UNSYS digital

Journal of Unmanned System Technology

Implementation of a PID-Controller on a Hardware Quadcopter System using dSPACE Platform

Sumaila E. Musa

Department of Control and Instrumentation, University of Derby, United Kingdom.

Abstract— Whilst different techniques have been explored for designing quadcopters, like other helicopters, the onus of navigating quadcopters rest upon using an efficient controller for controlling all its available DOF. Designing a rugged and robust universal control-system for quadcopter, in the face of environmental disturbances and uncertainties, are challenging task for control-engineer.

It is universally agreed that simulation and software models are not exact representation of physical systems and their behaviors. This paper goes behind modeling and simulations of quadcopters and it controller; it present a real-time hardware demonstration of a quadcopter and it controller. In this paper a quadcopter and a PID-controller were modeled mathematically and designed in MATLAB/ SIMULINK. A hardware quadcopter prototype was also constructed. A real-time implementation of the PID-Control algorithm on the hardware Quadcopter by dSpace1104 R&D controller-board; via the quadcopter Simulink model

Keywords— dSPACE1104 R&D Controller-board, PID Algorithm, Quadcopter, Reference-tracking.

DOI: 10.21535/just.v5i3.993

I. INTRODUCTION

QUADCOPTERS are not a new concept. History of their designs and development dated back to early twentieth century and a summary of these are presented by [11].

With the growing demands for covert surveillance by government enforcing agencies, military and even aerial photography by civilians, there is high demands for unmanned robots (such as quadcopters-drone) that are well apt for this purpose and capable of functioning effectively in various environmental terrains.

While academic works on quadcopters and their controllers are based on software simulations using CAE packages like Matlab, OrCad, Labview, *etc*. It is important to point that this software models cannot directly interface with real-time hardware devices on which these models are based on. Programming these models onto a microcontroller that can be installed on a hardware quadcopters and controllers is great platform for interfacing software models with hardware devices. But this approach is cumbersome, frustrating in terms of programming, prone to program syntax errors, *etc*. dSpace, another approach, is a software-hardware platform that not only allows the direct interfacing of software models (developed using Matlab Simulink) with hardware devices, it eradicates the problem encountered via the aforementioned microcontroller approach by generating a complete C-program codes of the models. This code can now be compiled onto a microcontroller with no or little modifications to suit ones need. Models as well as the hardware in question are central to the operation of the dSpace platform.

Corresponding author: Sumaila E. Musa
(e-mail: sumailaesq@yahoo.com)

This paper was submitted on November 29, 2017;
and accepted on December 30, 2017.

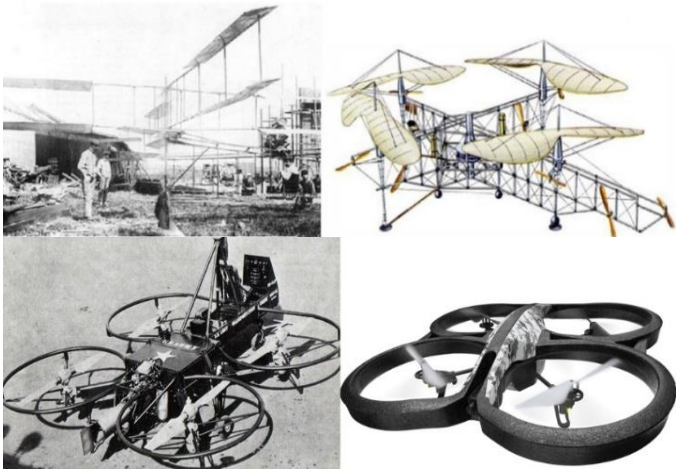

Figure 1 Quadcopter historic evolution [10][11]:
(a) Brequet-Richet Gyroplane (1907), (b) Oehmichen No2 (1920), (c) Curtiss-Wright VZ-7 helicopter, (d) Parrot AR Drone (2014)

The paper discusses on using dSpace DS1104 controller board to interface a software quadcopter & PID-controller model developed in Simulink with a hardware quadcopter. As well as using the software controller model to manipulate and control the hardware quadcopter by sending control signal via a PWM electronic generator to the ESC of the motors.

II. DSPACE HARDWARE-SOFTWARE INTERFACING

dSpace DS1104 controller-board, used for this paper, is an integrated software-and-hardware platform fully programmed from MATLAB environment and allows interfacing of Simulink models with hardware devices in real-time (Figure 4 shows dSpace toolsets incorporated in Simulink for achieving its herein described functionality [3]). The dSpace platform

enables the development of RCP (Rapid Control Prototyping) and HIL (Hardware-in-the-Loop) experiments by taking advantage of MATLAB-Simulink high-level functions – this allows user to design digital controller by simply drawing block diagram using Simulink graphical interface [3][6][12].

dSpace-RTI is the executing-feature of the dSPACE-platform that executes real-time simulation of a hardware device via Simulink; it initiates the compiling and building of the Simulink-model into SDF-file, which is used by the DS1104 compiler for carrying real-time simulation of the hardware-device based on the developed Simulink-model. The SDF-file is a C-code representation of the Simulink-model. (RTI-tools are configured and installed as a Simulink tool on the computer installed with the dSpace PC-card) [3][6].

The essential aspect of the real-time implementation comprises of converting the digital control-signal from the Simulink control-system to varying analog-signal based on the control function intended to perform on the hardware-device. Reading analog-signal (such as feedback-signal from sensors) and converting this analog-signal into digital control-signal(s) for use by the Simulink control-system. The aforementioned attributes of the controller-board are achieved via it digital and analog channels.

Our real-time implementation of using a software control-algorithm for controlling a hardware quadcopter in real-time is divided into sections:

- Software-side: which describes the mathematical and CAE modeling of the quadcopter and the PID-controller.
- Hardware-side: which presents the hardware quadcopter, power supply & PWM generator DS1104 for interfacing the software-side and the hardware-side together.

Our software-hardware architectural implementation used for this paper is summarized in Figure 3.

General Notes:

The dSpace DS1104 platform used in this paper, an ADC (or DAC) Simulink block is multiplexed with 4 channels. Scaling of acquired or transmitted signal occurs in each channel and hence each DAC or ADC; the real-time physical input signal ranges between –10V to + 10V.

- The unit has a built-in gain of 10 for analog outputs. Whatever signal is sent from Simulink to a dSPACE DAC block will be multiplied by 10. To compensate for this, we use a gain of 1/10 in front of the DAC block.
- The unit has a built-in gain of 0.1 for analog inputs. Whatever signal is read in Simulink from a dSPACE ADC block will be multiplied by 0.1. To compensate for this, we use a gain of 10 at the output of an ADC block.

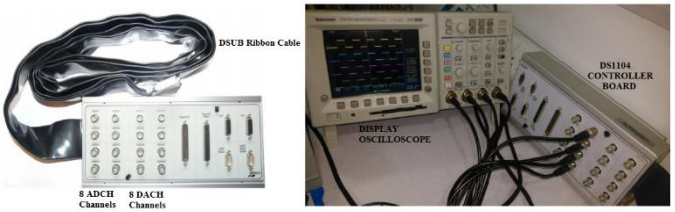

Figure 2 dSPACE R&D Controller-Board and Oscilloscope

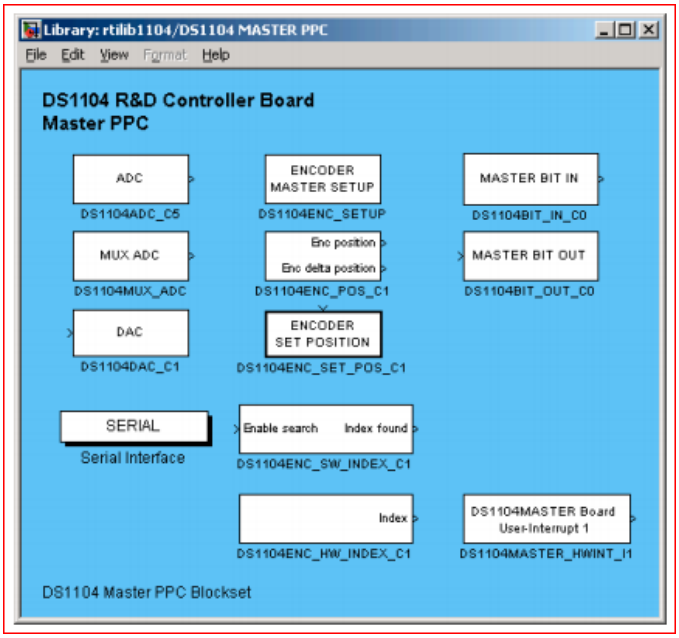

Figure 3 dSPACE RTI toolbox in Simulink

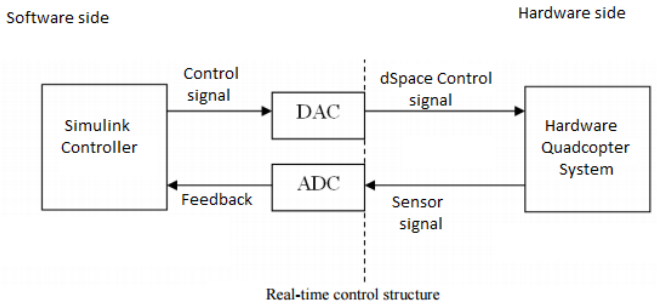

Figure 4 General Block-diagram of software-hardware interfacing using dSpace DS1104

III. SOFTWARE

A. Quacopter model and analysis

Modeling details of our quadcopter, including its equations-of-motion, control-input design, rotor modeling, *etc.* used for this paper can be found in our previous paper [10] as well as [1] [5][7].

Our helicopter model is based on cross-configuration as shown in Figure 5. Associated equation-of-motions, mathematical and SIMULINK modeling of the quadcopter is briefly expressed in this section.

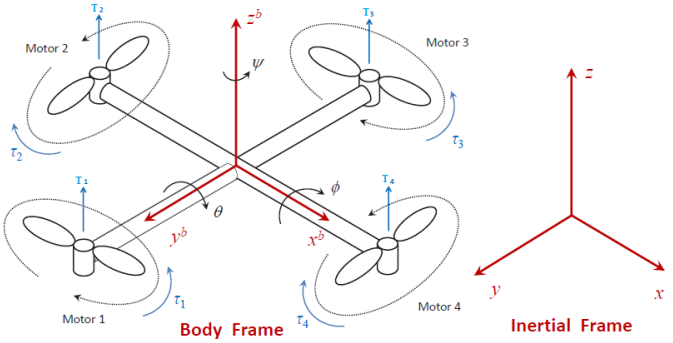

Figure 5 Quadcopter spatial free-body representation

Summaries of the prominent 6 equations used for modeling the quadcopter Simulink design are as the following. (These equations are derived from research work by [5][7][10] and subsequent coursework carried-out at the university of Derby.)

The equation for the roll subsystem was deduced as:

$$\ddot{\phi} = \frac{2\rho A l}{I_{xx}}\left(\frac{f\eta K_t}{K_q}\right)^2 (V_2^2 - V_4^2) \quad \textbf{(1)}$$

The equation for the pitch subsystem was deduced as:

$$\ddot{\theta} = \frac{2\rho A l}{I_{yy}}\left(\frac{f\eta K_t}{K_q}\right)^2 (V_3^2 - V_1^2) \quad \textbf{(2)}$$

The equation for the yaw subsystem was deduced as:

$$\ddot{\psi} = \frac{J}{I_{zz}}(\dot{\Omega}_1 + \dot{\Omega}_3 - \dot{\Omega}_2 - \dot{\Omega}_4) + \frac{D}{I_{zz}}(\Omega_1^2 + \Omega_3^2 - \Omega_2^2 - \Omega_4^2) \quad \textbf{(3)}$$

The equation for the altitude subsystem was deduced as:

$$\ddot{z} = \frac{2\rho A}{m}\left(\frac{f\eta K_t}{K_q}\right)^2 (V_1^2 + V_2^2 + V_3^2 + V_4^2)\cos\phi\cos\theta - g \quad \textbf{(4)}$$

The equation for the X-axis subsystem was deduced as:

$$\ddot{x} = \frac{2\rho A}{m}\left(\frac{f\eta K_t}{K_q}\right)^2 (V_1^2 + V_2^2 + V_3^2 + V_4^2)(\cos\phi\cos\theta\cos\psi + \sin\phi\sin\psi) \quad \textbf{(5)}$$

The equation for the Y-axis subsystem was deduced as:

$$\ddot{y} = \frac{2\rho A}{m}\left(\frac{f\eta K_t}{K_q}\right)^2 (V_1^2 + V_2^2 + V_3^2 + V_4^2)(\sin\phi\sin\theta\cos\psi - \cos\phi\sin\psi) \quad \textbf{(6)}$$

A denotes cross-sectional area of the helicopter arm.

Using the model's developed, the Quadcopter is defined by four control inputs, which are:

- Vertical thrust (z-axis)

$$U_1 = V_1 + V_2 + V_3 + V_4 \quad \textbf{(7)}$$

- Rolling moment (y-axis)

$$U_2 = V_4 - V_2 \quad \textbf{(8)}$$

- Pitching moment (x-axis)

$$U_3 = V_1 - V_3 \quad \textbf{(9)}$$

- Yawing moment

$$U_4 = V_1 - V_2 + V_3 - V_4 \quad \textbf{(10)}$$

This combination is expressed in matrix-form as:

$$\begin{bmatrix} V_1 \\ V_2 \\ V_3 \\ V_4 \end{bmatrix} = \begin{bmatrix} 0.25 & 0 & 0.5 & 0.25 \\ 0.25 & -0.5 & 0 & -0.25 \\ 0.25 & 0 & -0.5 & 0.25 \\ 0.25 & 0.5 & 0 & -0.25 \end{bmatrix} \begin{bmatrix} V_1 + V_2 + V_3 + V_4 \\ V_4 - V_2 \\ V_1 - V_3 \\ V_1 - V_2 + V_3 - V_4 \end{bmatrix} \quad \textbf{(11)}$$

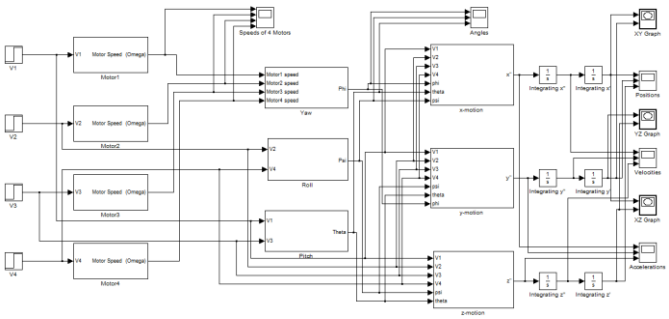

Figure 6 MATLAB/ Simulink micro-quadcopter model with its subsystems

The Simulink model of this voltage manipulating system is represented as follows:

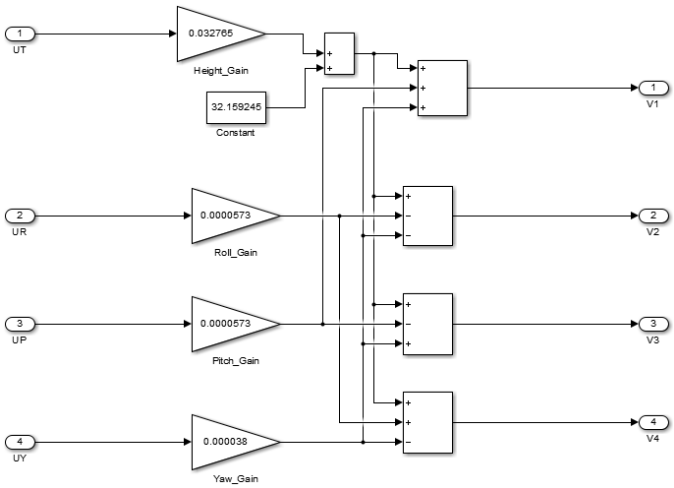

Figure 7 Voltage Splitter subsystem

First-order Taylor-Series approximation was used to linearize the non-linear quadcopter model developed via the mathematical equations. Details of our linearization approach and results is herein attached to this paper and can be found in our previous paper [10] with salient aspects of the linearized equation for VTOL is expressed as follows:

$$\ddot{z} = \frac{2\rho A}{m}\left(\frac{f\eta K_t}{K_q}\right)^2 (V_1^2 + V_2^2 + V_3^2 + V_4^2)\cos\phi\cos\theta - g \quad \textbf{(4)}$$

with

$$\phi = \theta = \psi = \dot{\phi} = \dot{\theta} = \dot{\psi} = \ddot{\phi} = \ddot{\theta} = \ddot{\psi} = 0$$
$$\Omega = \Omega_{\mathrm{h}}$$
$$\dot{\Omega} = \ddot{\Omega} = 0$$
$$\dot{x} = \ddot{x} = \dot{y} = \ddot{y} = \dot{z} = \ddot{z} = 0$$

Substituting the constants values of our quadrotor and the conditions listed above, the resulting equation becomes:

$$0 = \left(\frac{2.2 \cdot 0.08042}{0.65}\right)\left(\frac{0.005 \cdot 0.75}{0.0056}\right)^2 (V_1^2 + V_2^2 + V_3^2 + V_4^2)\cos 0 \cos 0 - 9.81$$

$$0 = (0.27219)(0.44842)(V_1^2 + V_2^2 + V_3^2 + V_4^2)\cos 0 \cos 0 - 9.81$$

$$0 = (0.12206)(V_1^2 + V_2^2 + V_3^2 + V_4^2)\cos 0 \cos 0 - 9.81$$

$$V_{\mathrm{h}} = V_1 = V_2 = V_3 = V_4$$

$$0 = (0.12206)(4V_h^2)\cos 0 \cos 0 - 9.81$$

$$V_{\mathrm{h}} = \sqrt{20.0933}$$

$$V_{\mathrm{h}} = 4.48 \text{ volts}$$

We assumed that all the rotors acts exactly in similar manner and the quadcopter's design symmetric; hence their control voltage at operating point are thought to be equal.

B. *PID controller modelling*

Classical PID-algorithm is depicted in Figure 8 is used for this paper. Generally, the PID algorithm is a feedback control-loop mechanism that continuously calculates an error-value difference between a desired set-point and a measured variable and applying controlling action by compensating for the error-difference; the controller attempts to minimize the error over time by adjusting the control variable (such as voltage supplied to the Quadcopter) to a new value determined by a weighted sum of its control-law given as [2][4][8]:

$$u(t) = K_{\mathrm{P}}e(t) + K_{\mathrm{I}}\int_0^t e(\tau)d\tau + K_{\mathrm{D}}\frac{de(t)}{dt} \qquad (12)$$

K_{P}, K_{I}, and K_{D} are all non-negative and denotes the coefficients for the proportional, integral and derivative terms respectively [3].

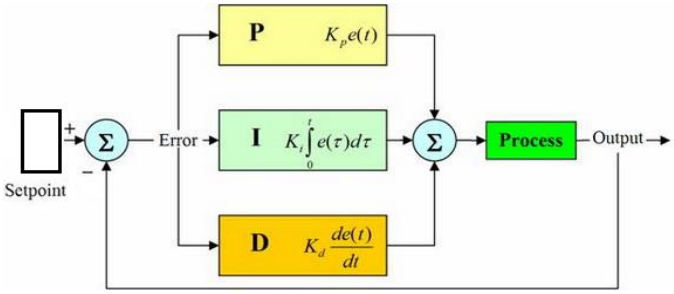

Figure 8 Classical PID control-algorithm [4]

Several PID tuning-techniques were investigated in the development of the PID controller used for the project of this thesis. Table 1 summarizes the investigated techniques:

Table 1 Investigated PID tuning-techniques

Method	Advantages	Disadvantages
Manual-tuning	No maths required; can be perform online	Requires personal experience
Ziegler-Nichols	Proven method; can be performed online	Based on trial-and-error, leads to process upset and very aggressive tuning
Tyreus Luyben	Proven method; can be performed online	Based on trial-and-error, leads to process upset and very aggressive tuning
Software tools	Consistent online or offline tuning; can employ automated computer control system design; can support non-steady-state tuning.	Requires capital and involved training
Cohen-Coon	Produces good process models	Requires some maths, only good for first-order processes
Astrom-Hagglund	Can be used for auto-tuning	The process itself is inherently oscillatory

Due to the simplicity of application, ease of use and it relative ability to perform real-time tuning while the system is online manual Ziegler-Nichols tuning techniques was employed for tuning the control-algorithm. Effects of this are summarized in Table 2.

Table 2 PID parameters summary

Parameter	Rise-time	Overshoot	Settling time	Steady State Error	Stability
K_p	Decrease	Increase	Small Change	Decrease	Decrease
K_i	Decrease	Increase	Increase	Eliminate	Decrease
K_d	Minor changes	Decrease	Decrease	No effect	Improve if Kd is small

IV. HARDWARE SIDE

A. *Quadcopter hardware*

The hardware quadcopter developed for this paper is shown in Figure 9 and comprises of 4 motors, each an ESC that control the speed and operation of the connected motor. Interfacing of this with the PID-controller is via a PWM subsystem attached to the dSpace DS1104 board. Block diagram representation of this software-hardware interfacing is shown in Figure 10.

DS1104 digital channel 1-4 (DACH1–DACH4) were used to interface the hardware quadcopter model via PWM generator - DACHs are digital-to-analog channels which convert digital control signal from the controller to analog signal necessary to generated the required angular velocity. Similarly, ADCHs are the complementary analog-to-digital channels.

B. *Sources and PWM generator*

As shown in Figure 9, our set-up comprises of four motors, which are labeled as shown on to a wooden platform, power-supply to the motors are supplied by a four-channel PWM-generator which generate the required pulse-width for normal operation of the motors. Power supply for this experimental setup was provided by an AC-DC power pack at the university laboratory.

The power-supply from each channel is connected to the respective ESC of the motors. ESCs are device which interpret control signal supplied into precise modulated current to the motors which in turn generates the correct spin-rates for lifting and maneuvering.

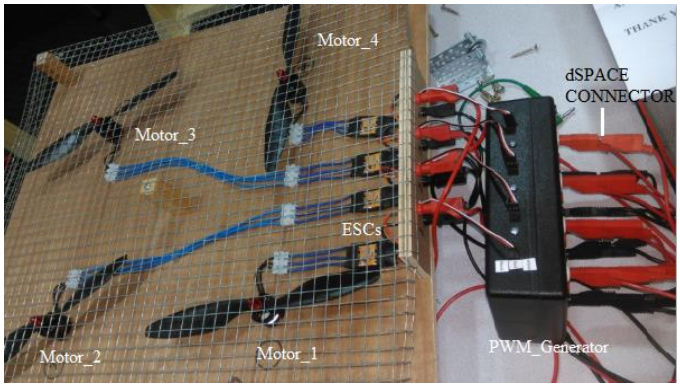

Figure 9 Quadcopter hardware prototype

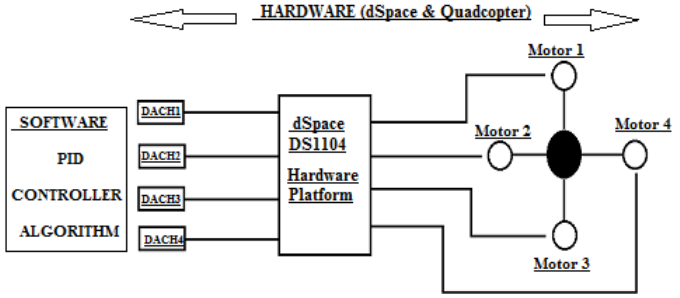

Figure 10 Schematic interfacing of control signal to the hardware Quadcopter via dSPACE 1104 R&D platform

Figure 11 Power supply and PWM Generator

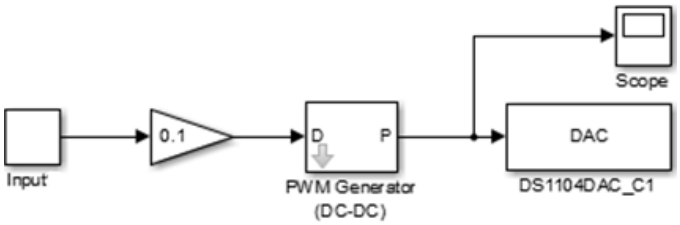

Figure 12 Simulink PWM-Model

The ESCs operates on PWM signal, hence power supply from the power generator were converted from DC to PWM signal by the PWM signal-generator and fed appropriately to the ESCs in accordance with desired input value of the Simulink block.

To mirror and control the PWM-signal from the generator with the control signal from the software controller, a Simulink PWM-generator model was created to interface the real-time PWM signal to the ESCs of the quadcopters motors (an extract is shown in Figure 12).

Digital display oscilloscope was used to output the value and waveform of the generated PWM signal as well as the supplied voltage and current from the power supply. Also signal from the DS1104 to the PWM-device is fed to an oscilloscope which provides visual details of the control-signal.

V. Discussion

Our overall Simulink for the realization of this project is shown in Figure 13.

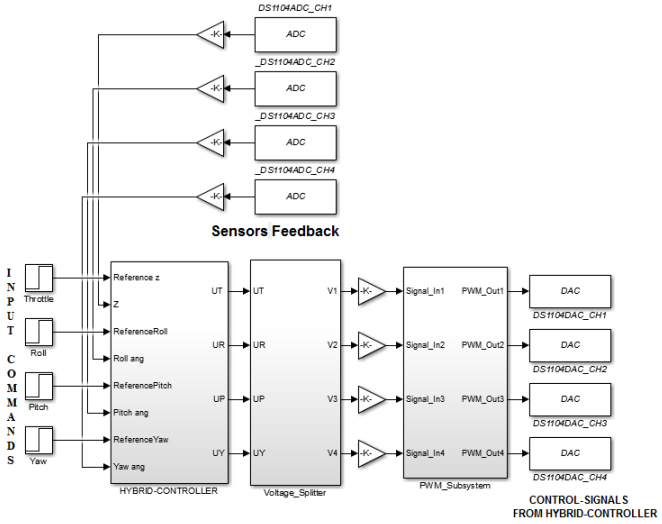

Figure 13 MATLAB/Simulink model representation

In Figure 13, the ADC and DAC blocks were the dSpace interface used by the model for receiving and outputting signal respectively. While input commands for achieving the desired setpoint is entered via the step input blocks.

Figure 14 PWM-signal generated for reference-input of 5m

Real-time PWM signal generated on applying control-signal to the hardware quadcopter model via PWM generator using DS1104 controller-board for reference input-command produces expected results with no overshoot; summary for input height of 5 meter for channel-1 of the display oscilloscope is shown in Figure 14 and presented as follows:

- PWM displayed-signal 6.005% duty-cycle
- 50Hz-frequency as required by ESCs
- 0.000% overshoot
- 5.40-volts input

Similar results were obtained for other channels.

Please see our video and model file for full results demonstration.

The PWM-generator operates at 50Hz and generates the required PWM signal to the ESC of the motors. The PWM-waveform generated by the generator is a function of the voltage supplied to it. Too high voltage results in a constant DC output (which is not a PWM-signal that can drive the motors) and too-low voltage results in a PWM-signal with too very little ON-time that is not sufficient to run the motors. As a result, it was important in the course of implementing the interface design process to understand the values and shapes of the PWM-signal that can drive the motors.

The ESCs incorporated in helicopter are programmed to be sensitive to under-voltage and too high-voltage by generating a low-pitch beeping sound for the former and a high-pitch beeping sound for the latter. As well as preventing spontaneous shutting down of the motor and erratic unsettling flight dynamics.

Table 3 shows the data gathered while testing the motors with PWM-signals to understand the dynamics and operation of the motors.

Table 3 ESCs PWM Signal Pulse-Width

PWM signal (us)	Threshold pulse-width (%)	Minimum pulse-width (%)	Minimum pulse-width (%)
Motor 1	5.8	6.1	8
Motor 2	5.8	6.1	8
Motor 3	5.8	6.1	8
Motor 4	5.8	5.1	7

The minimum and maximum pulse-width were the pulse-width that drove our ESCs.

VI. Conclusions

In this paper, a software and hardware model was designed, likewise a software PID controller. In this contribution, the author presented a real-time hardware experimental testing of a quadcopter and PID controller using dSpace DS1104 controller-board to interface a software model with a hardware helicopter.

The quadcopter and controller system was tested for flight condition. Results show the efficacy of integration and the ability of the software controller demonstrating proper control of the hardware quadcopter.

VII. Future Work

The author aim to implement the PID controller on a microcontroller and demonstrate a full functioning programmable hardware quadcopter; Furthermore, the author also aims to upgrade the quadcopter design to incorporate a SLAM (Simultaneous Localization and Mapping).

Acknowledgment

The designs, researches and results herein presented in this paper were part of my MSc dissertation project, undertaken at the University of Derby, UK, under the supervision of Dr. Amar Bousbaine.

Nomenclature

τ_d	load-torque (N·m)
ω	angular-velocity (rad·s^{-1})
η	motor's efficiency (%)
ψ	yaw-angle (rad)
ϕ	roll-angle (rad)
θ	pitch-angle (rad)
Ω	angular-velocity of motor (rad·s^{-1})
d	drag-coefficient
F	force (N)
g	acceleration due to Gravity (m·s^{-2})
i	motor-current (Amps)
I	identity-matrix
I_{xx}, I_{yy}, I_{zz}	rotational inertia about the x, y, and z-axis respectively (N·m·s^{-2})
J	motor's moment of inertia (kg·m^2)
K	LQR gain-matrix
k_e	motor's back-EMF constant (N·m/A)
k_q	motor torque-constant (Nm/A)
l	distance from center of the chassis-frame and the motor (m)
m	mass of quadcopter (kg)
R	motor-resistance (ohms)
T	thrust (N)
V	input voltage to quadcopter motor (volts)
X	x-axis of the quadcopter
Y	y-axis of the quadcopter

Abbreviations

ADCH	analog-digital channel
CAD	computer-aided design
DACH	digital-analog channel
DOF	degree-of-freedom
ESC	electronic speed controller
NB	note before
PID	proportional-integrator-derivative
PWM	pulse width modulation
RTI	real-time interface
UAV	unmanned aerial vehicle
VTOL	vertical take-off and landing

REFERENCES

[1] Andrew Gibiansky, 2012. Quadcopter Dynamics and Simulation (blog). Available at: http://andrew.gibiansky.com/blog/physics/Quadcopter-dynamics/ (Accessed 20th November, 2016).

[2] Argentim L., Rezende W.C. and Santos P.E., 2013. PID, LQR and LQR-PID on a Quadcopter platform. Research-Gate Publication.

[3] Azad Ghaffari (2012). dSPACE and Real-Time Interface in Simulink. San Diego State University, San Diego, USA.

[4] Araki M. (n.d). PID Control (pdf). Kyoto University, Japan.

[5] Bousbaine A., Wu M. H. and Poyi G. T., 2014. Modeling and Simulation of a Quad-Rotor Helicopter. University of Derby

[6] dSpace, 2009. www.dspace.com.

[7] Gwangtim T. Poyi, 2014. A Novel Approach to the Control of Quadrotor Helicopter Using Fuzzy-Neural Network. University of Derby.

[8] John, S. (2013) Artificial Intelligent-Based Feed-forward Optimized PID Wheel Slip Controller. AFRICON, 12 September, 2013, Pointe-Aux-Piments, 1-6.

[9] Mathworks Inc., 2016. www.mathworks.com.

[10] Sumaila Musa, 2017. Techniques for Quadcopter Modelling & Design: A review. Journal of Unmanned System Studies, Vol. 5(3), pp. 66-75.

[11] Praveenkumar U. and Prabhakaran G. (2014). Design and Fabrication of High Endurance RUAV with Rechargeable Solar Power Source (pdf). Proceedings of 5th IRF International Conference, India.

[12] Quijano N. and Passino K. (2015). A Tutorial Introduction to Control Systems Development and Implementation with dSPACE. Ohio State University, Columbus, USA.

SUPPLEMENTARY FILES

Name	Size	Type
993-3458-1-SP.mp4	20 MB	MP4 Video File
993-3537-1-SP.pdf	471 kB	Portable Document Format File
993-3460-1-SP.slx	38 kB	Simulink Model File