q} - K(\mathbf{q} - \tilde{\mathbf{q}}) \tag{15}$$

where B, K, $K_f \in \mathbb{R}^{3\times 3}$ are the positive-(semi)definite diagonal gain matrices, and $\tilde{\mathbf{v}}$ is defined as the vector $\tilde{\mathbf{q}} = (\tilde{q}_x\ \tilde{q}_\psi\ \tilde{q}_z)^T$ is such that

$$\tilde{q}_x = \frac{1}{k_v}(\cos\alpha \quad \sin\alpha)\begin{pmatrix} {}^{\mathcal{H}}\hat{\mathbf{v}}^x_{\mathcal{Q}} \\ {}^{\mathcal{H}}\hat{\mathbf{v}}^y_{\mathcal{Q}} \end{pmatrix} \tag{16}$$

$$\tilde{q}_\psi = \frac{1}{k_\psi}\left(\hat{\dot{\psi}} - \dot{\bar{\psi}}_s\right) \tag{17}$$

$$\tilde{q}_z = \frac{1}{k_v}{}^{\mathcal{H}}\hat{\mathbf{v}}^z_{\mathcal{Q}} \tag{18}$$

where ${}^{\mathcal{H}}\hat{\mathbf{v}}^x_{\mathcal{Q}}, {}^{\mathcal{H}}\hat{\mathbf{v}}^y_{\mathcal{Q}}, {}^{\mathcal{H}}\hat{\mathbf{v}}^z_{\mathcal{Q}}$ are the 3 components of ${}^{\mathcal{H}}\hat{\mathbf{v}}_{\mathcal{Q}}$ in (13) and $\hat{\dot{\psi}}$ is the estimate of the yaw rate obtained using the IMU.

The force in (15) is composed of three terms: the first is a damping term, the second is a spring which tries to bring the end effector back in $O_{\mathcal{D}}$, and the third provides the real haptic cue, with a force proportional to the error between the commanded and actual velocity/yaw-rate. Note the presence of α and $\dot{\bar{\psi}}_s$ in (16) and (17), needed to cancel the effect of persistent pan-scanning thus making this additional movement transparent to the operator.

B. Visual Feedback

The visual feedback is more articulated due to the oscillatory (pan-scanning) component added to the reference yaw rate. As stated before, the simple visualization of the image stream coming from the camera would produce an unpleasant effect for the operator, since the image would continuously rapidly change, even in hovering condition. Moreover, this effect would make it difficult to identify at each time instant the forward direction of motion of the quadrotor in the image.

A much better visualization would be achieved if the image moved on the screen so that the forward direction of motion (the center of oscillation of the yaw) was still on the screen. Based on this principle, we have implemented a filter (Fig. 5) which is able to estimate the actual frequency and the center of the oscillation and prints the images on the display so as to keep always the center of oscillation in the center of the screen.

In particular, since a direct measurement of the yaw is not available, the filtering is performed on the yaw rate measured by the IMU. Clearly a simple integration of the measured yaw rate would not work, because of the presence of noise and unavoidable bias. This signal is passed through a band-pass filter which selects only the frequency $f = \omega/2\pi$ of the oscillation, erasing all other components. Note that the filter is designed to have unitary gain and zero phase shift at the selected frequency. In principle, we would like the frequency of the oscillation to be not known and variable during operation. For this reason, the band-pass filter is tuned accordingly to the output of a peak-to-peak frequency estimation filter.

The output of this two block system is an estimate $\hat{\dot{\psi}}_s$ of the yaw rate due to the additional oscillation, which integrated over time returns an estimate of the yaw displacement with respect to the direction of motion. Nevertheless, this estimate is affected by a bias that can be estimated low-pass filtering the initial signal, hence subtracted in order to obtain an unbiased estimate $\hat{\psi}_s$. Then, the images on the screen are printed with a horizontal displacement in pixel equal to $px = k_{px}\hat{\psi}_s$, where k_{px} is a conversion factor computed through the knowledge of the image size and the angular FOV.

Similar compensations have been enacted also for the pitch and roll angles, but no filtering is needed because they are directly estimated by the UAV. Additional features to improve the pleasantness of the remote operation are the exponential decay of the images in the background and the re-projection on screen of the obstacle points estimated by the filter. Some example frames are given in Figs. 9 and 10, however to fully appreciate its dynamics we suggest to watch the attached video of the experiments.

VII. Experimental Validation

The main framework in which the platform is developed is TeleKyb [21], a ROS-based project specifically designed for the development of applications on UAVs and oriented to multi-robot execution. In addition to TeleKyb other general purpose tools as Matlab and Openni have been used in order to accomplish preliminary tasks, as the calibration of the camera and quadrotor frames, and online camera stream acquisition.

We have conducted several experiments in order to evaluate the performance of the proposed system. In all the experiments we used the estimated quantities in the flight controller. Additionally, in order to numerically evaluate the accuracy of the estimation algorithm, we used an external motion capture system as ground truth.We report here the results of a representative experiment. A clip of this and other experiments can be watched in the accompanying video and at `http://antoniofranchi.com/videos/onboard_haptele_icra.html`. The experiment has been performed in a 10 m $\times$ 10 m arena with a tall obstacle placed approximately at its center. During the experiment, the operator is not able to see directly the UAV and uses only the provided feedback to control the system. Since the arena is completely black and the estimation system exploits vision, it was necessary to add features to the environment, as a textured carpet, a desk, several boxes and other objects.

In Fig. 6 we show the plots of the estimated (blue), ground truth (red) and commanded (green) values of the x and

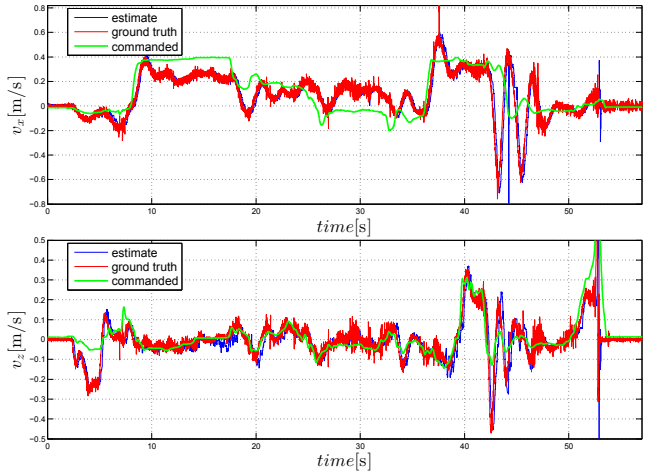

Fig. 6: Comparison between the 2 components of: the estimated velocity (blue plots), the ground truth velocity measured by an external motion capture system (red plots), and the velocity commanded by the human operator (green plots). All the velocities are expressed in the horizontal body frame $\mathcal{H}$.

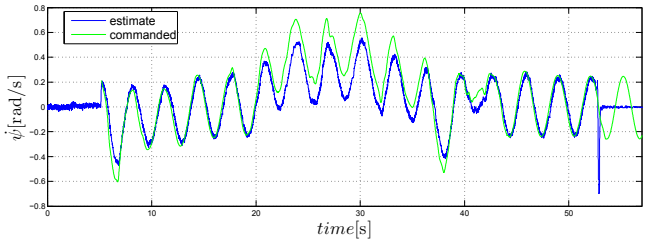

Fig. 7: Comparison between the measure of the yaw rate obtained from the gyroscope (blue plot) and the commanded yaw rate (green plot), i.e. the sum of the operator command and the pan-scanning sinusoidal carrier.

z velocity components expressed in $\mathcal{H}$. Commanded and measured yaw rate from the onboard gyroscope are shown in Fig. 7. The part of the rotation rate commanded by the operator is the mean of the sinusoidal plot. All plots show that the velocity estimate is very similar to the ground truth counterpart. In addition, even if not perfectly, the quadrotor reproduces quite faithfully the commanded velocity and yaw-rate. This precision level is definitely enough, to perform teleoperation, and constitutes a great achievement considering that it is obtained using only onboard measurements.

At the beginning the quadrotor is on the ground and at 2 s it takes off. Then the human operator commands the UAV to go straight in one direction until 25 s. Then the human operator rotates the UAV in the direction of the obstacle (in fact the mean of the sinusoidal plot increases from zero to 0.4 rad/s) and finally drives the UAV toward the central obstacle at constant speed (the phase lasts approximately from 36 s to 43 s).

At time 43 s the UAV comes close enough to the obstacle and the obstacle potential starts rising thus adding to the commanded velocity a repulsive component that lets the actual velocity greatly deviate from the commanded one. Between 43 s and 47 s the operator pushes the UAV against the obstacle twice thus generating two peaks in the actual velocities, which also significantly differs from commanded velocity. In this phase the operator feels high opposing forces

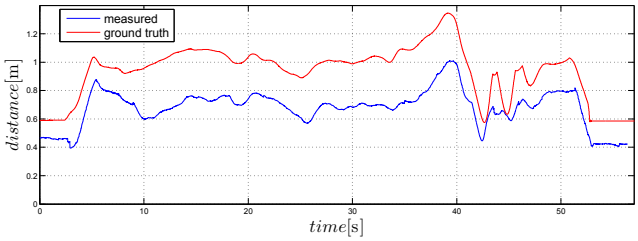

Fig. 8: Real (red plot) and measured (blue plot) distance between the quadrotor and the surrounding environment. When in flight (after 15 s) the minimum safe distance for the obstacle avoidance is set to 0.45 m

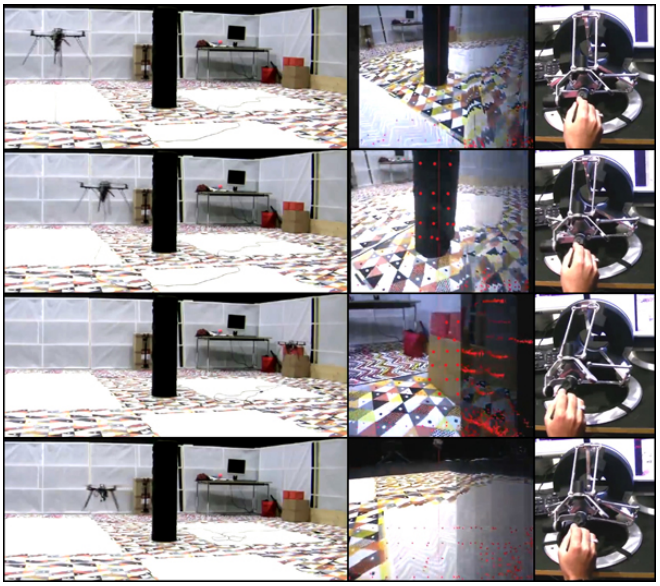

Fig. 9: Snapshots of a typical experiment in the arena. Each row refers to a different time instant. Left column: global views of the environment with the quadrotor and the obstacles. Middle column: onboard views from the visualizer (red dots are detected obstacles). Right column: haptic interface used by the human operator.

informing him/her about the presence of the obstacle. The experiment concludes with the UAV landing on the ground.

In Fig. 8 we show the evolution over time of the distance between the center of mass of the quadrotor and the surrounding obstacles, comprising the obstacle, the walls, the floor, and the ceiling. The distance obtained from the ground truth is shown with a red plot and the distance obtained fusing the depth-camera readings and the attitude estimation is shown with a blue plot. It is possible to see how the estimated distance always underestimates the real distance, which represents a good feature for the safety of the platform. In the second phase, when the UAV is flying, i.e., after approx. 42 s, as soon as the distance reaches the minimum admissible distance (which is set to 0.45 m), the quadrotor is pushed back from the obstacle. This happens in correspondence of the spikes in the plots of the velocities. Figure 9 presents some significant snapshots of the experiment with both global and onboard views, plus the haptic interface operated by the human.

In order to thoroughly validate the approach we conducted also experiments in real office environment, in which the UAV is driven over several desks. Some significant snapshots of one of these experiments are given in Fig. 10.

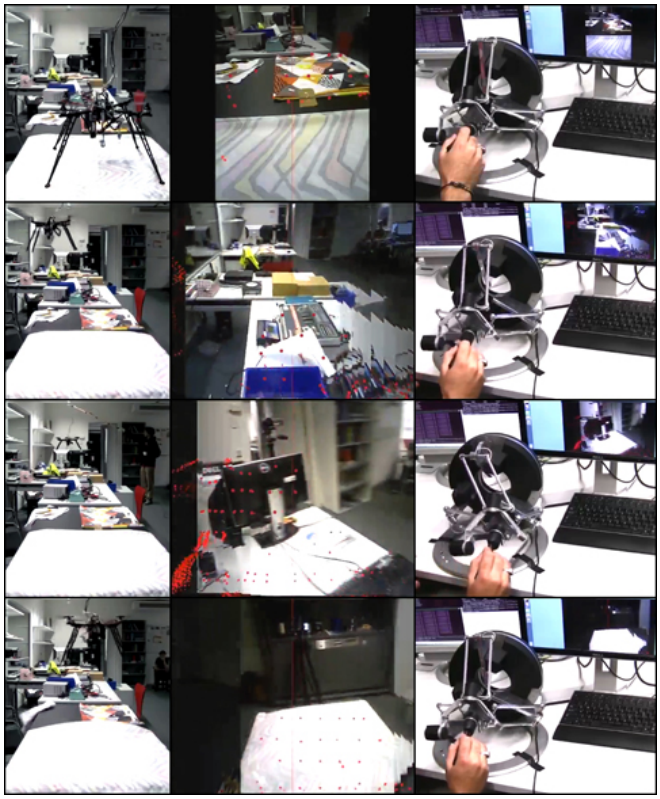

Fig. 10: Snapshots of a typical experiment in a real office environment. Each row refers to a different time instant. Left column: global views of the environment. Middle column: onboard views from the visualizer (red dots are detected obstacles). Right column: haptic interface used by the human operator. For safety reasons, a loose rope secures the UAV and is moved by a person following the UAV motion.

VIII. Discussion and Future Work

In this paper we have presented a semi-autonomous UAV platform that is used for indoor haptic teleoperation control and is able to exploit only onboard sensors, thus being independent from any motion capture system. No assumptions on the environment are needed, such, e.g., the presence of planar surfaces or objects of known sizes. The limitations of the sensor in terms of small FOV have been overcome by exploiting a pan-scanning action that also improves the detection of surrounding obstacles, estimated in a local moving map through filtering.

In the near future we are planning to install a single-board Odroid-based PC, which will be able to handle the camera images. Other improvements will consider different filtering strategies for the angular velocities and for the whole state. Once the platform is complete, we plan to employ it to perform teleoperation experiments over the internet, hence introducing significant delay on the commanded velocities.

References

[1] R. Murphy, S. Tadokoro, D. Nardi, A. Jacoff, P. Fiorini, H. Choset, and A. Erkmen, "Search and rescue robotics," in *Springer Handbook of Robotics*, B. Siciliano and O. Khatib, Eds. Springer, 2008, pp. 1151–1173.

[2] T. M. Lam, H. W. Boschloo, M. Mulder, and M. M. V. Paassen, "Artificial force field for haptic feedback in UAV teleoperation," *IEEE Trans. on Systems, Man, & Cybernetics. Part A: Systems & Humans*, vol. 39, no. 6, pp. 1316–1330, 2009.

[3] F. Schill, X. Hou, and R. Mahony, "Admittance mode framework for haptic teleoperation of hovering vehicles with unlimited workspace," in *2010 Australasian Conf. on Robotics & Automation*, Brisbane, Australia, December 2010.

[4] S. Stramigioli, R. Mahony, and P. Corke, "A novel approach to haptic tele-operation of aerial robot vehicles," in *2010 IEEE Int. Conf. on Robotics and Automation*, Anchorage, AK, May 2010, pp. 5302–5308.

[5] A. Y. Mersha, S. Stramigioli, and R. Carloni, "Switching-based mapping and control for haptic teleoperation of aerial robots," in *2012 IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, Vilamoura, Portugal, Oct. 2012, pp. 2629–2634.

[6] H. Rifa, M. D. Hua, T. Hamel, and P. Morin, "Haptic-based bilateral teleoperation of underactuated unmanned aerial vehicles," in *18th IFAC World Congress*, Milano, Italy, Aug. 2011, pp. 13 782–13 788.

[7] X. Hou, R. Mahony, and F. S. Schill, "Representation of vehicle dynamics in haptic teleoperation of aerial robots," in *2013 IEEE Int. Conf. on Robotics and Automation*, Karlsruhe, Germany, May 2013, pp. 1477–1483.

[8] S. Omari, M. D. Hua, G. J. J. Ducard, and T. Hamel, "Bilateral haptic teleoperation of VTOL UAVs," in *2013 IEEE Int. Conf. on Robotics and Automation*, Karlsruhe, Germany, May 2013, pp. 2385–2391.

[9] C. Masone, A. Franchi, H. H. Bülthoff, and P. Robuffo Giordano, "Interactive planning of persistent trajectories for human-assisted navigation of mobile robots," in *2012 IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, Vilamoura, Portugal, Oct. 2012, pp. 2641–2648.

[10] C. Masone, P. Robuffo Giordano, H. H. Bülthoff, and A. Franchi, "Semi-autonomous trajectory generation for mobile robots with integral haptic shared control," in *2014 IEEE Int. Conf. on Robotics and Automation*, Hong Kong, China, May. 2014.

[11] A. Franchi, C. Secchi, M. Ryll, H. H. Bülthoff, and P. Robuffo Giordano, "Shared control: Balancing autonomy and human assistance with a group of quadrotor UAVs," *IEEE Robotics & Automation Magazine, Special Issue on Aerial Robotics and the Quadrotor Platform*, vol. 19, no. 3, pp. 57–68, 2012.

[12] A. Franchi, C. Masone, V. Grabe, M. Ryll, H. H. Bülthoff, and P. Robuffo Giordano, "Modeling and control of UAV bearing-formations with bilateral high-level steering," *The International Journal of Robotics Research, Special Issue on 3D Exploration, Mapping, and Surveillance*, vol. 31, no. 12, pp. 1504–1525, 2012.

[13] A. Franchi, C. Secchi, H. I. Son, H. H. Bülthoff, and P. Robuffo Giordano, "Bilateral teleoperation of groups of mobile robots with time-varying topology," *IEEE Trans. on Robotics*, vol. 28, no. 5, pp. 1019–1033, 2012.

[14] D. J. Lee, A. Franchi, H. I. Son, H. H. Bülthoff, and P. Robuffo Giordano, "Semi-autonomous haptic teleoperation control architecture of multiple unmanned aerial vehicles," *IEEE/ASME Trans. on Mechatronics, Focused Section on Aerospace Mechatronics*, vol. 18, no. 4, pp. 1334–1345, 2013.

[15] M. Riedel, A. Franchi, H. H. Bülthoff, P. Robuffo Giordano, and H. I. Son, "Experiments on intercontinental haptic control of multiple UAVs," in *12th Int. Conf. on Intelligent Autonomous Systems*, Jeju Island, Korea, Jun. 2012, pp. 227–238.

[16] D. Scaramuzza, M. C. Achtelik, L. Doitsidis, F. Fraundorfer, E. B. Kosmatopoulos, A. Martinelli, M. W. Achtelik, M. Chli, S. A. Chatzichristofis, L. Kneip, D. Gurdan, L. Heng, G. H. Lee, S. Lynen, L. Meier, M. Pollefeys, A. Renzaglia, R. Siegwart, J. C. Stumpf, P. Tanskanen, C. Troiani, and S. Weiss, "Vision-controlled micro flying robots: from system design to autonomous navigation and mapping in GPS-denied environments," *Accepted to IEEE Robotics & Automation Magazine*, 2013.

[17] R. Mahony, T. Hamel, and J.-M. Pflimlin, "Nonlinear complementary filters on the special orthogonal group," *IEEE Trans. on Automatic Control*, vol. 53, no. 5, pp. 1203–1218, 2008.

[18] P. Martin and E. Salaün, "The true role of accelerometer feedback in quadrotor control," in *2010 IEEE Int. Conf. on Robotics and Automation*, Anchorage, AK, May 2010, pp. 1623–1629.

[19] C. Kerl, J. Sturm, and D. Cremers, "Robust odometry estimation for RGB-D cameras," in *2013 IEEE Int. Conf. on Robotics and Automation*, Karlsruhe, Germany, May 2013, pp. 3748–3754.

[20] G. Casiez, N. Roussel, and D. Vogel, "1 euro filter: a simple speed-based low-pass filter for noisy input in interactive system," in *SIGCHI Conference on Human Factors in Computing Systems*, Austin, Texas, May 2012, pp. 2527–2530.

[21] V. Grabe, M. Riedel, H. H. Bülthoff, P. Robuffo Giordano, and A. Franchi, "The TeleKyb framework for a modular and extendible ROS-based quadrotor control," in *6th European Conference on Mobile Robots*, Barcelona, Spain, Sep. 2013.