MarCONI - One-Hand Controller for Unmanned Aerial Vehicle

Roberto Carluccio, Alfio Messina

Abstract - Here authors present MarCONI (Multi Channel One haNd Interface), a system born to control remotely piloted aircrafts (RPAs), in particular multi-rotors, by means of new generation peripherals. Among those, used in personal computing environment, a generation of 6 degree-of-freedom (DOF) advanced controllers is the SpaceMouse family by 3Dconnexion. MarCONI is a hardware-software system, acting as a bridge between the USB peripheral and the UAV's radio-controller. A shaping block has been added to the system in order to process raw data flow generated by the SpaceMouse. This step allows the user to adapt the controller feedback to the specific vehicle features and response. Shaping parameters are fully customizable by a specific Web GUI, accessible through a Wi-Fi connection, making possible the setup tuning by means of mobile devices, such as smartphones or laptops. A side benefit of this system is the possibility to pilot UAVs using one hand only, with no restriction.

Index Terms - 3D USB HID, Arduino, Drone, FPV, Multi-rotor, RPA, SpaceMouse, SpaceNavigator, UAV.

I. INTRODUCTION

During last years unmanned aerial vehicle (UAV) systems have been subjected to enormous growth in terms of reliability, autonomous capability and precision. This is mainly due to the development of solid-state low cost inertial units (IMU) and to the increased computing power of on-board pilot processors. IMUs are electronic devices in which tri-axial accelerometers, gyroscopes and magnetometers are built with solid-state silicon low cost technology and are now capable to sense and measure with high precision minimal variations in vehicle attitude, generating streams of data at high rates, up to few hundred hertz. The huge quantity of sensor data requires a big computational power that modern processors technology can now offer and this, together with advances in control system algorithms, has led to the development of *flight controllers* (or electronic pilots) i.e. systems that can stabilise in the air and guide on paths, various propulsion flying systems, with high reliability. In this application we will mainly deal with electric multi-copter *drones*. The undoubted innovative aspect of these systems is their robotic nature: they can be easily programmed to be effectively utilized in different configurations. With modern UAVs it is possible to plan surveys on GIS systems, by setting up flight paths or waypoints in advance or even during-the-flight. The surveys can be autonomously executed by the onboard pilot, while flight and sensors data can be logged in local memory or sent to *ground control stations* through bi-directional telemetry data links.

UAV data acquisition offers an effective monitoring tool of rapidly-changing and difficult to access places, allowing a fine spatial resolution [1], with unprecedented economy and ease. Many once expensive activities can now be conducted with low budgets, from photogrammetry for digital elevation models acquisition [2][3] to critical or archaeological areas exploration [4], ending with environmental and geophysical measurements. Also fauna monitoring protocols can be developed using UAVs: a recent work shows how they can be employed for acquiring aerial images of penguin colonies [5]. UAV applications are many, with a strong upward trend. Current UAVs research is aimed towards *local intelligence deployment* and *sensor data fusion techniques* in order to enhance decisional capabilities both for single UAVs and for flock operations [6][7].

A. Drone for aerial photography

One of the most advantageous features of multi-rotor systems is the ability to *hover in the air*, independently from their speed; this ability is highly desirable for *aerial movie photographic systems*. Cinematographic techniques, as well as computer graphics simulations have accustomed us to unusual points of view, but the opportunity to realize them physically, using low-cost techniques, is an absolute innovation. Key-point for a flying camera is its *steadiness*. Several stabilisation systems were thus developed in order to decouple camera attitude from drone oscillations due to wind and air turbulence. In latest systems, camera is mounted on a special active support called *gimbal*. Camera attitude is referenced by an IMU mounted on the camera frame itself and the frame is free to move through a mechanical decoupler around three orthogonal axes served by high torque motors each. This solution allows the system to control camera attitude and dynamically compensate for any sudden drone movement [8]. In principle, camera could be operated around all three angles, but it is usually preferred to maintain an "horizontal image" roll angle and operate on pitch and yaw axes only (see Fig. 1 for angles naming).

Revised Version Manuscript Received on June 17, 2015.

Dr. Roberto Carluccio, Istituto Nazionale di Geofisica e Vulcanologia, Rome, Italy.

Mr. Alfio Messina, Istituto Nazionale di Geofisica e Vulcanologia, Rome, Italy.

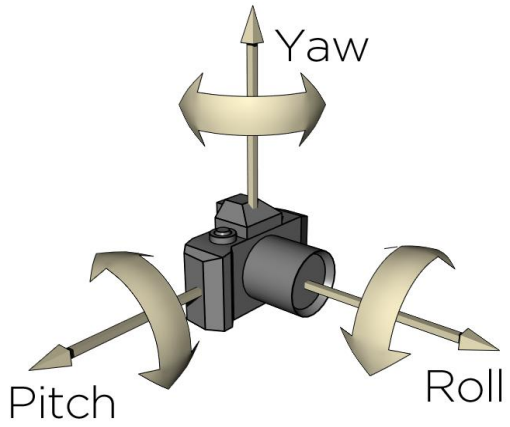

Fig. 1. Frame reference and Rotation angles naming

UAVs are typically guided through the use of radio-controllers. These devices operate over a digital link and a bi-directional protocol that allows real-time streams of numerical vector values (channels). Manual control of aircraft is achieved through two main *proportional thumb-sticks*, historically developed for fixed-wing airplanes, than can control two parameters each: a maximum of four channels can thus be continuously controlled by operator. A multi-rotor drone is much more similar to an helicopter and it needs three channels for space translations and one channel for vertical axis (yaw) rotation to be assigned to thumb-sticks. All of existing RC-radio channels standard mapping modes do require the usage of both pilot hands, not allowing easy control of on-board camera. In light of this, during aerial shots the pilot is usually helped by a cameraman operating on an independent RC-radio.

B. Space Mouse

Computer science evolution, together with advanced graphics design software, often deal with complex data structures, usually organized over *dimensions* higher than the classical 2D screen plane, requiring new effective and intuitive *human-machine interfaces*. Thus, many innovative peripherals have been developed for Computer Aided Design (CAD) environment: among those, the SpaceMouse family by 3Dconnexion [9][10]. This *Human Interface Device* (HID) help operations in virtual 3D environments, controlling simultaneously up to 6 parameters. The controller is basically a very high precision (4 µm) *tracked optical sensor* (CAP) elastically bounded to the controller body and free to move in a region around a reference position and orientation (see Fig. 2). The displacement due to operator force and torque is measured by the system and exported as translations and rotations around a orthonormal axes controller reference frame. A stream of 6-component vectors is thus generated with a refresh rate of 60 Hz and sent through a USB serial port. Assigning these 6 values to translational and angular velocities of computer generated virtual point-of-view, an extremely effective, powerful and easy-to-use interface is obtained to navigate computer generated 3D spaces.

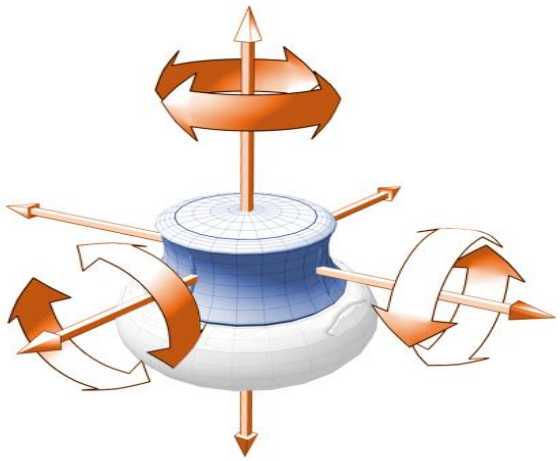

Fig. 2. Space Navigator CAP 6 degree-of-freedom

II. SYSTEM OVERVIEW

A. General Description

The idea behind *MarCONI* is to build a bridge system that sits between HID controller and RC-radio and permits the use of the *SpaceNavigator* (the most portable among SpaceMouses) to pilot the UAV, like it is done for the software virtual camera in a 3D CAD.

To achieve this goal, the bridge will have to:

1. acquire data from input unit;
2. re-process them;
3. generate a suitable signal to inject in the UAV control flying rig.

Presented system is developed around SpaceNavigator HID, but easy adaption to other HIDs is possible by proper rewrite of import (1) and process (2) software modules. To control UAV, MarCONI will generate a proper signal on *RC-radio data port* (3). This port is used by teachers during pilots training, allowing students to have control on selected UAV channels in a safe and supervised procedure. Trainer port requires the usage of pulse position modulation (PPM) standard protocol. Fig. 3 shows system logical flow: how the different parts and operations do interact and how tasks are divided on two processors.

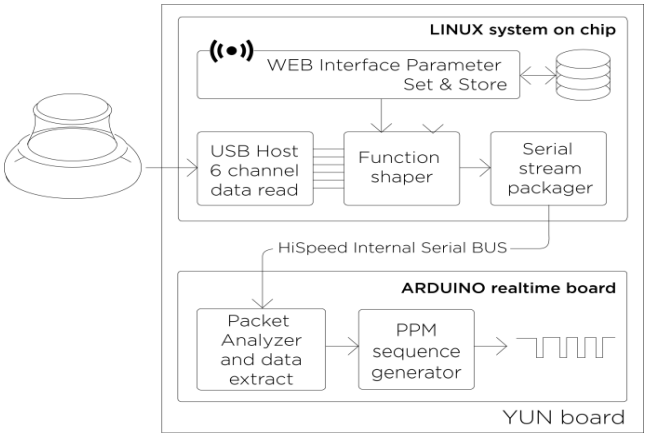

Fig. 3. Scheme of logical in three steps: 1) USB data acquisition, 2) data processing, 3) output signal

A GUI setup software allows for channels assignment and lets to use controller in different configurations. If we refer to Fig. 1 for angle naming, two configurations, among other possible, could be:

- HID controls camera only using two or three channels for yaw-pitch or full (yaw, pitch, roll) rotations; provided a sufficient number of available channels on the main RC-radio, this solutions prevents the need for a second one.
- HID controls UAV completely in first person view (FPV) operation, mapping all translations channels on UAV translations, yaw rotation channel on UAV yaw rotation and pitch channel on gimbal pitch control. Yaw angle is fixed towards forward view, for FPV, while yaw and roll are fully stabilized by the gimbal system.

B. Hardware

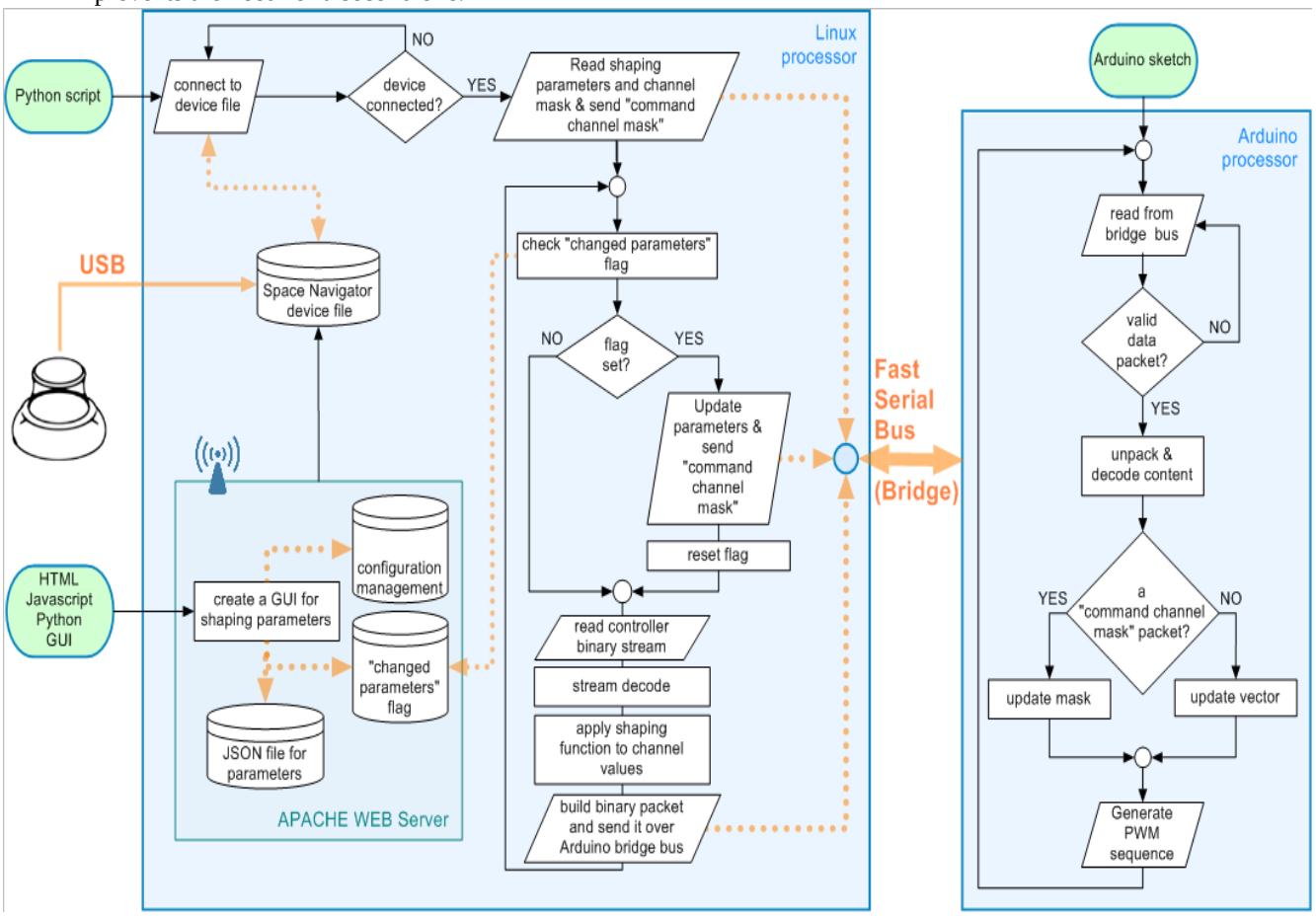

Fig. 4. Modular software system scheme

This project requires a flexible hardware-software environment; we need high level functions to expose GUI parameters to user, math computational capabilities, a good processing power and low level USB stack access, together with a real-time processor system for time critical routines execution. Our choice among the many commercial general-purpose boards has been the *Arduino Yún* [11], a dual microcontroller board based on low-power CMOS 8-bit microcontroller *ATmega32u4* and *Atheros AR9331*, which includes a 32-bit MIPS 24K RISC processor. The former allows execution of time-critical code while the latter, on the other hand, supports a Linux distribution based on *OpenWrt*. Built-in *Ethernet* and *WiFi* network interfaces as well as the *USB host port* and the *micro-SD card slot* are directly accessible from Linux OS. The two processors communicate through a *high speed internal serial bus*.

C. Software

Our project is based on a modular software system specially designed to exploit the power and the features provided by Arduino Yún. The Wi-Fi interface can be configured to make Yún act as a *Wi-Fi access point*; we developed a complete *Web GUI* for MarCONI configuration, in order to allow users to change system behaviour using their laptop, tablet or smartphone. The *USB host port* is accessible through Linux driver, which makes it easy to manage HIDs. Moreover, MIPS 24K processor is powerful enough to correctly read the incoming binary strings from the peripheral and apply a proper *shaping function* on the retrieved values. A specific *Python application*, which runs on Linux OS, has been written to perform the above-mentioned tasks. Finally, thanks to ATmega32u4 microcontroller real-time capabilities, an *Arduino sketch* has been written to generate a *PPM signal* for the RC-radio input. A detailed scheme, in Fig. 4, shows how these software modules interact each other.

III. CONTROLLER RESPONSE TUNING

Before forwarding vector numbers to the multichannel PPM pulse forming routine,

a proper shaping is applied to the raw HID signal. Fig. 5 shows the shape transfer function and highlights shaping parameters. This step is needed mainly to model the *system sensitivity behaviour* with respect to UAV response.

Threshold and *Exponential* parameters do control the sensitivity near the CAP rest position in order to minimize unwanted channels interactions while maintaining a good control over tiny corrections. *Gain* fixes the maximum channel value that is sent to the UAV electronic pilot. The *Sensitivity* parameter actually shrinks the controller dynamical range: it allows to reach full-scale output before controller raw input maximum. Even if this feature seems to be a limitation, the parameter has been added to shaping chain after a thorough analysis of controller behavior. Let's think controller as in a *6-dimensional configurations space*, where any CAP position is represented by a single point. Since for any channel a maximum value of V_{max} can be reached, a perfect controller should explore, in its movements, an entire *hyper-cube* of side $2V_{max}$. It has however been found that CAP mechanics has some limitations: it is not actually possible for instance, to have positions giving out maximum values in rotations and translations at the same time: the hyper-space portion, S, accessible to the controller is thus not an hypercube but a complex, smaller region. To represent S we have acquired ~10k random CAP positions trying to map all accessible CAP positions (6-tuples). Fig. 6 is a matrix of 2-dim S projections scatter plots on all possible channels pairs subspaces, showing how far from the ideal case is the actual controller mechanics. Full dynamic channels region is thus a smaller (around one third of max value) hyper-parallelepiped region achievable adjusting sensitivity parameters.

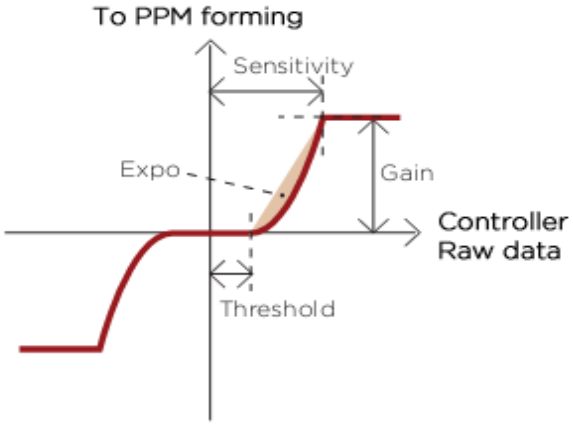

Fig. 5. Shaping function and related parameters, x-axis is raw controller data, y-axis shaped data

The shaping function is defined as:

$$f(x) = t_1 + t_2 * sgn(x) * \left(offs + gain * \begin{cases} \text{minx} - \text{thrssens} - \text{thrsexpo}, 1 & if\ x > thrs0 \\ if\ x \leq thrs \end{cases} \right) \quad (1)$$

where t_1 is the duration, in milliseconds, related to the 0 value in PPM standard, t_2 the maximum duration, *offs* indicating the offset applied to the curve, *gain* the maximum output amplitude, *x* the value generated by the HID and normalized within the interval [-1, +1], *sgn(x)* the sign function applied on x value, *thrs* the threshold value, *sens* the sensitivity and *expo* the exponential function coefficient.

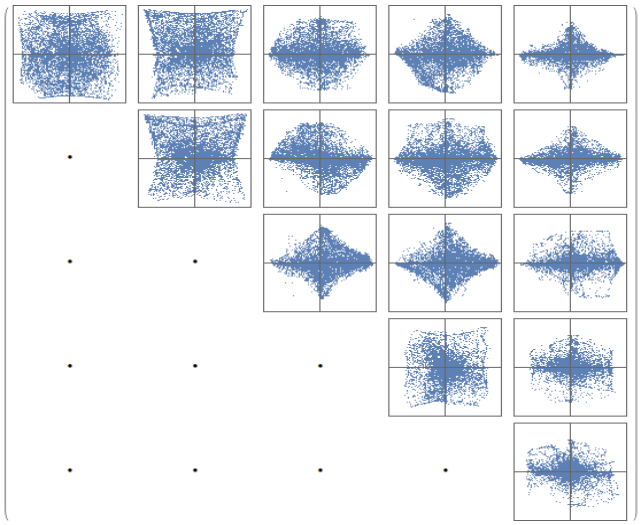

Fig. 6. Controller accessible configurations 6-space, shown as 2-subspaces projections

IV. CONFIGURATION WEB GUI

Exploiting AR9331 chipset access point features, a Web GUI has been designed to configure the system behaviour (Fig. 7). Any mobile, Wi-Fi capable, device can be used to access configuration pages, simply connecting it to the network generated by MarCONI. GUI source code exploits a mix of *HTML*, *Javascript* and *Python* languages. In particular, *jQuery UI* [12], based on jQuery Javascript library, has been integrated, providing a pool of modern widgets. Python language is widely employed inside web servers for dynamic pages creation. We used it for tasks requiring access to *Linux commands* and *filesystem*. The operator will be presented with four different forms: the *Global Pars Form*, the *Channel Form*, the *Configuration Manager Form* and the *Hardware Status Form*.

- *Global Pars Form* is the 6th tab of the GUI (Fig. 7a); it allows to create the mapping between SpaceNavigator and RC-radio's channels. Even if, each radio-controller can have data port configuration functions, nevertheless it would be preferable for the operator not to modify his radio settings and use MarCONI's configuration GUI instead. For each and every channel it is possible to assign a fixed data value offset if necessary. Lower and Upper bounds related to pause phase between two PPM pulses (expressed as milliseconds) have to be set up in the dedicated input fields present in this tab.

- *Channel Form* contains slider controls relative to Sensitivity, Threshold, Exponential and Gain parameters, as well as two check-boxes for channel Inversion and Integration (Fig. 7b). GUI has six Channel tabs, one for each SpaceNavigator DOF: translations *Left-Right*, *Front-Rear*, *Up-Down* and rotations: *Roll*, *Pitch* and *Heading*.
- *Configuration Manager Form* can be accessed from any GUI page. It provides the user with a complete MarCONI configurations management, i.e. storage, loading and deletion of files from internal memory (Fig. 7c). Using these functions, user can keep a configuration file for each radio-controller and/or multi-rotor setup.
- *Hardware Status Form* provides user with information on both HID connection status and Python script running status issuing proper color warnings (Fig. 7d). Status queries are implemented in Linux standard commands launched by Python. A *Restart* button has been included in order to launch again Python script.

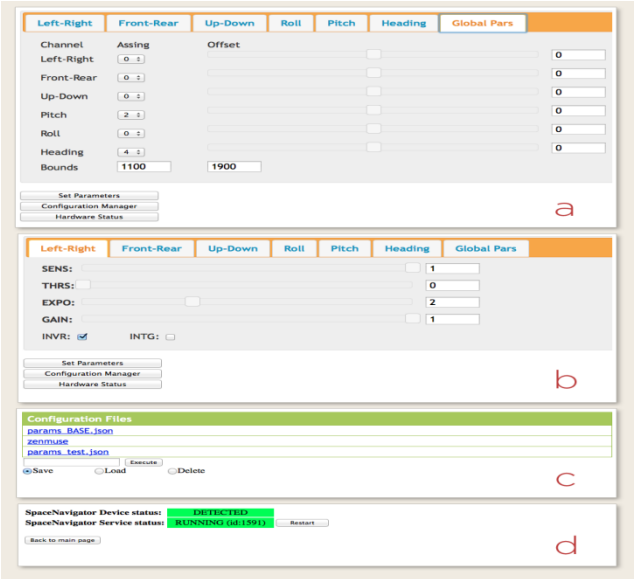

Fig. 7. MarCONI's GUI for parameters setting

V. CONTROLLER RECEIVING AND PROCESSING

SpaceNavigator generates a flow of 14 bytes binary strings. Scheme 2 shows string format, i.e. a double series of three 16-bit integer values. Each series is preceded by a byte whose value can be 1,2 or 3 indicating a translation triplet, a rotation triplet or a button operation respectively.

B | LH | LH | LH | B | LH | LH | LH (2)

where *B* is one byte and *LH* is a 16-bit value, in low(*L*) high(*H*) order. Communication between SpaceNavigator and MarCONI is handled through Linux. This OS makes possible to access the *raw data* generated by a HID-USB device (as well as a Bluetooth one), using *HIDRAW* driver. It provides a *common interface* to HID peripherals, which can be exploited by a user application to access the data stream produced by the device. As result, the USB device can be handled as any other binary file on the filesystem. Python script is launched during the *Linux boot phase*, when MarCONI is connected to the power supply. The script reads and decodes SpaceNavigator stream flow and finally applies the shaping function described above (1). Real-time user changes on shaping parameters must be immediately applied by the system. During polling, at SpaceNavigator read cycle frequency, the script checks the status of a specific *flag* set up by the GUI. Its activation indicates a configuration change and starts a new reading of the *JSON file* containing upgraded shaping parameters. Since a 8-channel PPM sequence is expected on RC-radio data port, a *remapping mechanism* is implemented in Arduino RISC processor. To this purpose, a *channel mask* vector do exist and contains the mapping of the 6 HID channels on the 8 radio-controller channels. Changes on channel mask are immediately stored in the Linux filesystem, causing also the transmission of the new mapping vector to the Arduino subsystem, where it resides in the processor volatile memory. Summarizing, each 6-tuple containing HID translations and rotations is shaped according to (1) and real-time transmitted, along with channel mask occasional updates, on the *fast serial bus* connecting the two processors. These 6-tuples and channel mask vector are used inside Arduino to build proper 8-tuples from which multi-channel PPM signal is then generated.

VI. RC PPM GENERATION

A specific sketch has been developed for Arduino system running on ATmega32u4. This sketch receives the incoming binary strings previously generated by the Python script. Arduino is then responsible of *PPM signal generation* to be sent to the radio-controller. The input binary flow is organized as reported in Scheme 3, i.e. a sequence of six 16-bit integers, followed by 3 control characters.

LH | LH | LH | LH | LH | LH | ESC | CR | LF (3)

where *ESC* is the escape character (27 in ASCII), *CR* is the carriage return (13) and *LF* is the line feed (10). Once a complete string is decoded, the sketch performs a check on the second byte. It is a special byte, indicating if the received packet is a normal *data sequence* or a *control sequence*. The latter has been introduced in order to manage random changes in the transmission scheme. Currently, it is only used for transmission of a new channel mask. The arrival of a new data sequence updates *PPM array*. It is a six integer values vector of channel intensities, normalized within the PPM range and expressed in milliseconds (see *Bounds* in Fig. 7a). *PPM signal generation* requires high precision in terms of timing. This time-critical task is carried out by using *ArduinoRCLib* open-source library [13]. The signal (square wave, 5 Vpp) is sent to the radio-controller's *trainer port*. An output PPM signal plot, obtained from digital oscilloscope data capture, is shown in Fig. 8 together with info showing *channel pulses phase measurements*. In order to get a simple and intuitive feedback from MarCONI, a digital output pin is held in HIGH condition during all binary string decoding procedure, setting it to LOW only when string is recognized as valid.

This signal feeds a LED on MarCONI's panel and gives immediate feeling of proper system behaviour. Transmissions errors lead to a steady intense light, while problems to the Python script operation result in no LED light during HID usage. A fast, flickering illumination during SpaceNavigator CAP displacement, with no time delays means that system is working well.

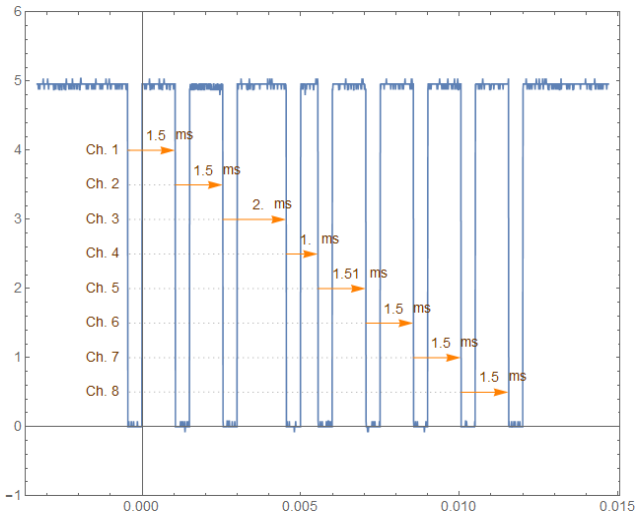

Fig. 8. Plot of output PPM signal obtained from Digital Oscilloscope captured data. Arrows show channels phases together with measured values (in milliseconds)

VII. CONCLUSIONS

The idea behind MarCONI's development is to overcome the limits of traditional control interfaces for RPAs. Thumb-sticks are useful and precise for flight control, but definitely unsuitable for aerial filming. The latter is becoming a very popular activity, taking advantage of cost breakdown made possible by UAVs usage. This situation brings to the acceleration of research and development of new technologies as well as new piloting techniques, for UAVs. MarCONI's basic idea is to make simpler and more intuitive multi-rotor control, increasing, at the same time, the number of channels that can be simultaneously used by the operator, who is able to control the drone movements as well as camera rotations autonomously. MarCONI allows a complete control using *one hand only*. Therefore, the pilot could control other instruments by the other hand, carrying out other activities during the flight. Moreover, the usability of multi-rotors is not tied to the usage of both hands, as is usually assumed for classical control interfaces, which require a minimum level of coordination. The latter could be challenging or impossible for people with physical impairments. A recent study showed as 3D controllers (also called 3D mice), such as the SpaceNavigator, can represent a valid interface alternative for these people [14].

REFERENCES

1. A. Puttock, A. M. Cunliffe, K. Anderson, R. E. Brazier, "Aerial photography collected with a multirotor drone reveals impact of eurasian beaver reintroduction on ecosystem structure", in *Journal of Unmanned Vehicle Systems*, 2015, doi:10.1139/juvs-2015-0005.
2. S. Ullman, "The interpretation of structure from motion", in *Proc. R. Soc. London*, Ser. B 203, 1979, pp. 405-426, doi:10.1098/rspb.1979.0006.
3. S. Seitz, B. Curless, J. Diebel, D. Scharstein, R. Szeliski, "A comparison and evaluation of multi-view stereo reconstruction algorithms", in *Computer Vision and Pattern Recognition*, IEEE Computer Society Conference, 2006, pp. 519 - 528. doi:10.1109/CVPR.2006.19.
4. J. Rudolf, K. Lehmann, K. Z. Smithson, T. Prinz, "Making the invisible visible: Using uas-based high-resolution color-infrared imagery to identify buried medieval monastery walls", in *Journal of Unmanned Vehicle Systems*, 2014, doi:10.1139/juvs-2014-0017.
5. N. Ratcliffe, D. Guihen, J. Robst, S. Crofts, A. Stanworth, P. Enderlein, "A protocol for the aerial survey of penguin colonies using uavs", in *Journal of Unmanned Vehicle Systems*, 2015, doi:10.1139/juvs-2015-0006.
6. G. Vásárhelyi, C. Virágh, G. Somorjai, N. Tarcai, T. Szörényi, T. Nepusz, T. Vicsek, in "Outdoor flocking and formation flight with autonomous aerial robots", in *IEEE IROS Conference*, 2014.
7. D. Habib, H. Jamal, S. A. Khan, "Employing multiple unmanned aerial vehicles for co-operative path planning", in *Int J Adv Robot Syst.*, 2013, doi:10.5772/56286.
8. C. Zych, A. Wrońska-Zych, J. Dudczyk, A. Kawalec, "A correction in feedback loop applied to two-axis gimbal stabilization", in *Bulletin of the Polish Academy of Sciences Technical Sciences*, 2015, doi:10.1515/bpasts-2015-0025.
9. 3Dconnexion SpaceMouse Family Overview, url: http://www.3dconnexion.co.uk/products/spacemouse.html.
10. T. A. Group, "The economic payback of 3d mice for cad design engineers", Tech. rep., 2008. url: http://www.3dconnexion.com/fileadmin/user_upload/manuals_docs/english_intl/3dx_whitepaper_cadpayback_en_intl.pdf
11. Arduino Yún Product Overview, url: http://www.arduino.cc/en/Main/ArduinoBoardYun.
12. jQuery UI - Official Website, url: https://jqueryui.com/.
13. Arduinorclib - Library for Arduino Based R/C Equipment, url: http://sourceforge.net/projects/arduinorclib/.
14. M. Martins, A. Cunha, I. Oliveira, L. Morgado, "Usability test of 3dconnexion 3d mice versus keyboard + mouse in second life undertaken by people with motor disabilities due to medullary lesions", in *Universal Access in the Information Society* 14, 1, 2015, 5-16. doi:10.1007/s10209-013-0329-9.

AUTHORS PROFILE

Dr. Roberto Carluccio, Degree in Physics in 1989. Research experience in various research institutes on different topics: nanotechnologies, solid state electronics and semiconductor spectroscopy techniques. From early 2004 on he is part of INGV (National Institute of Geophysics and Volcanology) Geomagnetism and Aeronomy group, participating to numerous geophisical airborne and ground based measurements campaigns . Parallel to this activity is the work on software applications development and scientifical data processing: data decoding systems, acquisition and elaboration of geodata from worldwide netwoks, expert systems and AI techniques for knowledge management, events alerts and discrimination, immersive augmented and virtual reality environment development. In latest years works on UAV systems to develop remote sensing techniques, mainly for geophysical purposes. http://www.researchgate.net/profile/Roberto_Carluccio.

Mr. Alfio Messina, Bachelor's degree in Computer Engineering, confirmed by Università degli Studi di Catania, in 2011. He is part of Istituto Nazionale di Geofisica e Vulcanologia (INGV) since 2006. His main work activities have been carried out in the context of engineering and development of software for desktop PCs, ARM devices, Android devices and microcontroller boards. Other followed research lines concerned unsupervised classification of numerical pattern, with particular application to seismo-volcanic signals. Latest activities are related to virtual reality (Oculus Rift) and UAVs (multi-rotors) applications. A complete publication list can be retrieved at the following url: https://www.researchgate.net/profile/Alfio_Messina.