Attitude and Thrust Strategies for Fully-Actuated Multirotors: 
The Fast-Track to Real-World Applications

Azarakhsh Keipour¹, Mohammadreza Mousaei², Andrew T Ashley³ and Sebastian Scherer⁴

Abstract—The introduction of fully-actuated multirotors has opened the door to new possibilities and more efficient solutions to many real-world applications. However, their integration had been slower than expected, partly due to the need for new tools to take full advantage of these robots.

As far as we know, all the groups currently working on the fully-actuated multirotors develop new full-pose (6-D) tools and methods to use their robots, which is inefficient, time-consuming, and requires many resources.

We propose methods that extend the existing flight controllers to support the new fully-actuated robots and bridge the gap between the tools already available for underactuated robots and the new fully-actuated vehicles. We introduce attitude strategies that work with the underactuated planners, controllers, tools, and remote control interfaces, all while allowing taking advantage of the full actuation. Moreover, new methods are proposed that can properly handle the limited lateral thrust suffered by many fully-actuated UAV designs. The strategies are lightweight, simple, and allow rapid integration of the available tools with these new vehicles for the fast development of new real-world applications.

The real experiments on our robots and simulations on several UAV architectures show how the strategies can be utilized. The source code of the PX4 firmware enhanced with the proposed methods and its simulator with our fully-actuated hexarotor model are provided with this paper.

I. INTRODUCTION

The past two decades have seen rapid growth in the number of multirotor applications. Traditional multirotors are designed to have co-planar rotors. Although this design choice is simple and maximizes the energy efficiency, these UAVs suffer from underactuation (i.e., their rotational motion is unavoidably coupled with their translational motion).

New designs for multirotors have been proposed in recent years that allow fully independent control over both linear and angular motions [1], [2]. These robots have less energy efficiency and generally have more complex hardware designs than the underactuated UAVs. However, in many cases, the independent control over all six dimensions greatly simplifies the payload and the required approach to a task, ultimately reducing the weight and final costs of the hardware, software, and UAV products’ development process. Moreover, it makes many new applications possible.

The integration of the new fully-actuated designs into real-life applications had been much slower than the underactuated ones. The added design complexity, the lower energy efficiency, and the additional efforts required to develop new software tools prevent their widespread use in real projects. There have been some efforts to commercialize the fully-actuated multirotor designs (e.g., [3], [4], [5]); however, their success has been minimal so far.

Over the years, many methods and tools for underactuated multirotors have been developed and are widely available to plan missions and trajectories for different purposes and to control the motion of the robots based on the planned trajectories. Most of the controllers developed for fully-actuated vehicles require full-pose trajectories and cannot interact with the tools developed for underactuated UAVs. An even more significant challenge is the need for new Remote Control (RC) interfaces that can control the robot in all the six-degrees-of-freedom, making it difficult for the pilots to learn flying or transition to flying these new robots.

On the other hand, many research groups utilize tools such as Simulink to design and generate the code for their new fully-actuated controllers. While these tools can also generate the code required for a full flight control system (such as the state estimation), the generated code is too basic for any tasks performed beyond heavily-controlled lab environments. Moreover, the code cannot be directly integrated with the existing autopilots that already have a more powerful flight stack, slowing down integrating the new robots into real applications even further.

We have realized that most of the applications do not require a completely free change in orientation and the vehicle’s attitude generally needs to follow a set of patterns. We can define a set of fully-actuated operation modes for the attitude (we refer to them as attitude strategies).
strategies can be switched during the flight to address the different needs of applications. Identifying such a set of strategies allows using the readily-available flight controllers, software tools, and RC interfaces developed for the underactuated UAVs.

Many fully-actuated UAVs, including the most common architectures, such as fixed-pitch hexarotor designs, can generate only limited lateral forces compared to the forces perpendicular to their bodies. Franchi et al. [6] have proposed to call these vehicles as LBF (vehicles with laterally bounded force), which we will also use in this paper. The operation of the LBF robots requires particular attention to handling their lateral thrust limits. We propose a set of methods (thrust strategies) that can handle the lateral thrust limits with minimal changes to the available flight controllers, allowing the use of either 6-D (full-pose) or traditional 4-D planners and motion controllers.

The proposed attitude and thrust strategies have minimum overhead and can be directly integrated into any autopilot, extending the Position Controller module.

Our contributions include:

- Proposing a method to quickly extend the already existing flight to work with the fully-actuated robots, enabling the integration of fully-actuated robots with already existing flight stacks and tools available for underactuated multirotors.
- Proposing a set of attitude strategies that allow a wide range of applications to use the full actuation for indoor and outdoor environments, and free flight and physical interaction tasks.
- Proposing a set of thrust strategies to properly handle the limited lateral thrust of the LBF vehicles depending on the application requirements.
- Providing the source code for the proposed strategies implemented on the PX4 flight controller firmware along with our simulation environment.

We show our experiments in Gazebo PX4 SITL and our MATLAB simulator as well as on the real robot to illustrate the proposed strategies and methods.

Section II discusses the minimum changes required to modify the existing underactuated flight controllers to work with the fully-actuated robots. In Section III, we propose the attitude strategies identified to take full advantage of the full actuation using the new controllers. Section IV describes the methods for properly handling the limited lateral thrust in LBF robots. Finally, in Section V, we present the real and simulated results for the methods proposed in this paper for real-world applications.

II. FLIGHT CONTROLLER ARCHITECTURE

Figure 2 illustrates the architecture for a typical flight controller for underactuated UAVs. The inputs are normally the desired yaw, the desired position, and/or the desired position derivatives. These inputs are internally utilized to generate the full desired attitude (we call it attitude setpoint) for its Attitude Controller module and the desired linear acceleration or forces (we call it thrust setpoint) for its Control Allocation (or Mixer) module.

The generation of the attitude and thrust setpoints (i.e., reference attitude and thrust) usually happens within some parent module (e.g., Position Controller). However, here, without the loss of generality, we assume that they are separate submodules. Figure 3 illustrates the Position Controller module of the PX4 flight controller with the thrust and attitude setpoint generation functions depicted as separate submodules.

Some flight controller designs may have slightly different input/output combinations. Many such differences (such as using accelerations instead of forces) do not affect the methods described in this section.

We propose that extending the existing flight controllers is possible by the following minimal set of changes:

- Modifying or replacing the Control Allocation (or mixer) module to support the new architecture and prioritize angular acceleration over the linear acceleration.
- Extending the Attitude Setpoint Generation function into an Attitude Setpoint Generator module, which allows utilizing the full actuation based on chosen strategies (see Section III).
- Extending (or adding) the Thrust Setpoint Generation function into a Thrust Setpoint Generator module, which manipulates the desired thrust to respect the thrust limits of the fully-actuated LBF vehicles (see Section IV).

Figure 4 illustrates the design of the controller we implemented for fully-actuated multirotors based on the existing PX4 autopilot architecture ([7]). The controller only modifies the Control Allocation module and extends the Thrust and Attitude Setpoint Generator modules but devises all other parts of the existing controller on PX4 autopilot.

The next two sections explain the Attitude Setpoint Generator and the Thrust Setpoint Generator modules in more detail.
Fig. 4: PX4 flight controller extended for fully-actuated multicopters. (a) The input is the desired position (and/or its derivatives) and the desired yaw, requiring an Attitude Generation module to calculate the pitch and roll. (b) The controller input is the desired 6-D pose.

Fig. 5: The Attitude Setpoint Generation module.

III. REFERENCE ATTITUDE GENERATION

The Attitude Setpoint Generator module is illustrated separately in Figure 3. Given the desired thrust force in the inertial frame ($\vec{F}_{des}$), the desired yaw ($\psi_{des}$), and the UAV’s current state $x$, the module calculates the full reference attitude ($\Phi_{sp}$) for the Attitude Controller. There are additional strategy-specific inputs (enclosed in dashed boxes), which are not present in the underactuated controllers. These inputs will be explained in their relevant strategies.

We propose a set of five attitude strategies for fully-actuated multicopters that we believe can cover a wide range of applications. These strategies are zero-tilt, full-tilt, minimum-tilt, fixed-tilt, and fixed-attitude. For each strategy, it is explained how to derive the attitude setpoint in the rotation matrix form. As a reminder, the columns of the rotation matrix are the unit vectors $\hat{I}_S$, $\hat{J}_S$ and $\hat{K}_S$, in the directions of $X_S$, $Y_S$ and $Z_S$ axes of the setpoint frame $F_S$.

We assume that the given desired thrusts have already compensated for the gravity force to achieve the desired acceleration, i.e., the thrust equal to the vehicle’s weight is already added to the upward $z$ component of the desired thrust.

Finally, we use the North-East-Down (NED) convention for the inertial frame $F_I$ and Forward-Right-Down (FRD) convention for the body-fixed $F_B$ and setpoint $F_S$ frames.

The rest of this section describes the proposed attitude strategies and the applications where each can be useful.

A. Zero-Tilt Attitude Strategy

Keeping the multirotor’s attitude horizontally during the flight can be beneficial for many situations, such as when making precise contact with the vertical surface or capturing a video using an onboard camera, eliminating the need for a gimbal.

Figure 6(a) shows a model used for the zero-tilt attitude generation strategy.

In this strategy, the direction of $\hat{Z}_S$ axis for the attitude setpoint is always the same as $\hat{Z}_I$ (i.e., $[0 \ 0 \ 1]^T$). The $\hat{X}_S$ axis is in the direction of the input desired yaw $\psi_{des}$ and $\hat{Y}_S$ axis is perpendicular to it and both lie on the horizontal plane:

$$R_{IS} = \begin{bmatrix} \cos \psi_{des} & -\sin \psi_{des} & 0 \\ \sin \psi_{des} & \cos \psi_{des} & 0 \\ 0 & 0 & 1 \end{bmatrix}$$  (1)

B. Full-Tilt Attitude Strategy

With coplanar underactuated multicopter designs, the only way for the robot to achieve control of the tilt is to tilt so that the direction of the desired thrust is normal to the plane of the rotors. This strategy is the most useful to oppose the external forces and disturbances and reduce the total energy consumption when control of the tilt is not crucial in a given task.
Figure 6(b) shows a model that demonstrates the full-tilt attitude strategy. $\vec{F}_{\text{des}}^I$ should be normal to the $\hat{X}_S\hat{Y}_S$ plane and in the opposite direction of $\hat{Z}_S$ axis. Therefore, we have

$$\vec{k}_S = -\vec{F}_{\text{des}}^I / \| \vec{F}_{\text{des}}^I \|.$$  

If an imaginary unit vector in the desired yaw direction on the $\hat{X}_S\hat{Y}_S$ plane is rotated for $+90^\circ$ around $\hat{Z}_S$ axis, it will fall on the plane made from axes $\hat{Y}_S$ and $\hat{Z}_S$. Thus, $\vec{j}_S = [-\sin \psi_{\text{des}} \cos \psi_{\text{des}} 0] \times \vec{k}_S$. Finally, we have $\vec{n}_S = \vec{k}_S \times \vec{j}_S$.

C. Minimum-Tilt Attitude Strategy

The zero-tilt strategy described in Section III-A requires a lateral thrust input that is less than the maximum lateral thrust limit $F_{\text{lmax}}$ in LBF robots. In some applications, keeping the tilt as close to zero as possible is desirable, but achieving the desired thrusts and accelerations is more important than maintaining the zero-tilt attitude. An example scenario is filming a highly dynamic target or in high-gust winds. This strategy, first described in [8], minimize the tilt of the robot by using up the lateral thrust. Figure 6(c) helps demonstrating how the minimum-tilt attitude is calculated.

If the horizontal component of $\vec{F}_{\text{des}}^I$ is less than the maximum lateral thrust $F_{\text{lmax}}$, then the zero-tilt attitude strategy can be used to calculate the attitude setpoint. Otherwise, all the available thrust on the lateral plane is used first, then the remaining required thrust determines the required tilt $\lambda_{sp}$, roll $\phi_{sp}$ and pitch $\theta_{sp}$. To calculate the tilt $\lambda_{sp}$ we have:

$$\lambda_{sp} = \arcsin \left( \frac{\| \hat{X}_S \hat{Y}_S \vec{F}_{\text{des}}^I / \| \vec{F}_{\text{des}}^I \|}{\| \vec{F}_{\text{des}}^I \|} \right) - \arcsin \left( \frac{F_{\text{lmax}}}{\| \vec{F}_{\text{des}}^I \|} \right)$$

where $\chi$ and $\mu$ angles are as illustrated in Figure 6(c) and $\hat{X}_S \hat{Y}_S \vec{F}_{\text{des}}^I$ is the projection of the desired thrust on the horizontal plane.

The axis of rotation $\vec{r}$ is perpendicular to the plane consisting of the desired thrust $\vec{F}_{\text{des}}^I$ and the inertial $\hat{Z}_I$ axis. Hence, it can be calculated as $\vec{r} = (\vec{F}_{\text{des}}^I \times \vec{k}_I) / \| \vec{F}_{\text{des}}^I \times \vec{k}_I \|$.

Finally, by rotating the $\vec{k}_I$ unit vector around $\vec{r}$ using the Rodrigues’ rotation formula we have:

$$\vec{k}_S = (1 - \cos \lambda_{sp}) (\vec{r} \cdot \vec{k}_I) \vec{r} + \vec{k}_I \cos \lambda_{sp} + (\vec{r} \times \vec{k}_I) \sin \lambda_{sp} \tag{3}$$

The unit vectors for $\hat{X}_S$ and $\hat{Y}_S$ axes can be calculated similarly to the full-tilt strategy.

D. Fixed-Tilt Attitude Strategy

Some applications require keeping a specific tilt angle for the multirotor. An example scenario is flying in constant wind when keeping a fixed tilt against the wind independent of the yaw and the movement direction is desirable to increase the thrust that remains after opposing the wind.

Figure 6(d) shows a model demonstrating the axes and angles used in the calculation of the fixed-tilt attitude strategy.

The strategy requires two additional inputs $\theta_{\text{des}}$ and $\kappa_{\text{des}}$ to the system, representing the angle of the desired tilt and the direction of the tilt, respectively (see Figure 5). We assume here that the desired direction $\kappa_{\text{des}}$ is given in the inertial frame (i.e., with respect to the north direction).

The axis of rotation $\vec{r}$ to tilt the robot is perpendicular to $\hat{Z}_I$ axis and the projection of the vector pointing in the direction of tilt on the $\hat{X}_I \hat{Y}_I$ plane. Hence, it can be computed as $\vec{r} = [\cos \kappa_{\text{des}} \sin \kappa_{\text{des}} 0] \times \vec{k}_I$.

Finally, the unit vector for the $\hat{Z}_S$ axis is calculated using Equation (3) and the $\hat{X}_S$ and $\hat{Y}_S$ axes are calculated similarly to the full-tilt strategy.

E. Fixed-Attitude Strategy

In practice, some applications require keeping specific attitude angles without explicitly including the roll and pitch angles in the trajectory. An example scenario is during the robot’s contact with the wall, when keeping a constant orientation may help control the end-effector’s pose and wrench.

The strategy requires two new inputs: the desired roll $\phi_{\text{des}}$ and the desired pitch $\theta_{\text{des}}$ (see Figure 5). The rotation matrix can be directly calculated from the given Euler angles:

$$R_{I_S} = \begin{bmatrix}
\cos \theta \cos \psi & \sin \theta \cos \psi & s \sin \psi - c \phi \\
\cos \theta \sin \psi & \sin \theta \sin \psi & s \cos \psi + c \phi \\
-\sin \theta & s \phi & c \phi
\end{bmatrix}$$

where $c$ and $s$ are shorthand for $\cos$ and $\sin$ functions, respectively, and $\phi, \theta, \psi$ are used for $\phi_{\text{des}}, \theta_{\text{des}}$ and $\psi_{\text{des}}$.

IV. THRUST SETPOINT GENERATION

The output of the Thrust Generator module in Figure 4 is expressed in $\mathbb{F}^I$ frame, while its input is in $\mathbb{F}^Z$ frame. With an unlimited available thrust, the input thrust can be simply rotated from $\mathbb{F}^Z$ to $\mathbb{F}^I$ (i.e., $\vec{F}_{\text{des}}^I$ can be projected on the current body-fixed axes) to compute the thrust setpoint: $\vec{F}_{\text{sp}} = R_{I_Z} \cdot \vec{F}_{\text{des}}^Z$.

In practice, the available thrust is not unlimited. In particular, fully-actuated LBF multirotors have even more limited lateral thrust than the thrust normal to their body. If the element of $\vec{F}_{\text{des}}^I$ on the body $\hat{X}_S \hat{Y}_S$ plane is larger than the maximum lateral thrust ($F_{\text{lmax}}$), some motors will saturate, and the system may lose its stability.

Franchi et al. [6] has proposed a method to handle the limited thrust of LBF robots. Their work targets a specific flight control architecture and ultimately requires 6-D planners and motion controllers to work. However, we propose thrust strategies that can handle the lateral thrust limits with minimal changes to the existing flight controllers, allowing both full-posel and traditional underactuated planners and motion controllers.

First, we should define the terminology:
Lateral thrust ($\vec{F}_{\text{lat}}$): The component of the thrust on the body-fixed $X_B Y_B$ plane.

Normal thrust ($\vec{F}_{\text{nor}}$): The component of the thrust on the body-fixed $Z_B$ axis.

Horizontal thrust ($\vec{F}_{\text{hor}}$): The component of the thrust on the inertial $X_I Y_I$ plane.

Vertical thrust ($\vec{F}_{\text{ver}}$): The component of the thrust on the inertial $Z_I$ axis.

Hover thrust ($\vec{F}_{\text{hov}}$): The vertical thrust required to keep the UAV hovering when no force other than gravity is acting on it.

Maximum lateral thrust ($\vec{F}_{\text{lmax}}$): The maximum achievable force on the lateral plane ($X_B Y_B$) of the UAV in the direction of the desired force. Here we assume that $\vec{F}_{\text{lmax}}$ is a constant scalar.

Given $\vec{F}_{\text{des}}$, two cases can happen:

Case 1: $\|\vec{F}_{\text{lat}}\| \leq \vec{F}_{\text{lmax}}$: Generating $\vec{F}_{\text{des}}$ is feasible.

Case 2: $\|\vec{F}_{\text{lat}}\| > \vec{F}_{\text{lmax}}$: Generating $\vec{F}_{\text{des}}$ is not feasible.

We describe two different sets of strategies with different objectives for handling the latter case. Each method has its strengths and can be used depending on the situation.

Method 1: Keeping the Desired Vertical Thrust

Merely cutting the lateral input to $\vec{F}_{\text{lmax}}$ bounds can result in losing a portion of the vertical thrust, leading to altitude tracking errors and, in extreme cases, catastrophic results. Therefore, an important objective when handling the lateral thrust can be keeping the vertical desired thrust.

First, both the vertical and horizontal components of $\vec{F}_{\text{des}}$ (i.e., $\vec{F}_{\text{ver}}$ and $\vec{F}_{\text{hor}}$) are projected on $\mathbb{F}^B$ separately to compute the partial thrust setpoint vectors $\vec{F}_{\text{ver}}^B$ and $\vec{F}_{\text{hor}}^B$.

We assume that the lateral component of $\vec{F}_{\text{des}}$ is not greater than $\vec{F}_{\text{lmax}}$, otherwise the vertical desired thrust cannot be achieved. The assumption is reasonable if the robot’s tilt is limited (the limits can be set based on the dynamics of the multirotor). After consuming the lateral thrust required for the vertical component, the remaining available lateral thrust $\vec{F}_{\text{lmax}}$ can be used to bound the horizontal thrust $\vec{F}_{\text{hor}}^B$:

$$\vec{F}_{\text{hor}}^B = \frac{\vec{F}_{\text{lmax}}}{\sqrt{(x \vec{F}_{\text{hor}}^B)^2 + (y \vec{F}_{\text{hor}}^B)^2}} \vec{F}_{\text{hor}}$$ (5)

Finally, the thrust setpoint is $\vec{F}_{\text{sp}}^B = \vec{F}_{\text{ver}}^B + \vec{F}_{\text{hor}}^B$.

A different method is explained in [8], which directly constructs the thrust setpoint by utilizing the knowledge that the maximum lateral thrust is already being fully consumed.

Method 2: Keeping the Acceleration Directions

Method 1 guarantees that the output thrust setpoint has the same vertical component as the input desired thrust if it is feasible. However, it may cause a severe reduction in the remaining lateral thrust for horizontal motion when the vertical thrust command is large. Another method for handling the lateral thrust limits reduces both the vertical and horizontal thrusts simultaneously but ensures that the accelerations’ direction is maintained.

Knowing the hover thrust $\vec{F}_{\text{hov}}$, if the $z$ component of the input desired thrust is larger than $\vec{F}_{\text{hov}}$, then the hover thrust vector $\vec{F}_{\text{hov}}^I = [0 \ 0 \ F_{\text{hov}}^I]$ is rotated to the current body-fixed frame to obtain the baseline for zero acceleration ($\vec{F}_{\text{hov}}^B$).

The lateral thrust consumed by the hover thrust is used to obtain the new lateral thrust limit $\vec{F}_{\text{lmax}}'$. The rest of the desired thrust ($\vec{F}_{\text{des}}' = \vec{F}_{\text{des}} - \vec{F}_{\text{hov}}^B$) is rotated to $\mathbb{F}^B$ to get $\vec{F}_{\text{sp}}^B$. Then the lateral thrust of $\vec{F}_{\text{sp}}^B$ is bounded to $\vec{F}_{\text{lmax}}'$ to get $\vec{F}_{\text{sp}}''$. Finally, $\vec{F}_{\text{sp}}''$ is combined with $\vec{F}_{\text{hov}}^B$ to calculate the feasible thrust setpoint $\vec{F}_{\text{sp}}$.

V. EXPERIMENTS AND RESULTS

We have tested the proposed methods in simulation and on a real robot. This section explains our hardware and software setups and the experiments.

A. Robot’s Hardware and Software

We built fixed-tilt hexarotors with all rotors rotated sideways alternatively for 30 and -30 degrees, similar to the design described in [8].

The robots’ main frame is Tarot T960 with KDE-3510XF-475 motors, KDE-UAS35HVC ESCs, and 14-inch propellers. They are equipped with a mRo Pixracer autopilot hardware, an Nvidia Jetson TX2 onboard computer, a u-Blox Neo-M8N GPS module, Futaba T10J transmitter/receiver, and a 900 MHz radio for communication with the Ground Control Station computer.

The onboard computer runs Linux Ubuntu 18.04 (Bionic Beaver) with Robot Operating System (ROS) Melodic Morenia. Depending on the task at hand, different software packages run to plan and control the missions and trajectories.

Figure 7 shows one of the UAVs we have used for our experiments.

We enhanced several recent versions of PX4 firmware to support fully-actuated vehicles and implemented the methods presented in this paper (see Figure 8). The source code for the firmware and the documentation can be accessed from https://theairlab.org/fully-actuated.

B. Simulation

Two simulation environments were devised for testing the new fully-actuated UAV development and the methods described in this paper: the Gazebo simulator with PX4 SITL and a MATLAB simulator.

The developed MATLAB simulator can be used to develop, analyze, visualize, and test different architectures.
Fig. 8: Screenshots from the simulation environments showing our fully-actuated hexarotor while running the controller and strategies proposed in this paper. (a) Gazebo with PX4 SITL making unmodeled contact with the wall. (b) MATLAB simulator writing on the wall.

The Gazebo simulator uses the SITL simulation provided by the PX4 firmware, which we have enhanced to support fully-actuated vehicles and the proposed methods and include our fully-actuated hexarotor model. It is mostly used for testing the code developed for the real robots in simulation before performing real experiments. Our PX4 firmware code’s documentation explains how to run this simulation with the new fully-actuated UAVs and switch between the strategies described in this paper.

C. Real-World Experiments

We have performed tens of flights with our fully-actuated hexarotor platform running the proposed controller and strategies. The experiments include real flights in the wind and physical contact with the environment. Figure 1 shows our UAV running the zero-tilt strategy during contact with a wall to measure the properties of the contact point using an ultrasonic sensor. The contact is unmodeled, but the UAV can keep its zero-tilt attitude and reject the disturbance. Figure 9 shows the robot keeping its zero-tilt attitude while aggressively flying and turning.

Figure 10 plots another flight segment flying with the fixed-tilt attitude strategy in a strong and gusty wind. The

VI. CONCLUSIONS

In this paper, we described a quick way to enhance the available underactuated multirotor flight controllers to support the fully-actuated multirotors while retaining the ability to interact with all the available tools and interfaces for both the traditional underactuated and the new fully-actuated robots.

We proposed a set of attitude strategies that can accept the same trajectories as the underactuated robots but fully take advantage of the 6-D actuation. We also proposed a set of thrust strategies with different objectives that can enhance the robots’ performance with limited lateral thrust force (LBF multirotors).

Finally, we illustrated our methods working in a simulator with several different architectures and on real robots, keeping the intended full pose both in free flight and during physical interaction with the environment. The real-world experiments were performed with a robot equipped with a popular open-source autopilot firmware minimally modified to work with fully-actuated UAVs and enhanced with our proposed attitude and thrust strategies.

We hope that the described methods help the researchers and the industry accelerate the integration of fully-actuated robots into real-world applications.

ACKNOWLEDGMENT

This project became possible due to the support of Near Earth Autonomy (NEA). Also, the authors would like to thank Andrew Saba and Greg Armstrong for their contributions in building the UAVs and performing the experiments.
REFERENCES

[1] R. Rashad, J. Goerres, R. Aarts, J. B. C. Engelen, and S. Stramigioli, “Fully Actuated Multirotor UAVs: A Literature Review,” IEEE Robotics & Automation Magazine, vol. 27, no. 3, pp. 97–107, sep 2020. [Online]. Available: https://ieeexplore.ieee.org/document/8978486/

[2] A. Franchi, “Platforms with Multi-directional Total Thrust,” in Aerial Robotic Manipulation: Research, Development and Applications, A. Ollero and B. Siciliano, Eds. Cham: Springer International Publishing, 2019, pp. 53–65. [Online]. Available: https://doi.org/10.1007/978-3-030-12945-3_4

[3] “Skygauge, the Drone for any Inspection,” 2020. [Online]. Available: https://skygauge.co/

[4] “CyPhy LVL 1 Drone: Reinvented for Performance and Control,” 2018. [Online]. Available: https://www.kickstarter.com/projects/1719668770/cyphy-lvl-1-drone-reinvented-for-performance-and-control

[5] “Voliro Airborne Robotics,” 2020. [Online]. Available: https://www.voliro.com/

[6] A. Franchi, R. Carli, D. Bicego, and M. Ryll, “Full-Pose Tracking Control for Aerial Robotic Systems With Laterally Bounded Input Force,” IEEE Transactions on Robotics, vol. 34, no. 2, pp. 534–541, apr 2018. [Online]. Available: http://ieeexplore.ieee.org/document/8291488/

[7] L. Meier, D. Honegger, and M. Pollefeys, “PX4: A node-based multithreaded open source robotics framework for deeply embedded platforms,” in 2015 IEEE International Conference on Robotics and Automation (ICRA). IEEE, may 2015, pp. 6235–6240. [Online]. Available: http://ieeexplore.ieee.org/document/7140074/

[8] H. Mehmood and E. N. Johnson, “A Daisy-Chain Control Design for a Multirotor UAV with Direct Force Capabilities,” in AIAA Guidance, Navigation, and Control Conference. Reston, Virginia: American Institute of Aeronautics and Astronautics, jan 2017, pp. 1–19. [Online]. Available: http://arc.aiaa.org/doi/10.2514/6.2017-1043