Third International Conference on
Advances in Control and Optimization of Dynamical Systems
March 13-15, 2014. Kanpur, India

Quadrotor: Design, Control and Vision Based Localization

Anurag Sai Vempati Email:vanurag@iitk.ac.in
Vipul Choudhary Email:khadav@gmail.com
Laxmidhar Behera Email:lbehera@iitk.ac.in

Department of Electrical Engineering, Kanpur, UP 208016 INDIA

Abstract: Modelling and control of Unmanned Aerial Vehicles (UAV) is a challenging problem owing to the inherent non-linearity. They demand ideal control for generating precise motion. This work aims at designing and building a light-weight quadrotor with improved on-board computational power and a diverse set of applications. Computer Aided Design (CAD) has been used to estimate mass and inertia parameters of the system. We briefly describe the functionality of various hardware components used in the fabrication process. Our work also describes dynamics of the system and state-space formulation for the controller design. We discuss two control strategies and evaluate their performance: (i) a simple PID controller and (ii) a back-stepping controller. A Simulator is designed to easily verify the control strategies and evaluate their effectiveness. We also incorporated vision based localization using a simple monochrome camera output. In the end, we describe the future work and the possible applications of our system. All the simulation and the field test results are shown where and when they are necessary.

Keywords: Quadrotor; Back-stepping control; simulator; localization.

1. INTRODUCTION

As per Hoffmann et al. (2004), a quadrotor, also called a quadrotor helicopter or quadcopter, is a multicopter that is lifted and propelled by four rotors. Quadrotors are much easier to build and maintain and yet produce exceptional manueverability in the 3D space. Unlike a conventional helicopter, a quadrotor's rotor blade pitch angle need not be varied. Quadrotors, in their initial stages of development were very inefficient in terms of their payload capacity, agility and computational power. With recent advancements, both in hardware and software, the quadrotors today are much smaller, lighter and computationally rich. They have better manoeuvrability, longer flight time and fast on-board computation. They are enjoying almost all the capabilites as any ground vehicle with an added advantage of being able to move in an additional dimension. Owing to all these reasons quadrotors have opened up plethora of possibilites in robotics research in areas like control, vision based navigation and aerial mapping. Their applications include, but not restricted to, search and rescue missions (SAR), surveillance and disaster management. As external infrastructure for navigation and communication is usually not available in such tasks, robotic systems must be able to operate autonomously (Tomic et al., 2012).

UAVs that navigate in outdoor environment typically use GPS to get their current location. Indoor operation on the other hand use motion capture systems (VICON), lasers or deploy localization techniques (SLAM - Simultaneous Localization and Mapping) based on monochrome camera, lasers or ultrasonic waves. Each of these has it's own share of implementability issues. Technologies like GPS and motion capture limit the functionality of the UAV. A global positioning system (GPS) does not work indoors and motion-capture systems are expensive, and their sensor array has a limited spatial extent that is impractical to scale up for large indoor environments. Moreover, during the SAR (Search And Rescue) operations, the robots have to operate in unstructured indoor and outdoor environments, such as collapsed buildings or gorges. Navigation systems therefore have to work without GPS, since their availability cannot be guaranteed. SLAM techniques based on Laser range finders (LRFs) or monocular camera are constrained by hardware limitations. LRFs are heavy and power hungry which drain out the battery quickly, preventing their application to the next generation of much smaller quadcopters. SLAM on the other hand requires a lot of computational power, which is not readily available on flying systems. Therefore, the authors in Grzonka et al. (2009) and Bachrach et al. (2009) chose to send laser scanner data to a ground station for processing. Pose estimation and high-level tasks are done on the ground station, whereas control and stabilization of the platform are done on the quadrotor. In Visual SLAM, transmitting the images wirelessly to a ground station increases system complexity, control latency, and the susceptibility to interference and dropouts.

Vision over all other techniques, has the advantage that the sensor is small, light weight, and low power, which will become increasingly important as the size of aerial vehicles decreases. Vision can provide essential navigational competencies such as odometry, attitude estimation, mapping, place and object recognition, and collision detection both indoor and outdoor. Mahony et al. (2012) say there is a long history of applying vision to aerial robotic systems for indoor and outdoor environments, and the well-known Parrot ARDrone game device makes strong use of vision

for attitude and odometry (Bristeau et al., 2011). Vision can also be used for object recognition based on color, texture, and shape, as well as collision avoidance. More recently, through the optimization of algorithms and faster processors, visual SLAM techniques which once required huge computational resources, no longer required off-board computation support. Pose estimation and planning can now be done onboard. Further, the pose estimate of the SLAM can be fused with inertial measurement unit (IMU) measurements in an extended Kalman filter (EKF) to obtain a full-state estimate, which can then be used for control.

The control problem is challenging for several reasons. First, the system is underactuated: there are four inputs while the configuration space is six dimensional. Second, the aerodynamic model described above is only approximate. Finally, the inputs are themselves idealized. In practice, the motor controllers must overcome the drag moments to generate the required speeds and realize the input thrust and moments. The dynamics of the motors and their interactions with the drag forces on the propellers can be difficult to model, although first-order linear models are a useful approximation (Kumar and Michael, 2012). The simplest control method involves linearization of the quadrotor dynamics around the hover state and then using various linear control methods to stabilize the system (SATICI et al., 2013). Controllers designed using such linear methods may be used to track slow motions. However, nonlinear control methods are needed to achieve higher performance and to execute more aggressive motions.

2. ROTOR DYNAMICS

The quadrotor is a highly non-linear, six degree-of-freedom, multi-input-multi-output and under-actuated system. It is controlled by varying the thrust forces of each rotor and balancing the drag torque. A quadrotor has two sets of counter-rotating propellers, therefore neutralizing the effective aerodynamic drag. It has three principal modes of operation (Fig. 1):

(1) Vertical movement is controlled by simultaneously increasing or decreasing the thrust of all rotors
(2) Yaw moment is created by proportionally varying the speeds of counter-rotating parts
(3) Roll and pitch angles can be controlled by applying differential thrust forces on opposite rotors of the quadrotor

For the i^{th} rotor with angular velocity ω_i, lift force and drag torque is given as (Pounds et al., 2004):

$$F_i = b\omega_i^2$$

$$\tau_i = d\omega_i^2$$

where, b and d are thrust and drag moments respectively. The total thrust force can then be given as

$$f = \sum_{i=1}^{4} F_i$$

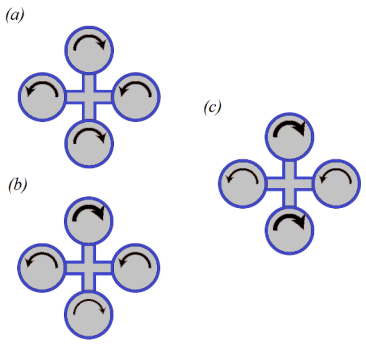


Fig. 1. Principal models of quadrotor flight. (a) hovering motion and vertical movement due to balanced torques; (b) difference in thrust to create roll/pitch moment; (c) difference in torque to create yaw moment

3. DYNAMIC MODELLING

3.1 State Variables

As shown in Fig., (u, ϕ, p) is the linear velocity along the roll-axis direction, angle rotated and the angular velocity about the roll-axis. (w, ψ, r) is the linear velocity along the yaw-axis direction, angle rotated and the angular velocity about the yaw-axis. (v, θ, q) is the linear velocity along the pitch-axis direction, angle rotated and the angular velocity about the pitch-axis. (x, y, z) is the position of the quadrotor along $\hat{i}$, $\hat{j}$ and $\hat{k}$ axes of the inertial frame.

3.2 Rotation Matrices

We have five coordinate reference frames in total. Namely, Inertial frame, vehicle frame, yaw-adjusted frame, pitch-adjusted frame and body frame. Inertial frame is fixed on the ground at a predefined home location. The unit vector $\hat{i}$ is directed North, $\hat{j}$ is directed East, and $\hat{k}$ is directed towards the Earth. Vehicle frame has axes parallel to the inertial frame but has the origin shifted to the quadrotor's center of gravity. Vehicle frame's yaw is adjusted to match the quadrotor's yaw to get the yaw-adjusted frame which is then pitch adjusted to get pitch-adjusted frame. Finally body frame is obtained by adjusting the roll of the pitch-adjusted frame. The transformation from the inertial to vehicle frame is just a simple translation. The transformation from vehicle to body frame is given by the following rotation matrix:

$$\begin{aligned} R_v^b(\phi,\theta,\psi) &= R_p^b(\phi)R_y^p(\theta)R_v^y(\psi) \\ &= \begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos\phi & \sin\phi \\ 0 & -\sin\phi & \cos\phi \end{pmatrix} \times \begin{pmatrix} \cos\theta & 0 & -\sin\theta \\ 0 & 1 & 0 \\ \sin\theta & 0 & \cos\theta \end{pmatrix} \times \\ &\quad \begin{pmatrix} \cos\psi & \sin\psi & 0 \\ -\sin\psi & \cos\psi & 0 \\ 0 & 0 & 1 \end{pmatrix} \\ &\quad \begin{pmatrix} c\theta c\psi & c\theta s\psi & -s\theta \\ s\phi s\theta c\psi - c\phi s\psi & s\phi s\theta s\psi + c\phi c\psi & s\phi c\theta \\ c\phi s\theta c\psi + s\phi s\psi & c\phi s\theta s\psi - s\phi c\psi & c\phi c\theta \end{pmatrix} \end{aligned} \tag{1}$$

where, R_p^b is the transformation from the pitch-adjusted frame to body frame, R_y^p is the transformation from the

yaw-adjusted frame to pitch-adjusted frame, R_v^y is the transformation from the vehicle frame to yaw-adjusted frame and R_v^b is the transformation from the vehicle frame to body frame. Since, each of the quadrotor parameters: roll, pitch and yaw are measured relative to different frames, these transformations help us in interplay of the variables.

3.3 Kinematics

Translational Kinematics:- The state variables $(\dot{x}, \dot{y}, \dot{z})$ are inertial frame parameters whereas, velocities (u,v,w) are body frame parameters. They can be related through the transformation ,matrix as follows:

$$\begin{aligned}\begin{pmatrix}\dot{x}\\ \dot{y}\\ \dot{z}\end{pmatrix} &= R_b^v \begin{pmatrix}u\\ v\\ w\end{pmatrix} = (R_v^b)^T \begin{pmatrix}u\\ v\\ w\end{pmatrix} \\ &= \begin{pmatrix} c\theta c\psi & s\phi s\theta c\psi - c\phi s\psi & c\phi s\theta c\psi + s\phi s\psi \\ c\theta s\psi & s\phi s\theta s\psi + c\phi c\psi & c\phi s\theta s\psi - s\phi c\psi \\ -s\theta & s\phi c\theta & c\phi c\theta \end{pmatrix} \begin{pmatrix}u\\ v\\ w\end{pmatrix}\end{aligned} \tag{2}$$

Rotational Kinematics:- Since the yaw(relative to vehicle frame), pitch(relative to yaw-adjusted frame) and roll(relative to pitch-adjusted frame) are measured relative to different coordinate systems, the transformation for each is different. So, the angular velocities (p, q, r) are obtained as follows:

$$\begin{aligned}\begin{pmatrix}p\\ q\\ r\end{pmatrix} &= \begin{pmatrix}\dot{\phi}\\ 0\\ 0\end{pmatrix} + R_p^b(\phi) \begin{pmatrix}0\\ \dot{\theta}\\ 0\end{pmatrix} + R_p^b(\phi) R_y^p(\theta) \begin{pmatrix}0\\ 0\\ \dot{\psi}\end{pmatrix} \\ &= \begin{pmatrix} 1 & 0 & -\sin\theta \\ 0 & \cos\theta & \sin\phi\cos\theta \\ 0 & -\sin\phi & \cos\phi\cos\theta \end{pmatrix} \begin{pmatrix}\dot{\phi}\\ \dot{\theta}\\ \dot{\psi}\end{pmatrix}\end{aligned} \tag{3}$$

3.4 Dynamics

Translational Dynamics By applying the Newton's second law to the translational motion, we get:

$$\mathbf{f} = m\frac{d\mathbf{v}}{dt} = m\left(\frac{d\mathbf{v}}{dt} + \boldsymbol{\omega_b} \times \mathbf{v}\right)$$

where, $\mathbf{v} = (u, v, w)^T$ and $\boldsymbol{\omega_b} = (p, q, r)^T$. We have used coriolis equation to evaluate time derivative of velocities in the non-inertial frame (body frame). Substituting the relevant equations we get:

$$\begin{pmatrix}\dot{u}\\ \dot{v}\\ \dot{w}\end{pmatrix} = \begin{pmatrix}rv - qw\\ pw - ru\\ qu - pv\end{pmatrix} + \frac{1}{m}\begin{pmatrix}f_x\\ f_y\\ f_z\end{pmatrix} \tag{4}$$

Assuming absence of any external disturbances, we have,

$$\begin{pmatrix}f_x\\ f_y\\ f_z\end{pmatrix} = R_v^b \begin{pmatrix}0\\ 0\\ mg\end{pmatrix} + \begin{pmatrix}0\\ 0\\ -f\end{pmatrix}$$

where f is the total upward thrust and mg is the weight of the quadrotor.

Rotational Dynamics By applying Newton's laws to rotational motion, we get:

$$\boldsymbol{\tau} = \frac{d\mathbf{L}}{dt} = \left(\frac{d\mathbf{L}}{dt} + \boldsymbol{\omega_b} \times \mathbf{L}\right)$$

where, $\mathbf{L} = J\boldsymbol{\omega_b}$ is the angular momentum and $\boldsymbol{\tau}$ is the applied torque. We have used coriolis equation to evaluate time derivative of angular momentum in the non-inertial frame (body frame). Assuming quadrotor a symmetric body, J simplifies to a diagonal matrix. This equation can be simplified into:

$$\begin{pmatrix}\dot{p}\\ \dot{q}\\ \dot{r}\end{pmatrix} = \begin{pmatrix} qr\frac{J_y - J_z}{J_x} \\ pr\frac{J_z - J_x}{J_y} \\ pq\frac{J_x - J_y}{J_z} \end{pmatrix} + \begin{pmatrix} \frac{\tau_\phi}{J_x} \\ \frac{\tau_\theta}{J_y} \\ \frac{\tau_\psi}{J_z} \end{pmatrix} \tag{5}$$

where, $\boldsymbol{\tau} = (\tau_\phi, \tau_\theta, \tau_\psi)$ and J_x, J_y, J_z are the diagonal entries of the inertial matrix J.

Equations 2-5 now describe the 6DOF model of the quadrotor. These will be the various modules that will be used in our simulator. Beard (2008) describe some approximations to make these set of equations more suitable for control design. The coriolis terms, time derivative of the transformation matrix are neglected to get these much simpler set of equations:

$$\begin{aligned}\ddot{\phi} &= \dot{\theta}\dot{\psi}\left(\frac{J_y - J_z}{J_x}\right) + \frac{U_2}{J_x} \\ \ddot{\theta} &= \dot{\phi}\dot{\psi}\left(\frac{J_z - J_x}{J_y}\right) + \frac{U_3}{J_y} \\ \ddot{\psi} &= \dot{\phi}\dot{\theta}\left(\frac{J_x - J_y}{J_z}\right) + \frac{U_4}{J_z} \\ \ddot{z} &= \frac{U_1}{m}\cos\phi\cos\theta - g \\ \ddot{x} &= \frac{U_1}{m}\left(\cos\phi\sin\theta\cos\psi + \sin\phi\sin\psi\right) \\ \ddot{y} &= \frac{U_1}{m}\left(\cos\phi\sin\theta\sin\psi - \sin\phi\cos\psi\right)\end{aligned} \tag{6}$$

where, the control inputs U_1-U_4 are given as:

$$\begin{aligned}U_1 &= b\left(\omega_1^2 + \omega_2^2 + \omega_3^2 + \omega_4^2\right) \\ U_2 &= b\left(\omega_4^2 - \omega_2^2\right) \\ U_3 &= b\left(\omega_3^2 - \omega_1^2\right) \\ U_4 &= b\left(\omega_4^2 + \omega_2^2 - \omega_3^2 - \omega_1^2\right)\end{aligned} \tag{7}$$

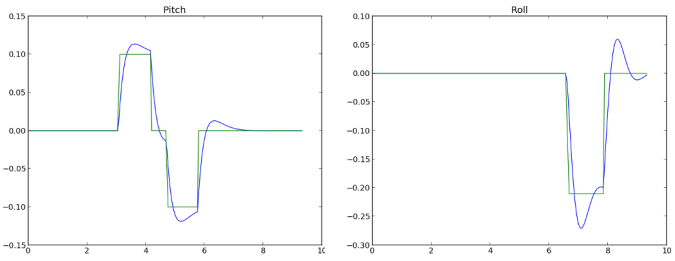
4. CONTROL

4.1 Attitude Control

PID:- As a baseline method, each of the control inputs U_i are modelled as $U_i = K_p e_i + K_d \dot{e}_i + K_i \int e_i$ where, $e_1 = z - z_{ref}$, $e_2 = \phi - \phi_{ref}$, $e_3 = \theta - \theta_{ref}$ and $e_4 = \psi - \psi_{ref}$. $(z_{ref}, \phi_{ref}, \theta_{ref}, \psi_{ref})$ are the reference values for (z, ϕ, θ, ψ). The parameters K_p, K_d and K_i are chosen from Ziegler-Nichols tuning (Ziegler and Nichols, 1942). The performance of PID controller in tracking the reference pitch and roll is shown in Fig. 2

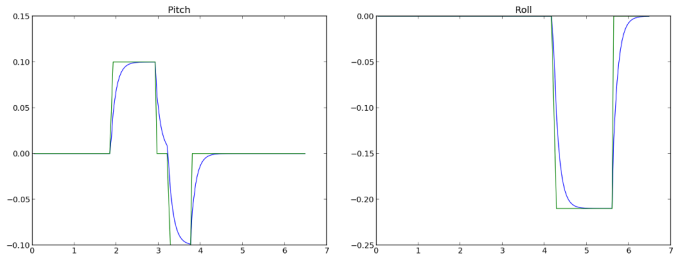
As can be seen, the control is quite poor to perform any dynamic manoeuvres. It takes a long time to settle and is quite oscillatory.

Back-Stepping:- Back-Stepping (Bouabdallah, 2007) is employed for building stable controllers for nonlinear systems with a recursive structure. We can design a



(a) PID: tracking reference pitch (b) PID: tracking reference roll

Fig. 2. PID controller performance



(a) Tracking reference pitch (b) Tracking reference roll

Fig. 3. Back-Stepping controller performance

controller for already-known-to-be-stable system and then propagate each controller to stabilize it's outer subsystem (Khalil, 2002). The system of equations [6] can be put in a strict-feedback form as follows:

$$\dot{X} = f(X,U) = \begin{pmatrix} x_2 \\ x_4 x_6 a_1 + b_1 U_2 \\ x_4 \\ x_2 x_6 a_2 + b_2 U_3 \\ x_6 \\ x_4 x_2 a_3 + b_3 U_4 \\ x_8 \\ \frac{U_1}{m}(\cos x_1 \cos x_3) \\ x_{10} \\ \frac{U_1}{m} u_x \\ x_{12} \\ \frac{U_1}{m} u_y \end{pmatrix} \tag{8}$$

$$X = (x_1, x_2, \ldots, x_{12})^T = \begin{pmatrix} \phi \\ \dot{x_1} = \dot{\phi} \\ \theta \\ \dot{x_3} = \dot{\theta} \\ \psi \\ \dot{x_5} = \dot{\psi} \\ z \\ \dot{x_7} = \dot{z} \\ x \\ \dot{x_9} = \dot{x} \\ y \\ \dot{x_{11}} = \dot{y} \end{pmatrix} \tag{9}$$

where,

$$\begin{aligned} a_1 &= (J_y - J_z)/J_x \\ a_2 &= (J_z - J_x)/J_y \\ a_3 &= (J_x - J_y)/J_z \\ b_1 &= 1/J_x \\ b_2 &= 1/J_y \\ b_3 &= 1/J_z \\ u_x &= \cos x_1 \sin x_3 \cos x_5 + \sin x_1 \sin x_5 \\ u_y &= \cos x_1 \sin x_3 \sin x_5 - \sin x_1 \cos x_5 \end{aligned} \tag{10}$$

Four Augmented Lyapunov functions are used, one for each control input. The Lyapunov functions chosen are:

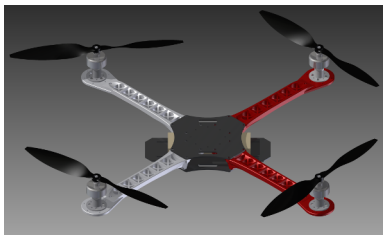


Fig. 4. CAD model of the quadrotor

$$\begin{aligned} V_1(z_1,z_2) &= \left(z_1^2 + z_2^2\right)^2; z_1 = x_{1d} - x_1; z_2 = x_2 - \dot{x_{1d}} - \alpha_1 z_1 \\ V_2(z_3,z_4) &= \left(z_3^2 + z_4^2\right)^2; z_3 = x_{3d} - x_3; z_4 = x_4 - \dot{x_{3d}} - \alpha_3 z_3 \\ V_3(z_5,z_6) &= \left(z_5^2 + z_6^2\right)^2; z_5 = x_{5d} - x_5; z_6 = x_6 - \dot{x_{5d}} - \alpha_5 z_5 \\ V_4(z_7,z_8) &= \left(z_7^2 + z_8^2\right)^2; z_7 = x_{7d} - x_7; z_8 = x_8 - \dot{x_{7d}} - \alpha_7 z_7 \end{aligned} \tag{11}$$

For the system to be asymptotically stable, we need to have $V_1(z_1,z_2) < 0, V_2(z_3,z_4) < 0, V_3(z_5,z_6) < 0$ and $V_4(z_7,z_8) < 0$. The four control inputs that we chose accordingly are:

$$\begin{aligned} U_1 &= \frac{m}{\cos x_1 \cos x_3}(z_7 + g - \alpha_7(z_8 + \alpha_7 z_7) - \alpha_8 z_8) \\ U_2 &= \frac{z_1 - a_1 x_4 x_6 - \alpha_1(z_2 + \alpha_1 z_1) - \alpha_2 z_2}{b_1} \\ U_3 &= \frac{z_3 - a_3 x_6 x_8 - \alpha_3(z_4 + \alpha_3 z_3) - \alpha_4 z_4}{b_2} \\ U_4 &= \frac{z_5 - a_5 x_8 x_{10} - \alpha_5(z_6 + \alpha_5 z_5) - \alpha_6 z_6}{b_3} \end{aligned} \tag{12}$$

The performance of Back-Stepping controller in tracking the desired Pitch and Roll is shown in Fig. 3. We can clearly see better performance in terms of reduced overshoot, smaller settling time and more stability as compared to PID control.

5. MECHANICAL DESIGN

5.1 Computer Aided Design (CAD)

A computer aided design model has been created in order to estimate the mass and inertial properties. This model has been created using Autodesk Inventor SE. Fig. 4 shows the assembly of the model. Mass and inertia properties are listed in the Table 1 and Table 2 respectively.

5.2 Hardware

Hardware components of the system are given in the Table 3.

Component	Mass (grams)	Quantity	Total Mass
Battery	230	1	230
Frame	350	1	350
Motor and Rotors	60	4	240
Embedded Systems	100	1	100
Quadrotor		Total	920

Table 1.

Axis	Moment of Inertia ($kg - m^2$)
X:J_x	0.009363917
Y:J_y	0.009382320
Z:J_z	0.018393126

Table 2.

Component No.	Name
1	Arduino Mega
2	ODROID-X2
3	XBee S2
4	Radio Controller
5	20 Amp ESC (Motor Driver)
6	700kV Brushless DC Motors
7	11*4.7 Propellers
8	3300mAh 3S 11.1V Li-Po Battery
9	Q450 Glass Fiber Quadrotor Frame
10	6DOF IMU (ITG3200/ADXL345)

Table 3.

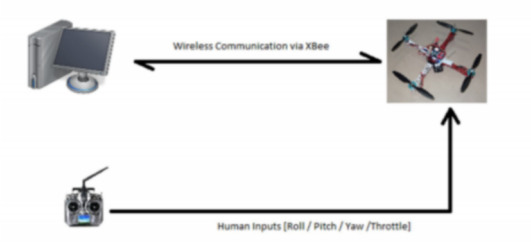


Fig. 5. Data Transfer Protocol

Actuators:- The motors are brush-less, out-runner, DC motors rated for 11.1 V, 12 amps [Kv: 700rpm/v]. Brush-less motors are preferred over brushed motors because of their efficiency, light weight and low power consumption. They use a pulsed DC fed to the stator field windings to create a rotating magnetic field and they operate at synchronous speed. The speed of rotation is controlled by the pulse frequency and the torque by the pulse current.

Electronic Speed Controller:- An electronic speed control or ESC is an electronic circuit with the purpose to vary an electric motor's speed. Brush-less ESC systems basically drive tri-phase Brush-less motors by sending sequence of signals for rotation.

XBee Radios:- The Xbee allows the microcontroller board to communicate wirelessly with the ground station. It has been used as a serial-usb replacement. XBee enables bi-directional communication between on-board microcontroller and ground station. Various sensors data are monitored and logged wirelessly using XBee. Various control parameters and human inputs [set points] can also be transmitted to the on-board microcontroller. For ease of use and simplicity we are using an additional r/c transmitter for transmission of throttle and desired orientation. Basic schematic for data transfer is given in the Fig. 5

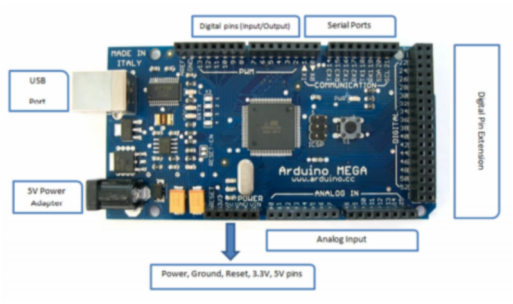


Fig. 6. Aurdino Mega (Image credits: Arduino)

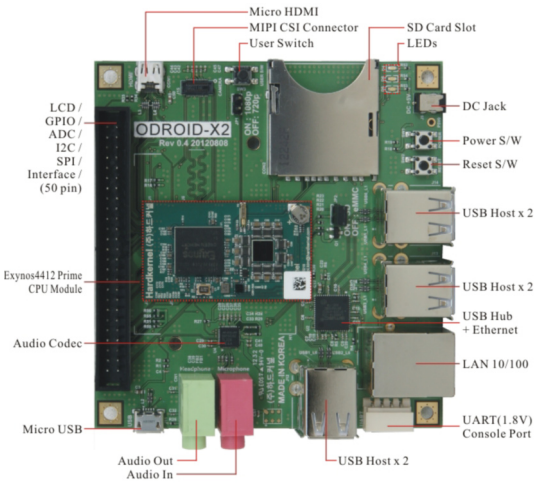


Fig. 7. ODROID-X2 (Image credits: Hardkernel)

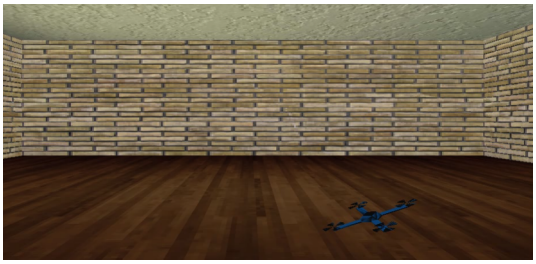
Inertial Measurement Unit:- An inertial measurement unit or IMU is an electronic device that measures and reports on a craft's angular velocity, orientation and gravitational forces using a combination of accelerometers and gyroscopes. The sensor communicates with microcontroller over a I_2C bus. An accelerometer simply measures acceleration, either due to motion or due to gravity. The gyroscope has been used for the measurement of angular velocity.

On-Board Controller:- Arduino Mega (Fig. 6) has been used as on-board control unit. This controller is based on ATmega 2560 microchip. It is an open-source, Atmel AVR processor based prototyping platform.

High Performance Board:- ODROID-X2 (Fig. 7) is an open development platform based on Exynos4412 Prime 1.7GHz ARM Cortex-A9 Quad Core with 2GB memory. Various hardware features can be seen in the picture. It can run Linux OS and has graphics support. This board is used to perform computationally expensive vision algorithms like SLAM in real-time.

6. SIMULATOR

Any slight error in the control strategy may result in a badly controlled flight and might damage the system. So, to overcome such problem we developed a Quadrotor simulator using PyGame and OpenGl. 3D model of the quadrotor is downloaded from Google warehouse. We can plug-in any control strategy and visualize the flight path



(a) Ground frame view

(b) First-person View

Fig. 8. Quadrotor Simulator

with user generated inputs from the keyboard. A First-person view is also simulated to get on-board camera feed (Fig. 8). A schematic of the simulator can be seen in Fig.

7. VISUAL SUBSYSTEM

For being able to do position control, one needs measurement of the quadrotor position in real-time. Many state-of-art quadrotors use techniques like motion capture or lasers for this purpose. This restricts the quad to a very controlled environment and requires a lot of calibration. We instead employ a localization algorithm that maps any unknown terrain and provides the position of the quadrotor with the help of the on-board camera.

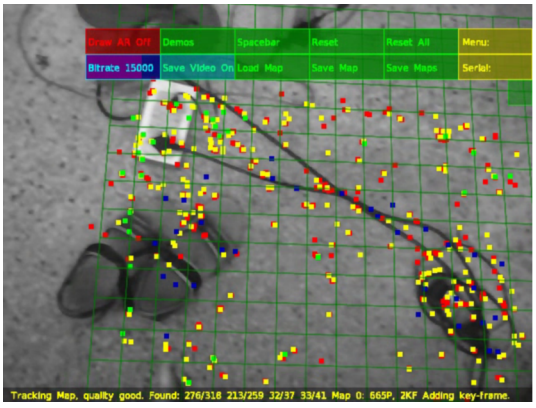
We use the software PTAM (Parallel Tracking and Mapping) developed by Georg Klein and David Murray at University of Oxford (Klein and Murray, 2007). It picks salient features from the surroundings and tracks these to get 6DOF transformation matrix of the camera and hence the quadrotors orientation and position relative to the ground frame. Fig. 9 shows mapping of the terrain (room's floor) with position and orientation of the quadrotor camera relative to this map as it is moving.

The PTAM algorithm is executed on the ODROID-X2 and realtime 3D position of the quadrotor is collected using data acquisition module. Fig. 10 shows the 3D trajectory of the quadrotor estimated by the PTAM as compared with the ground truth. The RMS error is about 5cms.

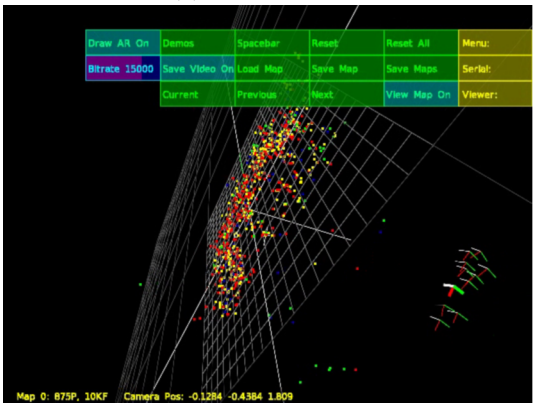
8. CONCLUSION

In this paper, the dynamics of a quadrotor is described and two control strategies namely PID and back-stepping are evaluated for attitude control of the quadrotor. A Simulator is designed to test the control algorithms. A completely vision based localization technique is employed to get the realtime position of the quadrotor. The mechanlical design of a quadrotor with complete onboard computational resources is also illustrated.

(a) Feature tracking



(b) Map generation



(c) Camera Localization

Fig. 9. Quadrotor Localization using PTAM

In the future, we plan to fuse PTAM estimate with IMU reading using an Extended Kalman Filter to remove the measurement noise and generate a precise trajectory. We also hope to implement position control to acheive set-point tracking of the quadrotor position. We can then address problems like path planning and obstacle avoidance based purely on vision.

REFERENCES

Bachrach, A., He, R., and Roy, N. (2009). Autonomous flight in unknown indoor environments. *International Journal of Micro Air Vehicles*, 1(4), 217–228.

Beard, R.W. (2008). Quadrotor dynamics and control. *Brigham Young University*.

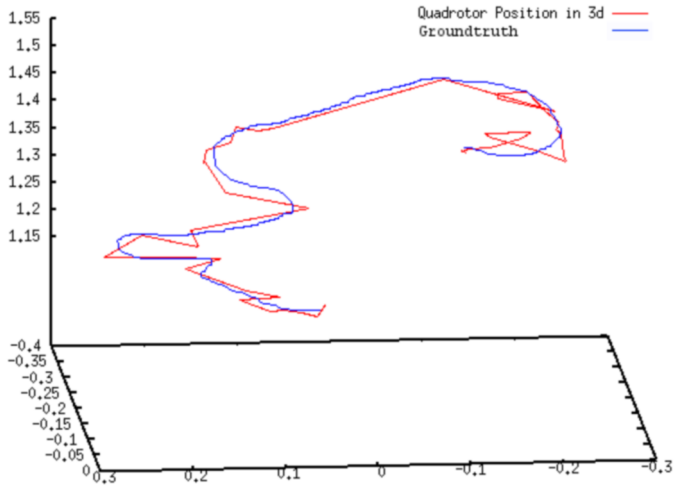


Fig. 10. 3D Trajectory of the quadrotor as estimated using PTAM in red and the ground truth in blue.

Bouabdallah, S. (2007). Design and control of quadrotors with application to autonomous flying. *Ecole Polytechnique Federale de Lausanne*.

Bristeau, P.J., Callou, F., Vissière, D., Petit, N., et al. (2011). The navigation and control technology inside the ar. drone micro uav. In *18th IFAC World Congress*, 1477–1484.

Grzonka, S., Grisetti, G., and Burgard, W. (2009). Towards a navigation system for autonomous indoor flying. In *Robotics and Automation, 2009. ICRA'09. IEEE International Conference on*, 2878–2883. IEEE.

Hoffmann, G., Rajnarayan, D.G., Waslander, S.L., Dostal, D., Jang, J.S., and Tomlin, C.J. (2004). The stanford testbed of autonomous rotorcraft for multi agent control (starmac). In *Digital Avionics Systems Conference, 2004. DASC 04. The 23rd*, volume 2, 12–E. IEEE.

Khalil, H.K. (2002). *Nonlinear systems*, volume 3. Prentice hall Upper Saddle River.

Klein, G. and Murray, D. (2007). Parallel tracking and mapping for small ar workspaces. In *Mixed and Augmented Reality, 2007. ISMAR 2007. 6th IEEE and ACM International Symposium on*, 225–234. IEEE.

Kumar, V. and Michael, N. (2012). Opportunities and challenges with autonomous micro aerial vehicles. *The International Journal of Robotics Research*, 31(11), 1279–1291.

Mahony, R., Kumar, V., and Corke, P. (2012). Multirotor aerial vehicles: Modeling, estimation, and control of quadrotor.

Pounds, P., Mahony, R., Gresham, J., Corke, P., and Roberts, J. (2004). Towards dynamically-favourable quad-rotor aerial robots. In *Proceedings of the 2004 Australasian Conference on Robotics & Automation*. Australian Robotics & Automation Association.

SATICI, A.C., POONAWALA, H., and SPONG, M.W. (2013). Robust optimal control of quadrotor uavs.

Tomic, T., Schmid, K., Lutz, P., Domel, A., Kassecker, M., Mair, E., Grixa, I.L., Ruess, F., Suppa, M., and Burschka, D. (2012). Toward a fully autonomous uav: Research platform for indoor and outdoor urban search and rescue. *Robotics & Automation Magazine, IEEE*, 19(3), 46–56.

Ziegler, J. and Nichols, N. (1942). Optimum settings for automatic controllers. *trans. ASME*, 64(11).