Outdoor Cooperative Control Experiments with Quadcopters

Apurva Joshi, Debraj Chakraborty,

AND

Mohit Gagrani, Subhadip Hazra, Saksham Singh, Pankaj Patel, Nitin Chavan, Indrajit Ahuja, Narendra Limbu, Sudeep Solanki, Soniya Jain, Harshal Sonar, Kritika Shahu, Abhishek G.S., Prof. Hoam Chung, Prof. D Manjunath, Dr. Ameer Mulla

Control and Computing Group, Electrical Engineering, IIT Bombay

Two Parts

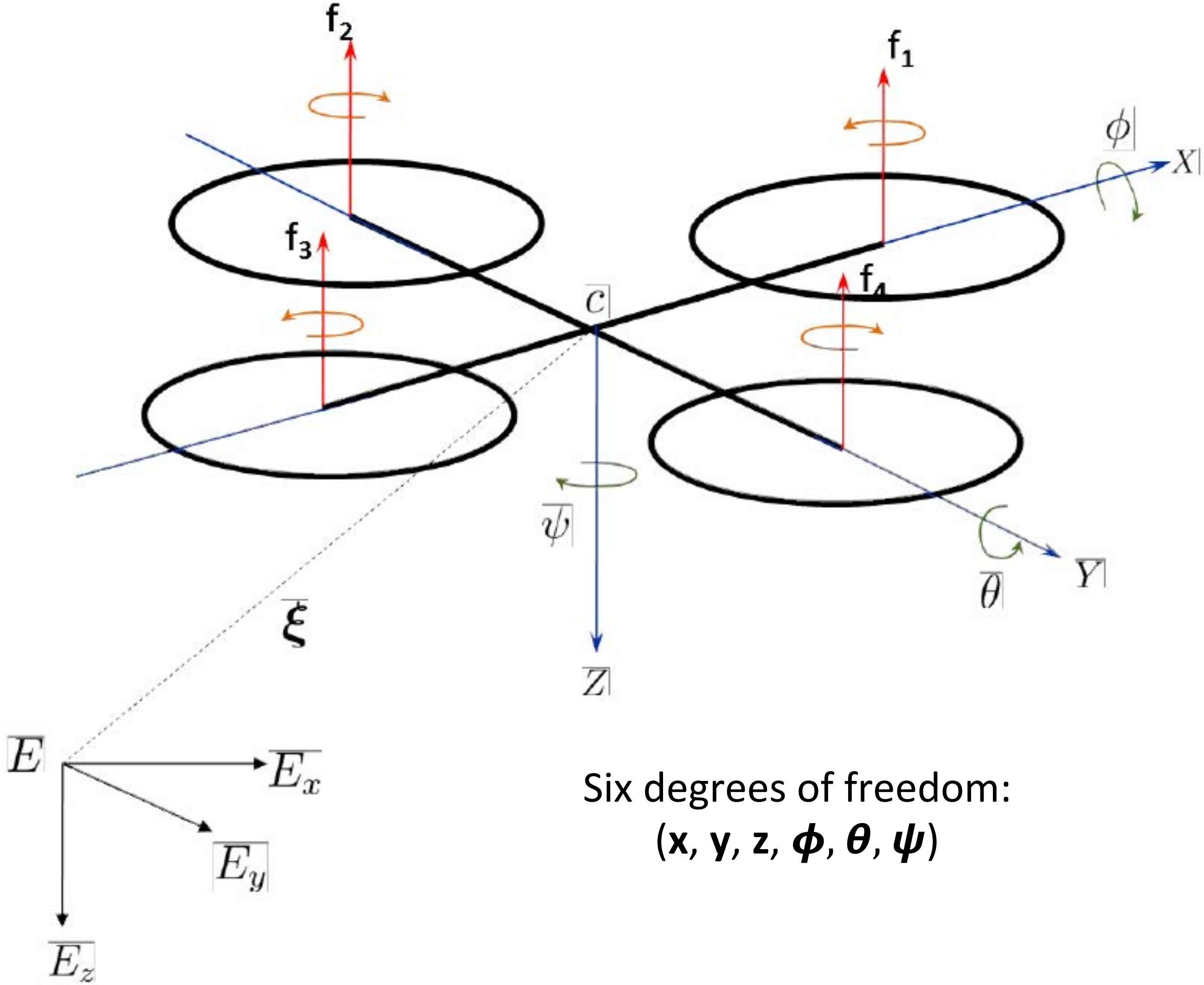
1. Quadcopters: Making, Controlling Flying
    - Basic Mathematical Model
    - Hardware Details
    - Software Platform
    - Autopilot Architecture
        - Primary Modes and
        - Autonomous Modes
            - GPS Hold
            - Waypoint Navigation
            - Leader Follower
2. Cooperative Control Experiments
    - Frames of reference
    - Inner Control loops
    - Waypoint navigation
    - Consensus law
    - Communication Protocols
    - Experiments

Quadcopters: Making, Controlling and Flying

Basic Mathematical Model

What is a Quadcopter?

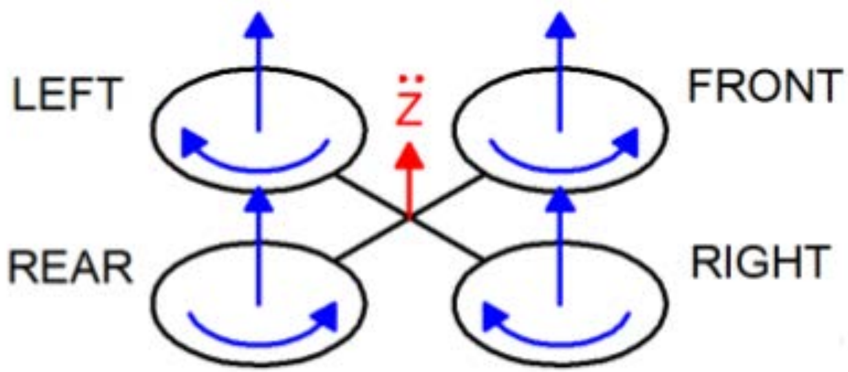
- Pair of rotating and counter-rotating rotors
- Fixed axes: Parallel to each other, perpendicular to the frame
- Fixed pitch propellers
- Variable speed of rotation



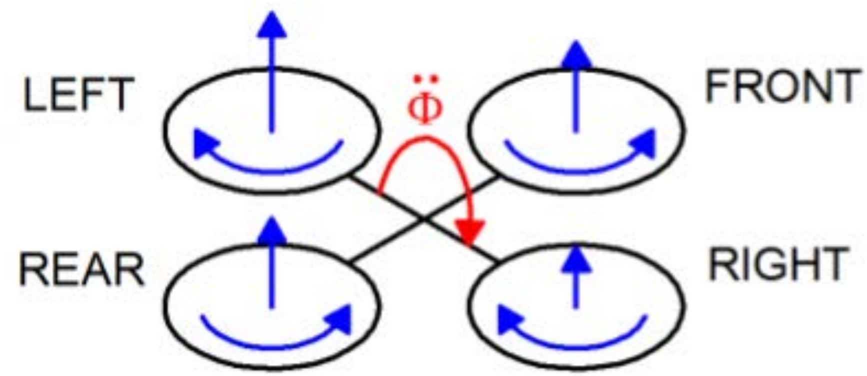
Six degrees of freedom:
(**x**, **y**, **z**, $\boldsymbol{\phi}$, $\boldsymbol{\theta}$, $\boldsymbol{\psi}$)

Basic Movements

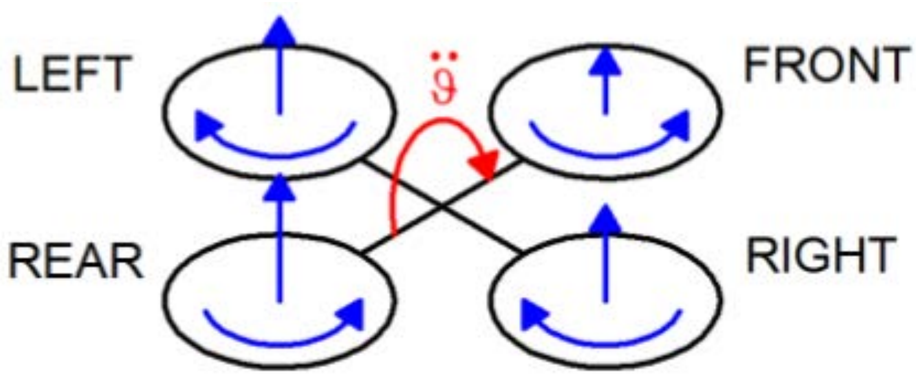
Lift: $f_i = b\omega_i^2$



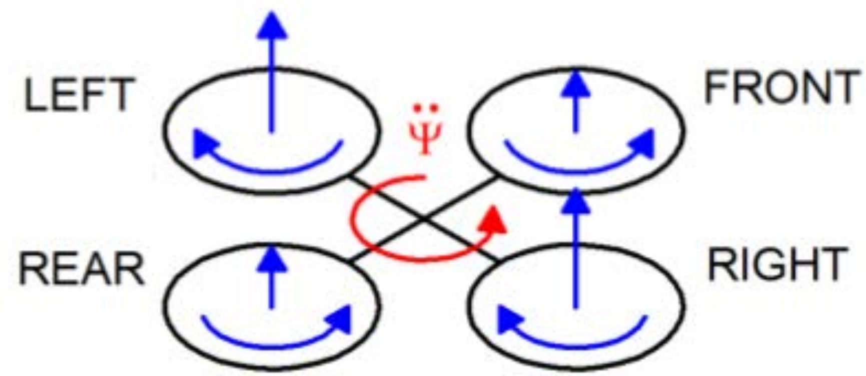
Altitude:



Roll $\tau_\phi = (f_2 - f_4)l$



Pitch $\tau_\theta = (f_1 - f_3)l$



$q_i = k\omega_i^2$

Yaw $\tau_\psi = q_1 - q_2 + q_3 - q_4$

Control Inputs

- Altitude
- Roll
- Pitch
- Yaw

$$\begin{aligned} u_1 &= f_1 + f_2 + f_3 + f_4 \\ &= b(\omega_1^2 + \omega_2^2 + \omega_3^2 + \omega_4^2) \end{aligned}$$

$$u_2 = bl(\omega_2^2 - \omega_4^2)$$

$$u_3 = bl(\omega_1^2 - \omega_3^2)$$

$$u_4 = d(\omega_1^2 - \omega_2^2 + \omega_3^2 - \omega_4^2)$$

Mathematical Modeling

To derive a model which:

- Describes motion of the quadrotor based on the control inputs
- Useful in predicting positions reached by the quadrotor by knowing the speeds of the four motors
- Assumptions in model:
  - Rigid, symmetrical structure
  - CoG and body fixed frame coincide
  - Rigid propellers

Mathematical Modeling

- The equations of motion obtained are:

$$m\ddot{x} = -(\cos\psi \sin\theta \cos\phi + \sin\psi \sin\phi)u_1$$
$$m\ddot{y} = -(\sin\psi \sin\theta \cos\phi - \cos\psi \sin\phi)u_1$$
$$m\ddot{z} = -(\cos\theta \cos\phi)u_1 + mg$$
$$I_x\ddot{\phi} = (I_y - I_z)\dot{\theta}\dot{\psi} - J_r\dot{\theta}\Omega + u_2$$
$$I_y\ddot{\theta} = (I_z - I_x)\dot{\phi}\dot{\psi} + J_r\dot{\phi}\Omega + u_3$$
$$I_z\ddot{\psi} = (I_x - I_y)\dot{\phi}\dot{\theta} + u_4$$

Motor Speeds and Input Voltage

$$
\begin{aligned}
u_1 &= b(\omega_1^2 + \omega_2^2 + \omega_3^2 + \omega_4^2) \\
u_2 &= lb(\omega_2^2 - \omega_4^2) \\
u_3 &= lb(\omega_1^2 - \omega_3^2) \\
u_4 &= b(\omega_1^2 - \omega_2^2 + \omega_3^2 - \omega_4^2)
\end{aligned}
\quad \Rightarrow \quad
\begin{aligned}
\omega_1^2 &= \frac{1}{4b}u_1 - \frac{1}{2bl}u_3 - \frac{1}{4d}u_4 \\
\omega_2^2 &= \frac{1}{4b}u_1 - \frac{1}{2bl}u_2 + \frac{1}{4d}u_4 \\
\omega_3^2 &= \frac{1}{4b}u_1 + \frac{1}{2bl}u_3 - \frac{1}{4d}u_4 \\
\omega_4^2 &= \frac{1}{4b}u_1 + \frac{1}{2bl}u_2 + \frac{1}{4d}u_4
\end{aligned}
$$

Hardware Details

Component Selection

Component	Parameter
Motors	Voltage, Wattage, Speed, Weight
Propellers	Diameter, Pitch
Battery	Voltage and Current Rating, Weight
Frame	Dimensions, Weight
ESCs	Current Rating
Controller and Electronics	Processing power, sensor integration, Software Support

Weight + Payload

Flight time

Thrust/Weight

Dimensions

Design Parameters

Motor + Propeller Selection

Total Thrust = 2.4 kg

Each motor = 600 g

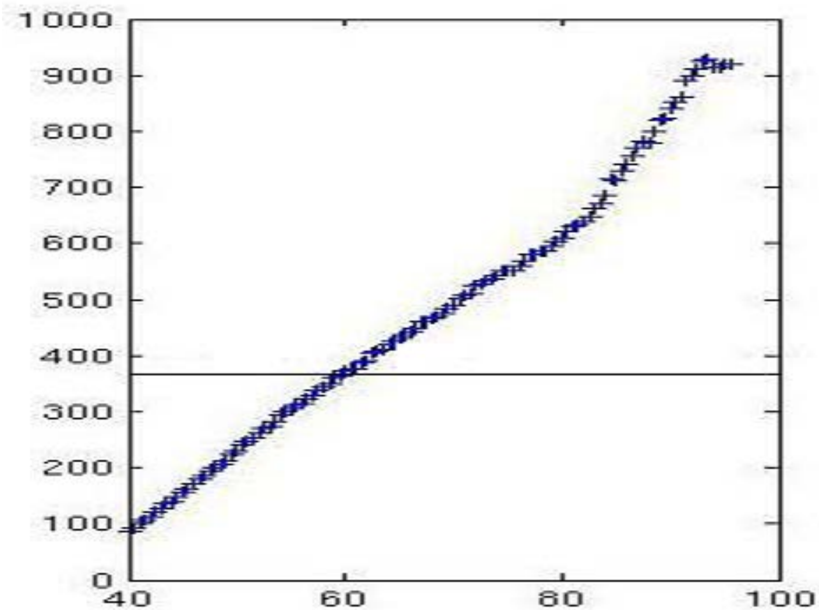
Turnigy Park 480 1320 Kv Motor

Select propeller with minimum dimensions

6x4: 420 g

7x5: 920 g

1320Kv Park 480 motor thrust plot:
Thrust produced v/s duty cycle



Linear operating range:
From 40% to 80%,
with Hovering thrust (380 gm) at 60%

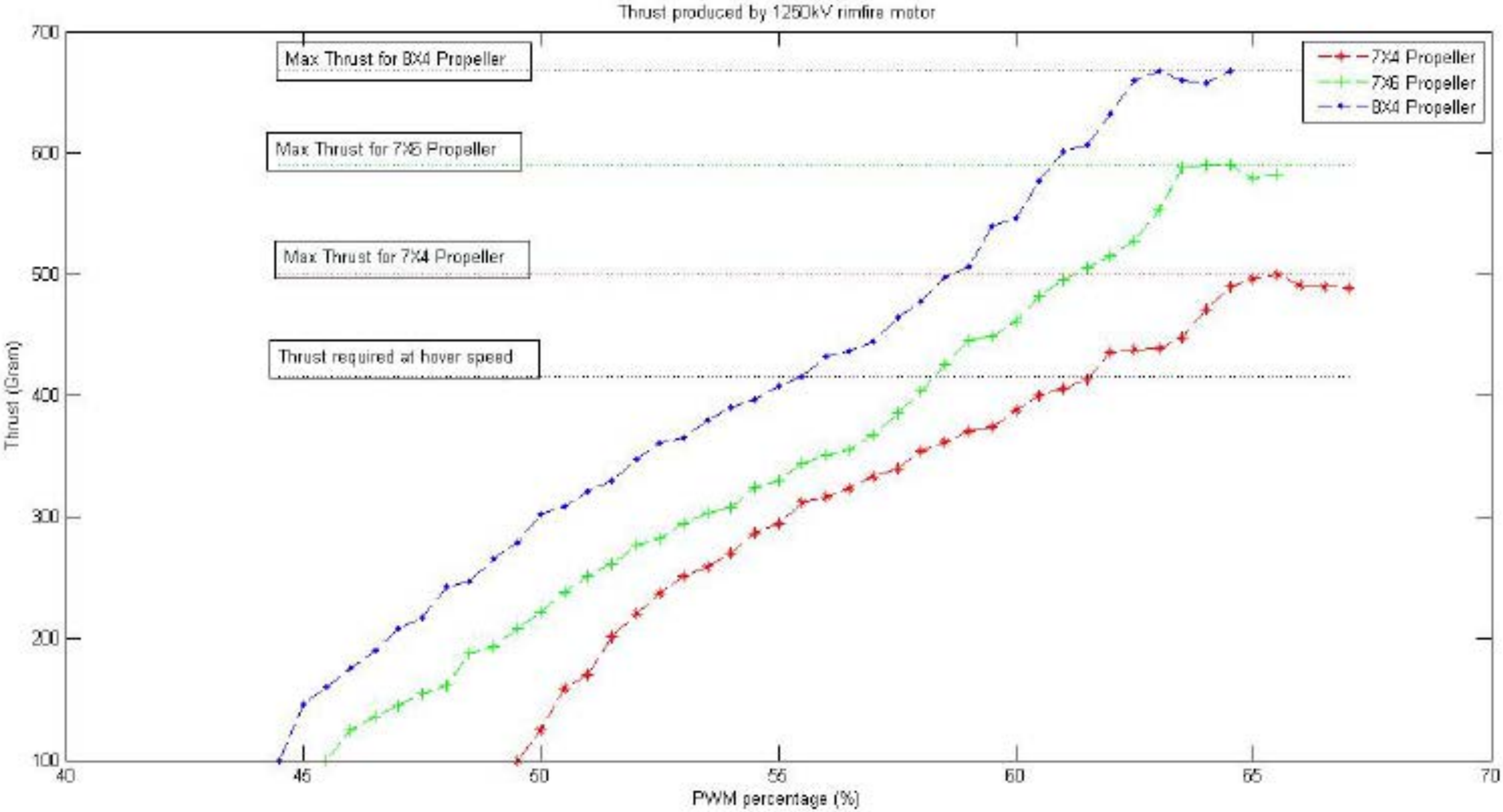
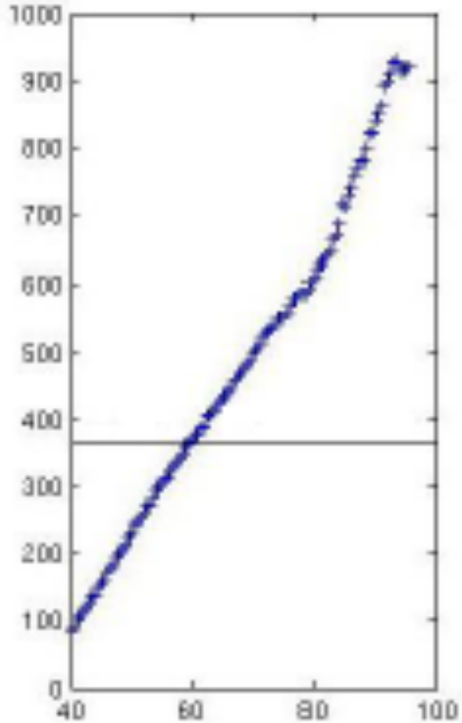


Figure 6.5: Thrust Plots for Rimfire 1250kV motor with different propellers

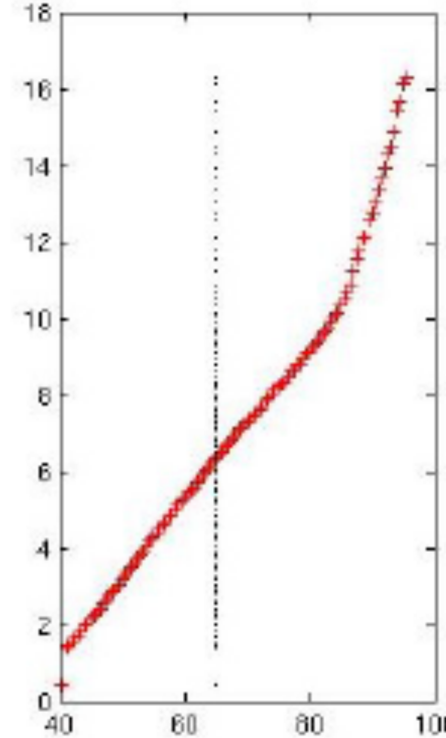
Table 6.1: Result Summary

Motor	Max Thrust(gm)			Max current(Amp)			Hover Current(Amp)			Hover Flight time(min)		
kV	7×4	7×6	8×4	7×4	7×6	8×4	7×4	7×6	8×4	7×4	7×6	8×4
1200	440.8	469.8	515	9	10	10	6	7	7	12.5	10.71	10.71
1400	478	554	674	10	12	13	7	9	9	10.71	8.33	8.33
1450	500	583	668.16	13	15	13	8	10	9	9.37	7.5	8.33
1250_1	500	590	667.79	11	13	13	8	10	10	9.37	7.5	7.5
1250_2	500	590	667.79	11	13	13	8	10	10	12	9.6	9.6

In Table 6.1, the 1200kV , 1400kV and 1450kV motors along with all the propeller combinations are tested with 5000mAh battery. 1250_1 represents the RimFire 1250kV motor tested along with the 5000mAh battery. Similarly, 1250_2 means the same motor tested with 6400mAh battery.



Park 480 1320 KV (8x4) Thrust vs PWM

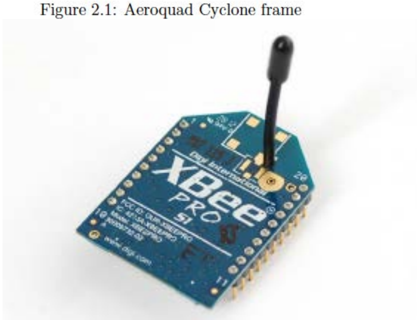


Park 480 1320 KV (8x4) Current vs PWM

Table 6.2: Result Summary

Motor (kV)	Hover Thrust(gm)	Max Thrust(gm)	Max current(A)	Hover current(A)	Hover flight time(min)
1200	340	515	10	7	10.71
1400	340	673	13	6	12.5
1450	340	661	13	6	12.5
1650	340	672	18	14	5.3
1320	375	920	16	5.5	13.6

Figure 2.1: Aeroquad Cyclone frame



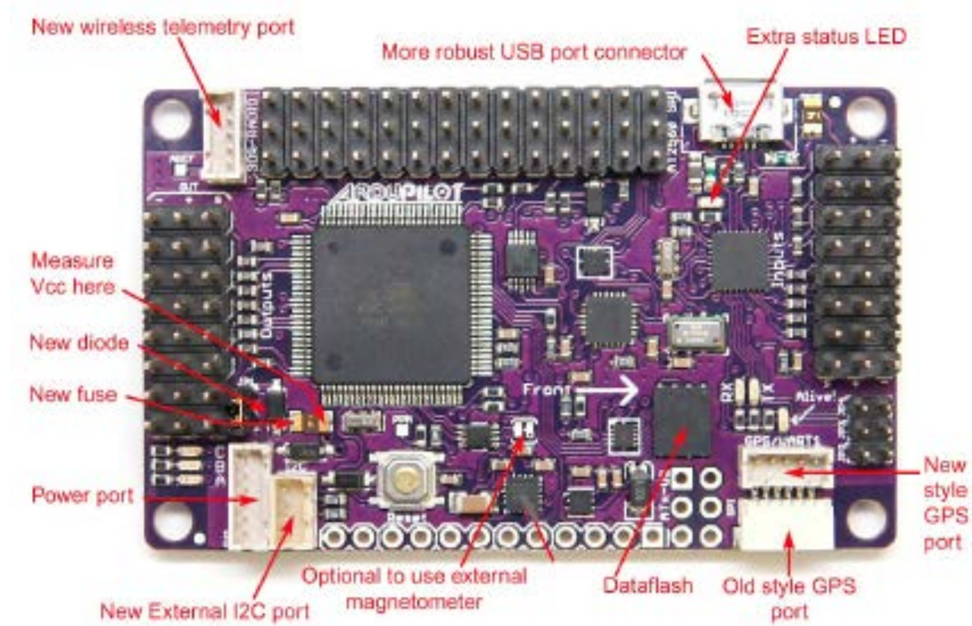


Figure 2.3: ArduPilotMega (APM) 2.6 [6].

Figure 2.4: Turnigy Park480 1320kv motors [7].

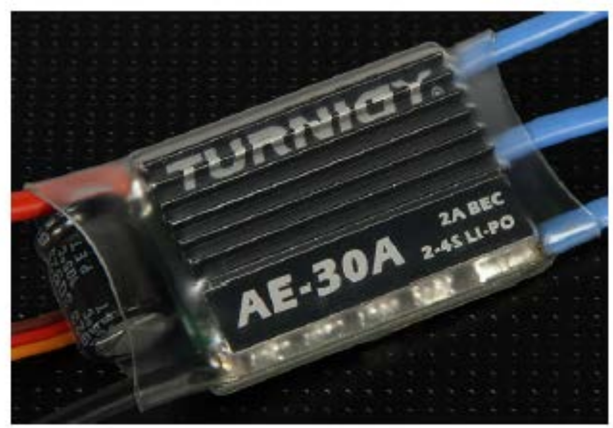


Figure 2.5: Turnigy AE 30A Brushless ESCs [8].

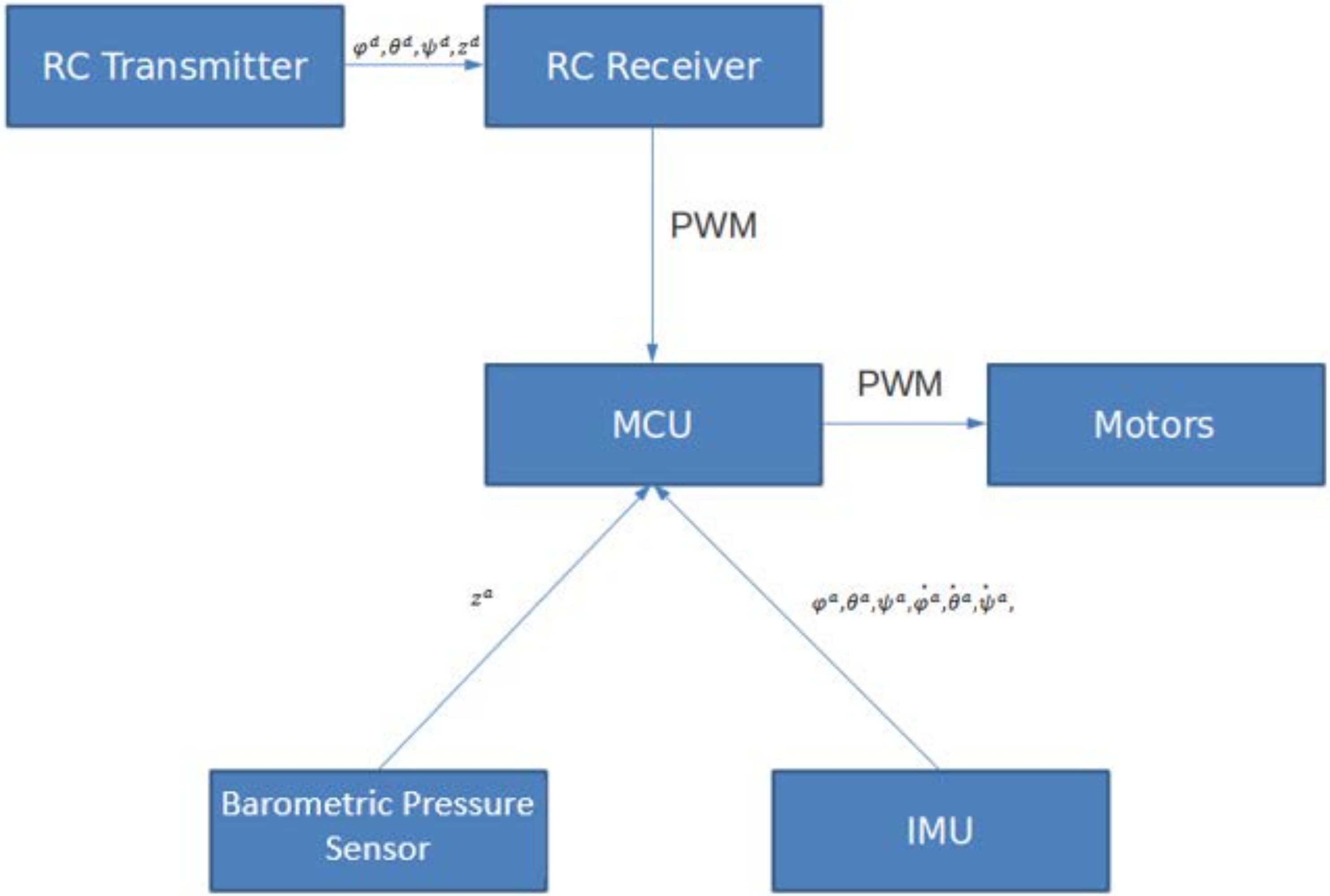
Hardware Specifications

- Aluminum/Carbon Fibre Structure
- Park 480 1320Kv motors
- Turnigy AE-30A Brushless ESC
- Atmega 2560 Microcontroller Unit
- Turnigy 9XR with Orange R800X 2.4 GHz Transmitter/Receiver Module
- MPU6000 IMU
- MS5611 barometer
- 3S, 11.1V, 5000mAh, 35-70C Turnigy Nanotech LiPo battery

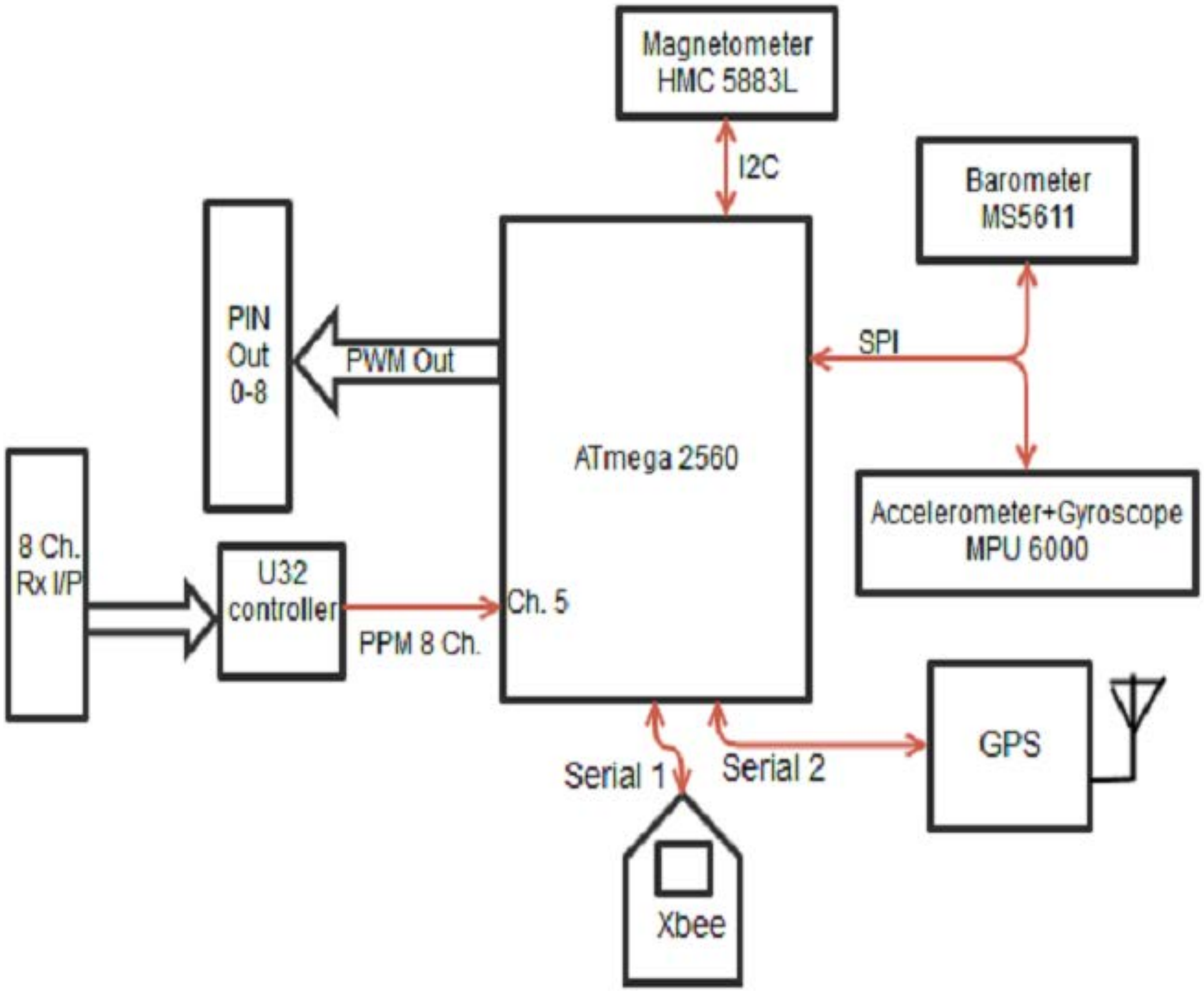
Table 6.3: Weight distribution for Design

Design	
Component	weight(in grams)
Battery	410
Frame	270
ESC	25×4=100
Spinner	10×4=40
Propeller	5×4=20
Motor	50×4=200
Other circuitry	460
Total	1500
Max Thrust achieved	$920 \times 4 = 3680$
Payload	2180
Flight time at hover thrust	13.6min
Flight time at max thrust	4.68min

Hardware Architecture



Hardware Architecture



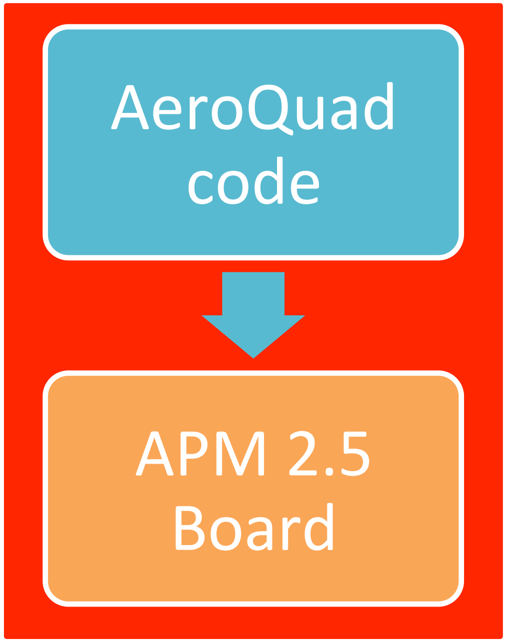
Software Platform

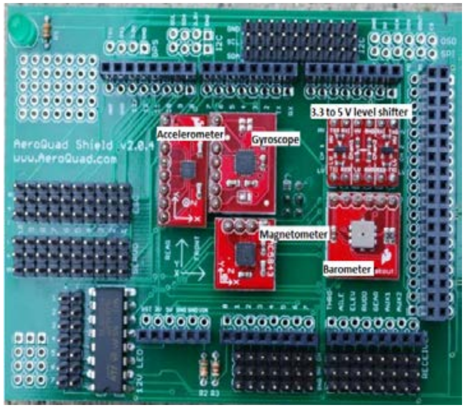
Software Platform Selection

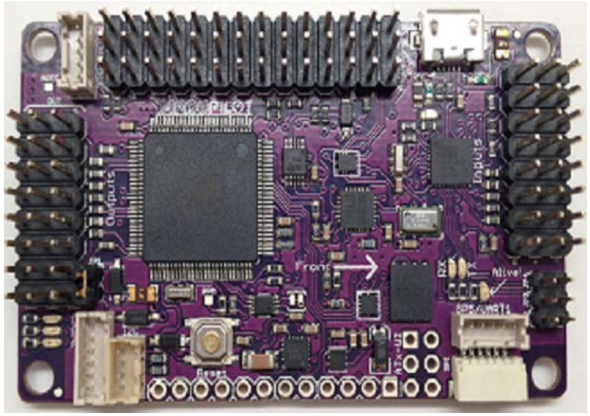
- Main aim: Testing of Cooperative Control strategies rather than developing whole software system
- Ease of coding and fast development:
  - Arduino selected over ARM platform
  - Open-source software over self developed code
  - AeroQuad v/s APM ArduPilot

AeroQuad v/s ArduPilot

	AeroQuad	**ArduPilot**
Modularity	Good	Poor
Readability	Good	Poor
Target Hardware	Multi-board	Compact
Target Processor	ATmega 2560	ATmega 2560
Hardware	Bulky, non-robust	**Compact, robust**



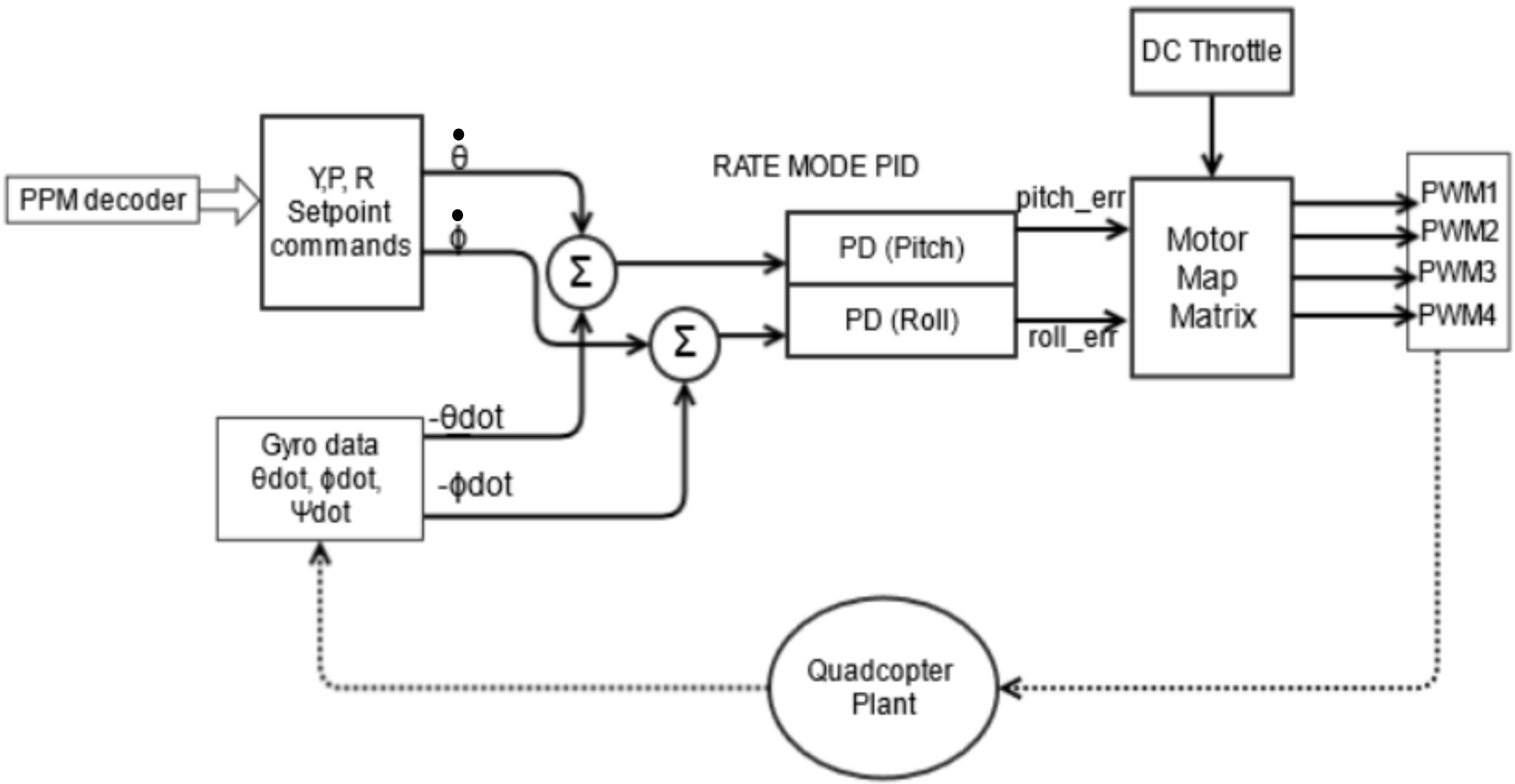




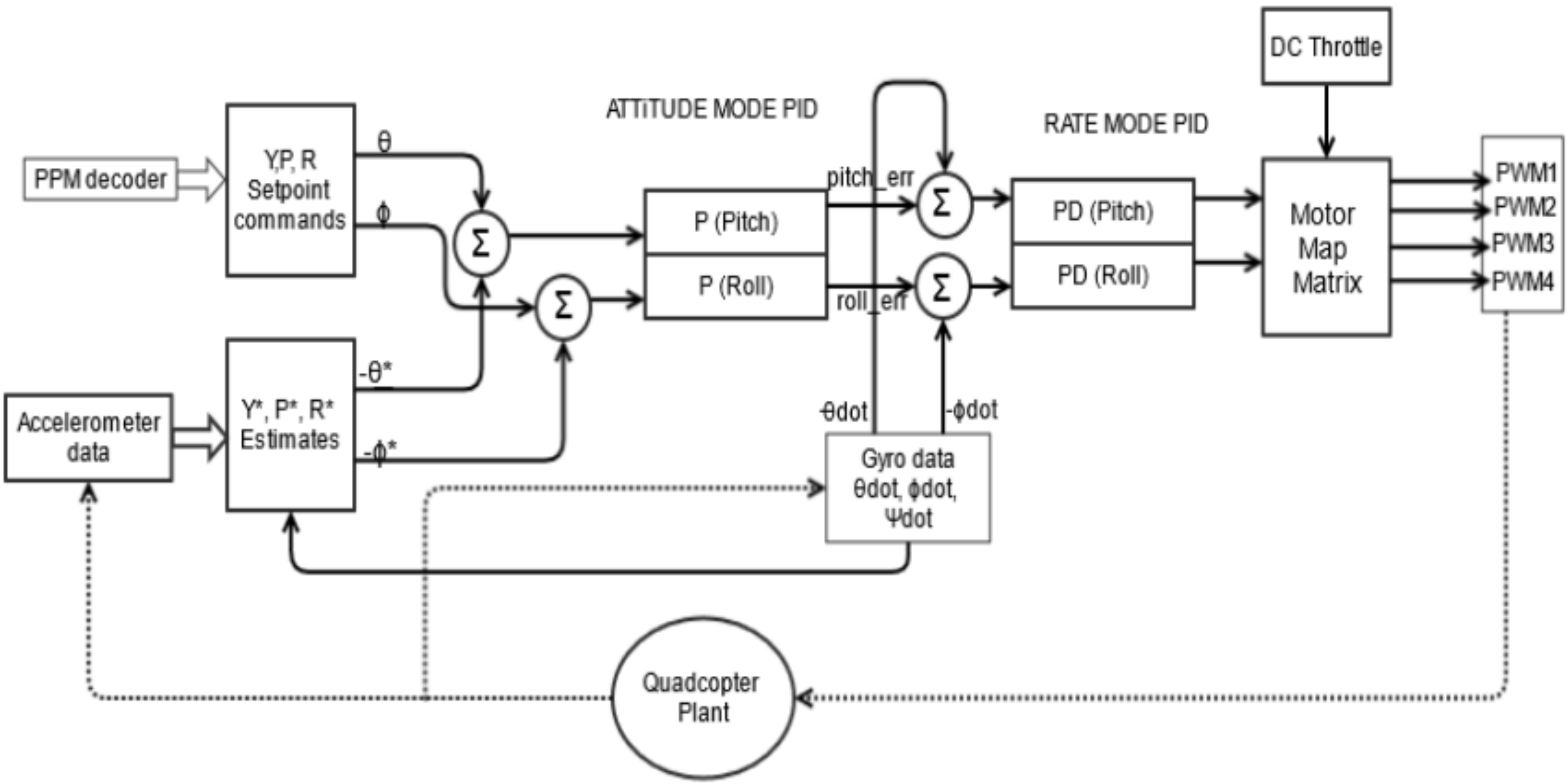
AeroQuad Configurator

Autopilot Architecture – Primary Modes

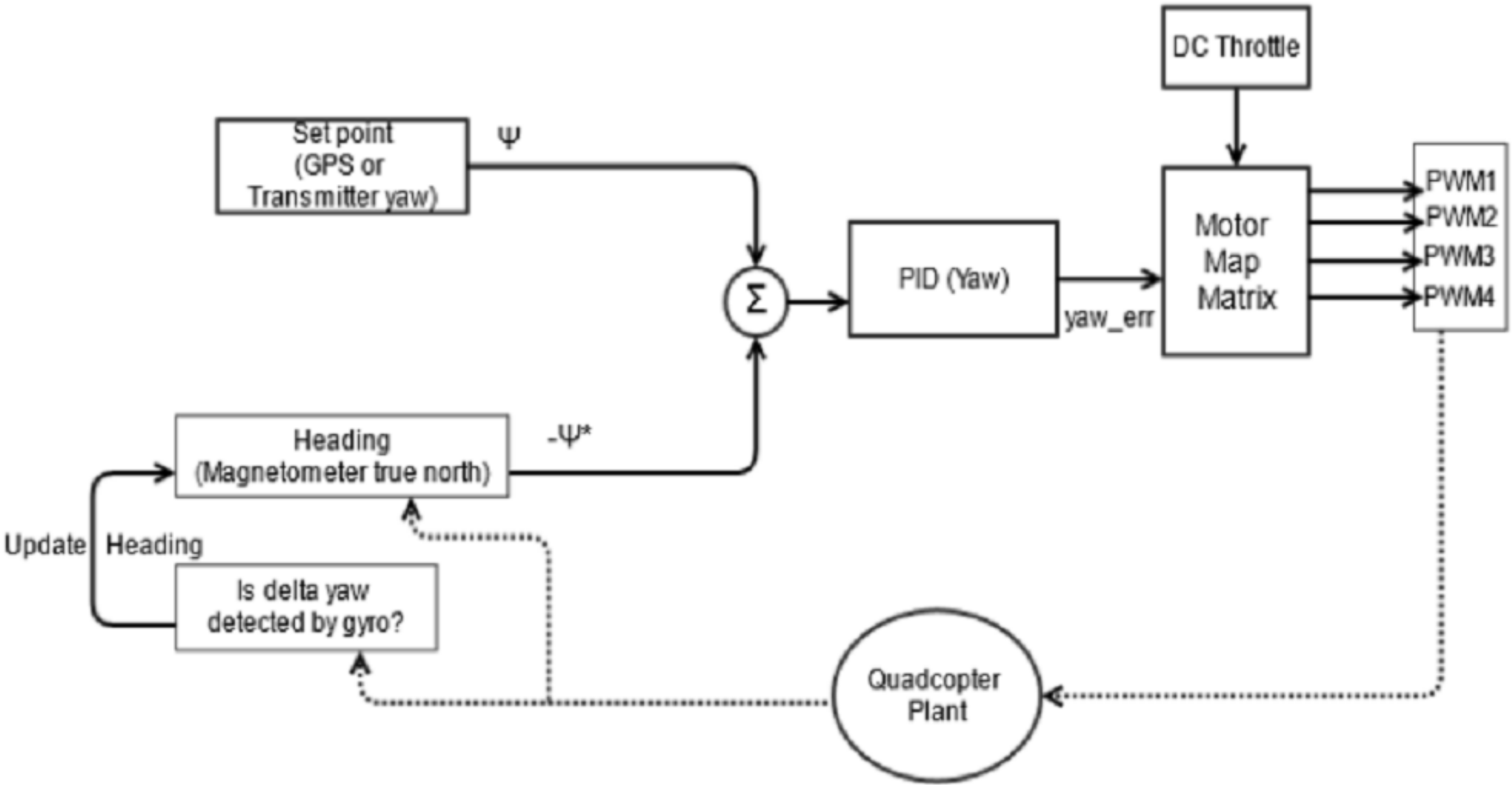
The Rate Mode Control Loop



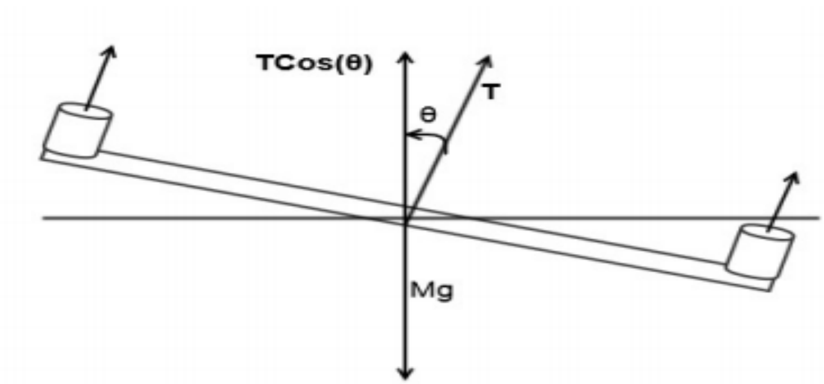
The Attitude Mode Control Loop



Heading Hold



Altitude Hold



$$T_{Corrected} \approx \frac{T_{Hover}}{\cos(\theta^{*})\cos(\phi^{*})}$$

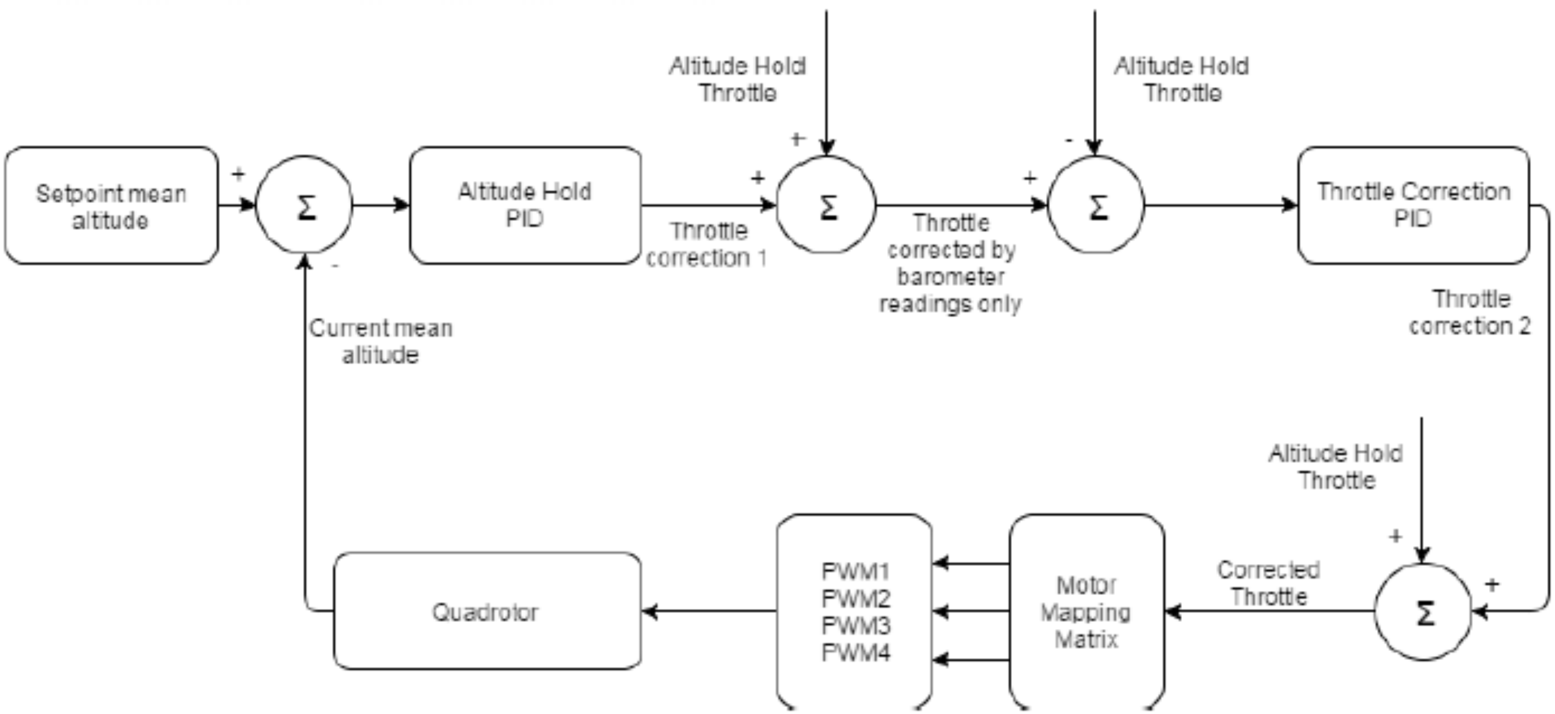


Figure 3.7: Experimental setup for Attitude mode gain tuning

Altitude Hold Mode

Quadrotor	K_{P1}	K_{I1}	K_{D1}	K_{P2}	K_{I2}	K_{D2}
Aeroquad Cyclone	75	2.5	1.1	0.75	1	0
HobbyKing x580 (Bravo)	60	2.5	1	0.05	0	0
HobbyKing x580 (Charlie)	60	2.5	1	0.05	0	0

Table 3.3: Altitude hold mode gains

Heading Hold Mode

Quadrotor	K_{P1}	K_{I1}	K_{D1}	K_{P2}	K_{I2}	K_{D2}
Aeroquad Cyclone	3	0.1	0	170	0	0
HobbyKing x580 (Bravo)	3	0.1	0	200	0	0
HobbyKing x580 (Charlie)	3	0.1	0	200	0	0

Table 3.4: Heading hold mode gains

Autonomous Modes

Waypoint Navigation

- Distance to Destination (D) and new heading are calculated as shown in Fig 11.

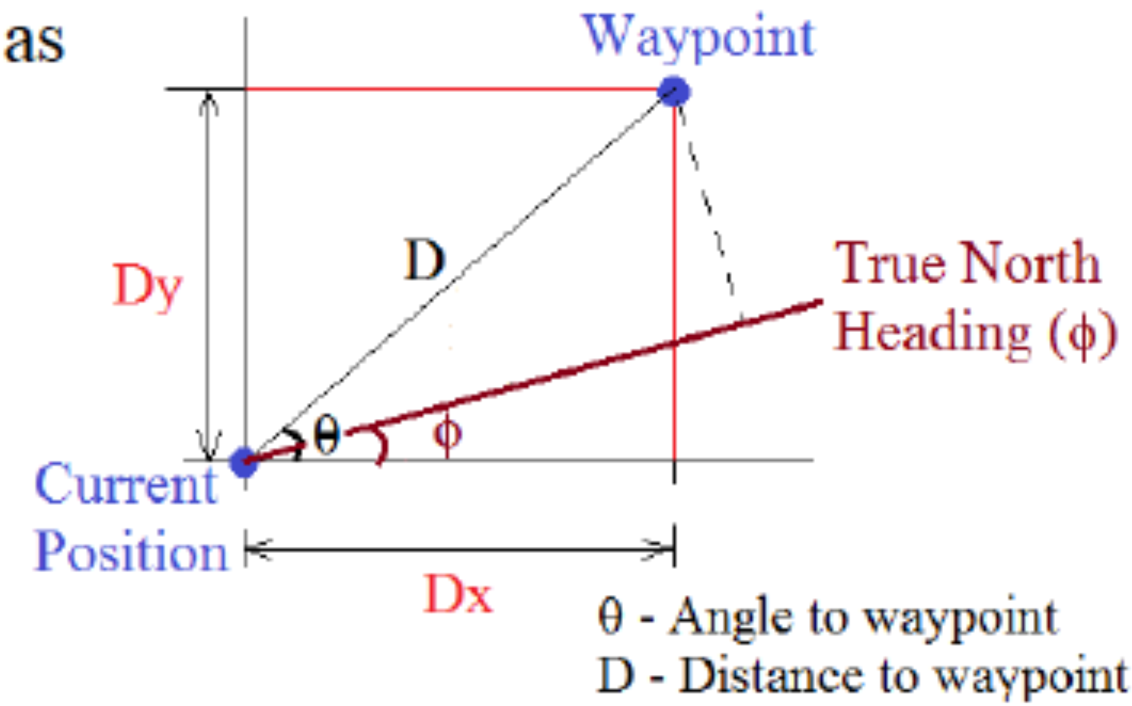


Fig 11 : Calculation of the heading during Navigation

- $D = \sqrt{(D_x^2 + D_y^2)}$
- $\tan\theta = \frac{D_y}{D_x}$
- $r_s = DN_s \sin\theta - \Phi$
- $p_s = DN_s \cos\theta - \Phi$
- The 'r_s' and 'p_s' variables are correction factors which are used along with the PI gains to correct the pitch and roll angles according to the destination. 'N_s' is a constant

Waypoint Navigation

•Estimates of speed & position are based on current and last GPS data and time elapsed.

•The new mission position is updated when the Quadcopter reaches within the circle of radius (Dmin = 2.5 m) centered at the waypoint.

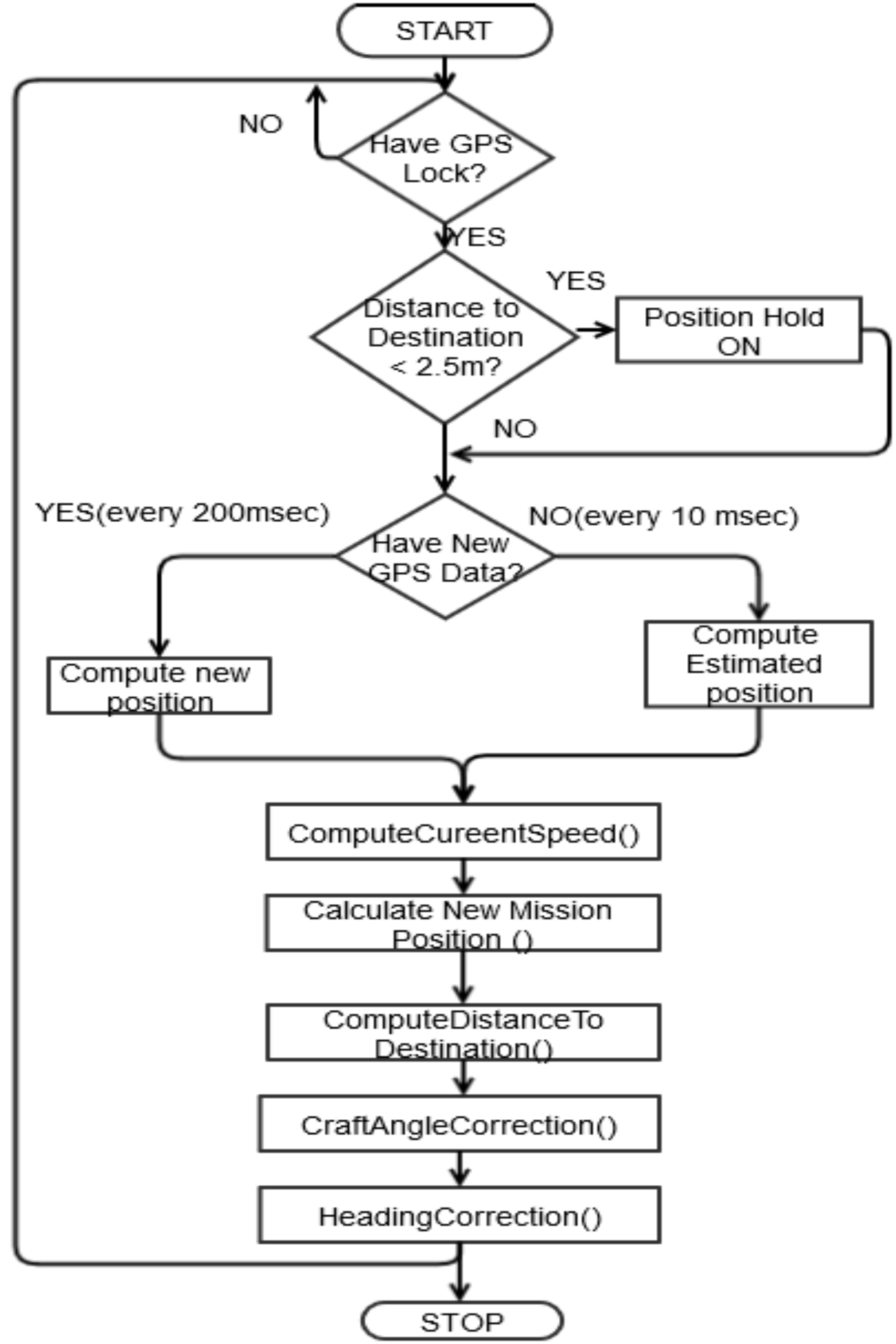
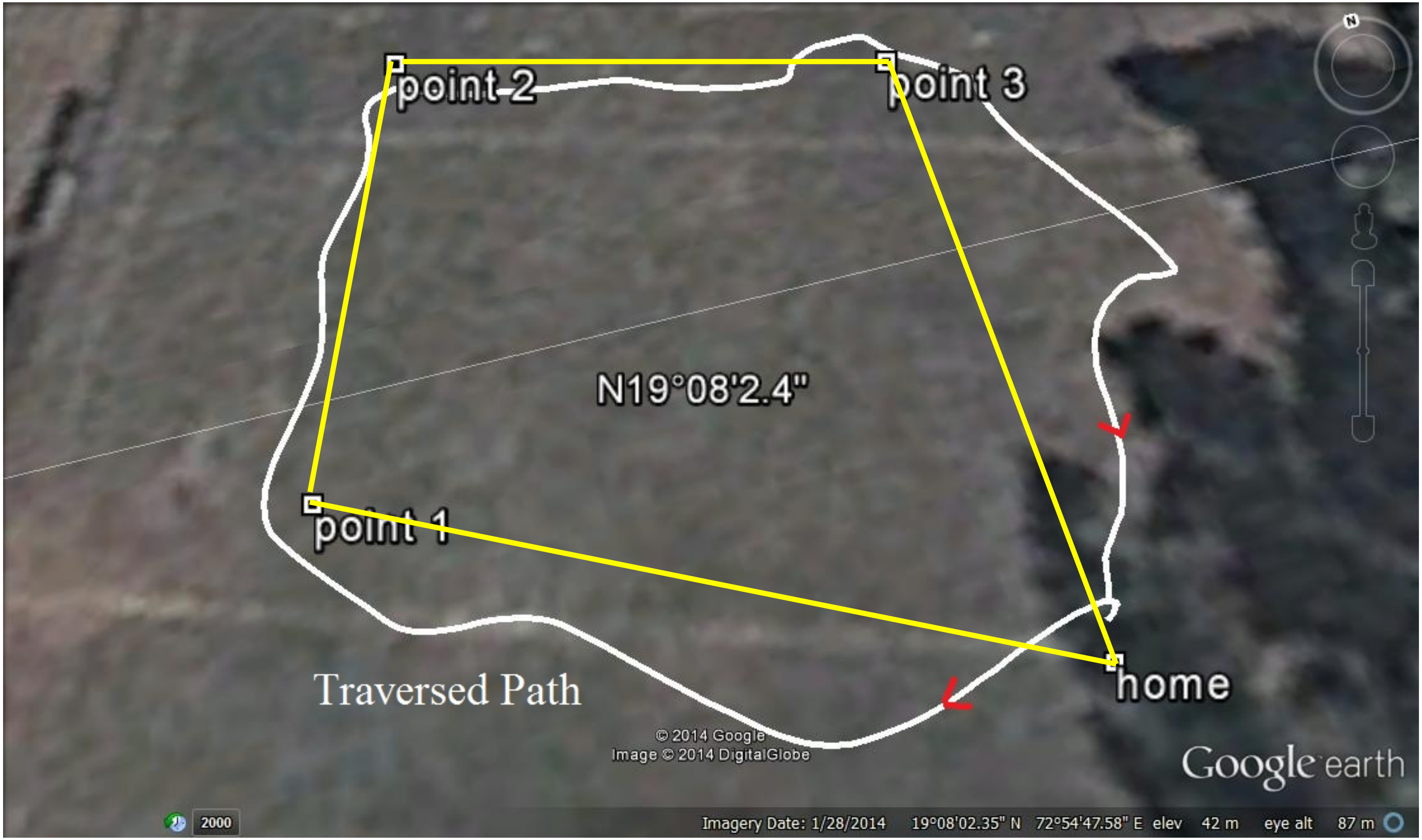


Fig 10 : GPS Navigation flowchart

GPS Position Hold

Fig: Google earth plot for GPS position hold mode

Waypoint Navigation



Leader – Follower Configuration

Waypoint Navigation to Leader's Coordinates

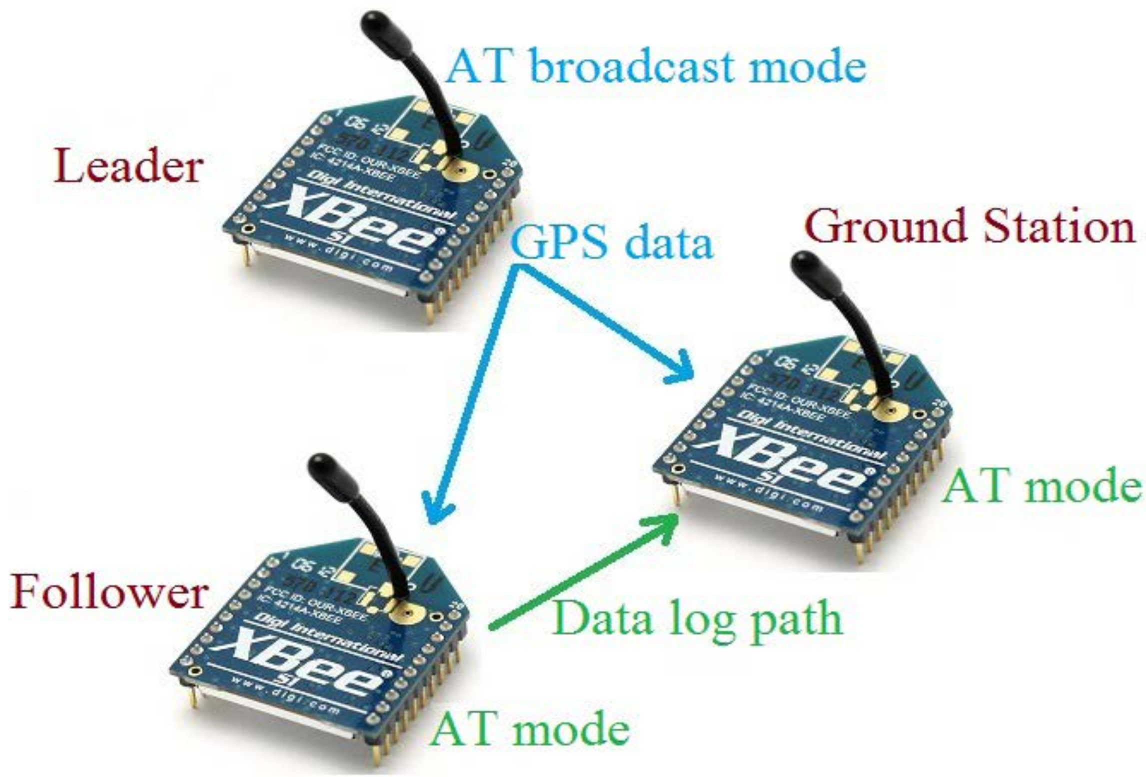


Fig 16 : Xbee configuration for Co-operative Control

Followers converging to leader's position

Leader – Follower Configuration Both Moving

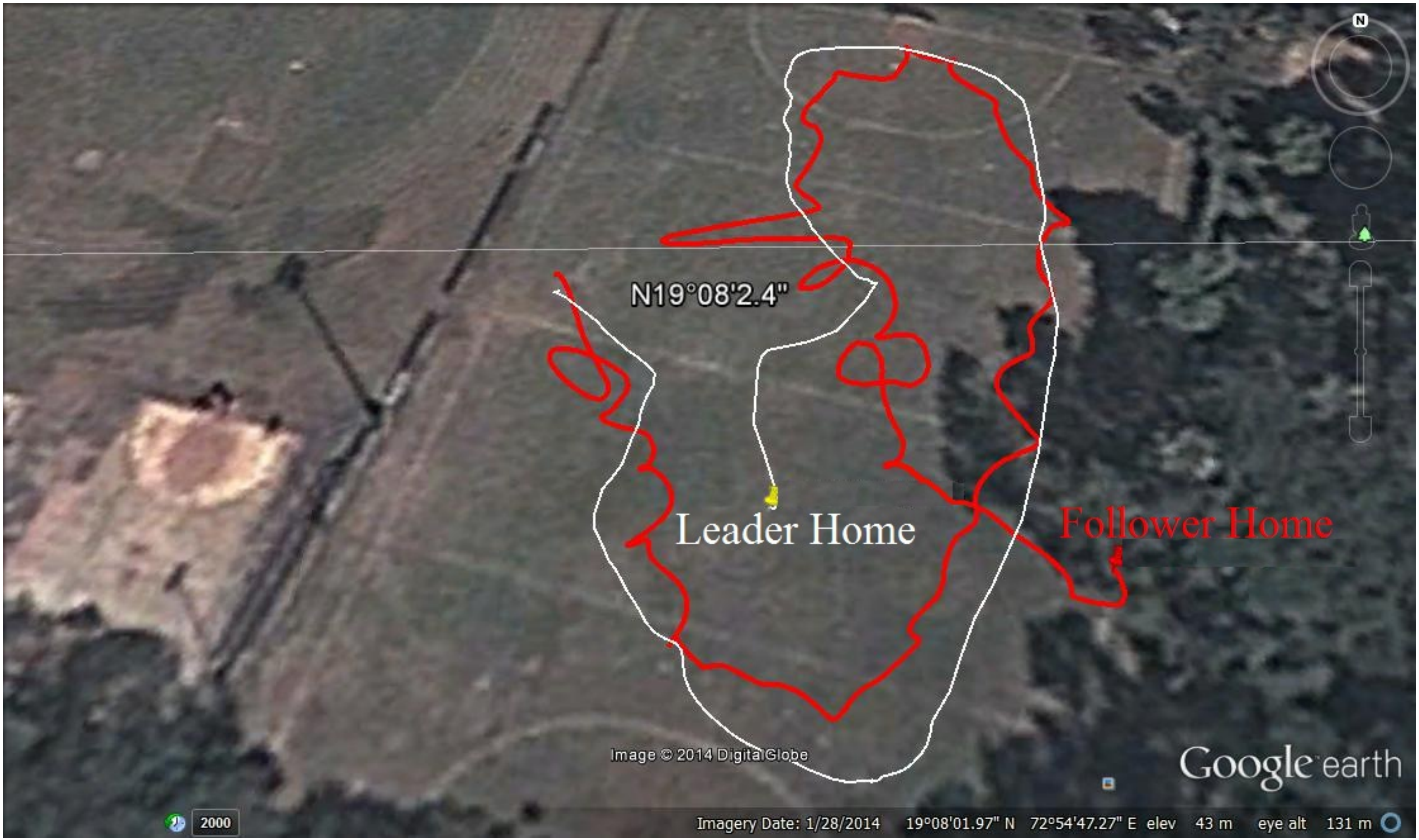


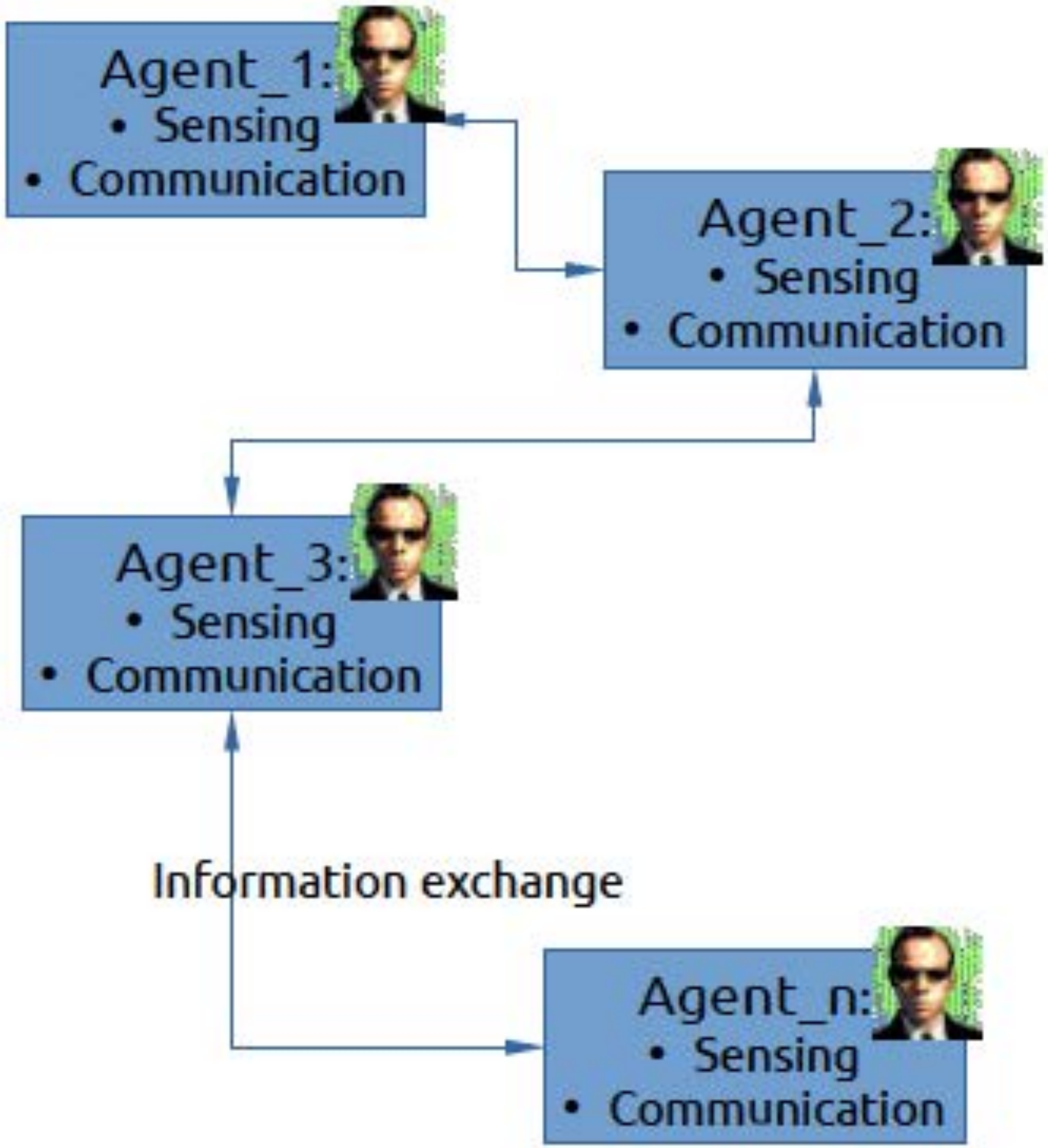
Fig 18 : Co-operative Control plot with leader-follower configuration

Part 2

Cooperative Control Experiments

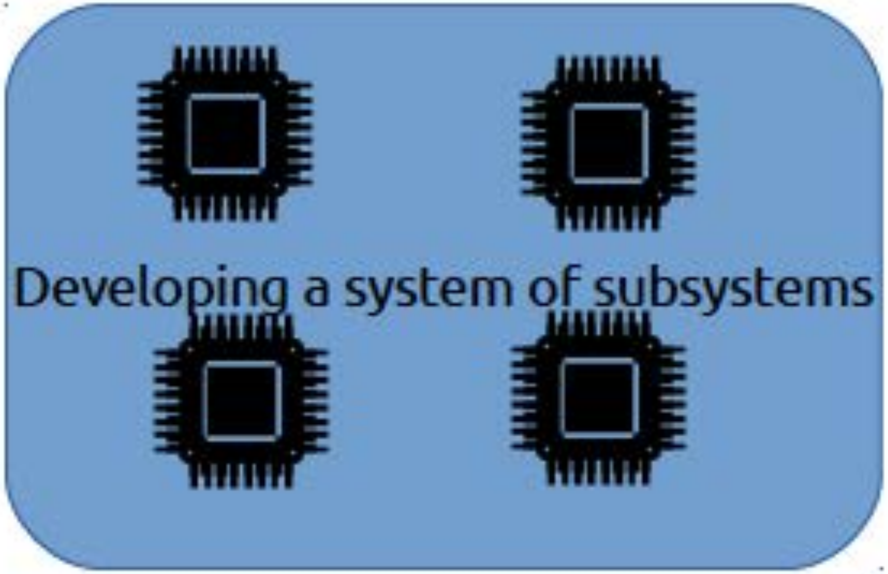
- Frames of reference
- Inner Control loops
- Waypoint navigation
- Consensus law
- Communication Protocols
- Experiments

Motivation

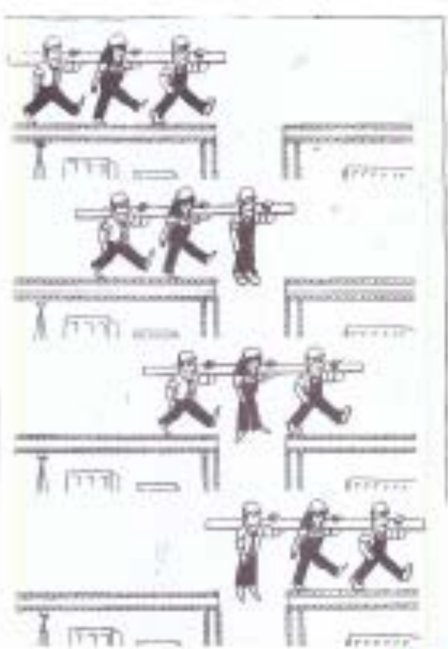


- Teams
  - Individual agents acting together to achieve a common goal
- Cooperation
  - Sensing
  - Information exchange
- Goal: consensus
  - Flocking
  - Rendezvous

Challenges



- Limited communication bandwidth and connectivity
- What to communicate and when?



- Arbitration between team and individual goals

Hence decentralized cooperative control!

What have we achieved?

- Experimental verification of a decentralized consensus law for double integrators on three physical quadrotors.
  - Run 1 Communication using the Zigbee protocol in broadcast mode (ECC 2016)
  - Run 2 Communication using an indigenously developed protocol with time-slotting for improved efficiency
- Under mild assumptions, shown that quadrotors can be modelled as double integrators

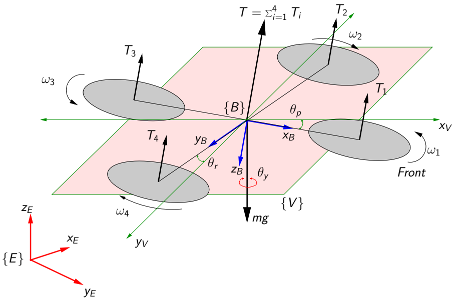
State of the art

- Sophisticated theory for multiagent consensus (single and double integrator agents) already developed
- Practical implementation:
  - Indoors, with motion capture cameras
  - Using centralized control algorithms
- COLLMOT group, Eötvös University, Hungary: distributed, empirical consensus law without proof of convergence

We experimentally verified a theoretically provable consensus law

Quadrotor dynamics

Frames of reference:



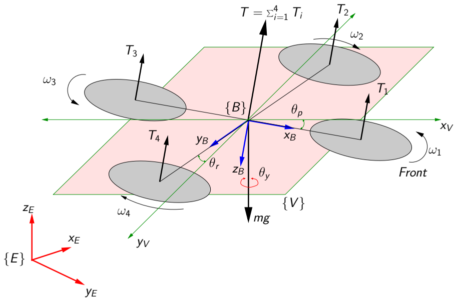
Earth frame, $\{E\}$

- x_E- axis points in the north direction
- y_E- axis points in the east direction
- z_E- axis points in the up direction

Use: Analysis of consensus of agents done in the Earth frame

Quadrotor dynamics

Frames of reference:



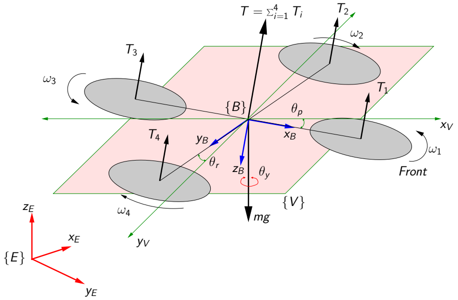
Body frame, $\{B\}$

- x_B- axis points towards front end
- y_B- axis points towards right end
- z_B- axis points downwards

Use: All state measurements done with respect to the body frame

Quadrotor dynamics

Frames of reference:



An auxiliary frame, {V},

- same origin as $\{B\}$
- z_V axis is parallel to z_E- axis
- x_V, y_V- axes are projections of x_B, y_B onto a plane parallel to the $x_E y_E-$ plane in $\{E\}$ and passing through the origin of $\{V\}$.

Use: Helps in approximating the quadrotors as double integrators

Quadrotor dynamics

- Motion along six degrees of freedom achieved by varying rotor speeds, $\bar{\omega}_i$
- Generating pairwise difference in rotor thrusts leads to rotational motion
-
$$\begin{bmatrix} T \\ \tau_x \\ \tau_y \\ \tau_z \end{bmatrix} = \begin{bmatrix} -b & -b & -b & -b \\ 0 & -db & 0 & db \\ db & 0 & -db & 0 \\ k & -k & k & -k \end{bmatrix} \begin{bmatrix} \bar{\omega}_1^2 \\ \bar{\omega}_2^2 \\ \bar{\omega}_3^2 \\ \bar{\omega}_4^2 \end{bmatrix} = A \begin{bmatrix} \bar{\omega}_1^2 \\ \bar{\omega}_2^2 \\ \bar{\omega}_3^2 \\ \bar{\omega}_4^2 \end{bmatrix} \qquad (1)$$

  where,
  - T is thrust generated, $\begin{bmatrix} \tau_x & \tau_y & \tau_z \end{bmatrix}^T$ are the torques generated
  - b,k: constants
  - d: distance of the motor from the CoG of the quadrotor

Control loops

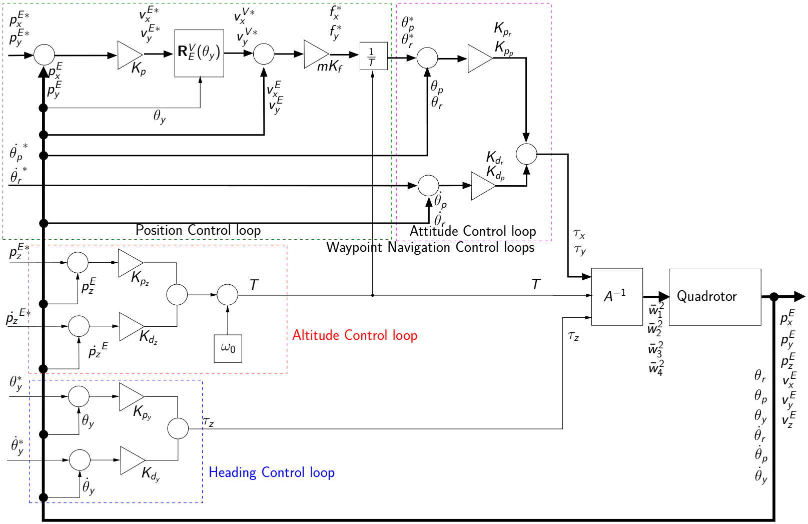


Figure: A block diagram of the quadrotor control loops

Waypoint navigation

- Process by which the quadrotor navigates to different positions $\mathbf{p}^E = \begin{bmatrix} p_x^E & p_y^E \end{bmatrix}^T \in \mathbb{R}^2$ in $\{E\}$
- Generate τ_x and τ_y to vary θ_p and θ_r and thus maneuver the quadrotor
- Keep θ_y constant using heading control loop.

Waypoint navigation

- Consider the frame $\{V\}$. To accelerate along x_V- and y_V- axes, we need to generate forces

$$
\begin{aligned}
f_x^V &= T\sin\theta_p \approx T\theta_p && (2)\\
f_y^V &= T\sin\theta_r\cos\theta_p \approx T\theta_r. && (3)
\end{aligned}
$$

for small θ_x and θ_y

- We control the motion using a PD law

$$
\mathbf{f}^{V*} = mK_f[K_p(\mathbf{p}^{V*} - \mathbf{p}^V) - \mathbf{v}^V] \tag{4}
$$

where $\mathbf{f}^V = \begin{bmatrix} f_x^V & f_y^V \end{bmatrix}^T$

- Then, desired values of angles, $\boldsymbol{\Theta}^* = \begin{bmatrix} \theta_p^* & \theta_r^* \end{bmatrix} \in \mathbb{R}^2$ are

$$
\boldsymbol{\Theta}^* = \frac{mK_f}{T}[K_p(\mathbf{p}^{E*} - \mathbf{p}^E) - \mathbf{v}^E] \tag{5}
$$

Waypoint navigation

- To attain $\mathbf{\Theta}^* = \left[\begin{array}{cc} \theta_p^* & \theta_r^* \end{array} \right] \in \mathbb{R}^2$, generate torques $\tilde{\mathbf{\Gamma}} = \left[\begin{array}{cc} \tau_x & \tau_y \end{array} \right]^T \in \mathbb{R}^2$ using a PD controller

$$\tilde{\mathbf{\Gamma}} = K_{p_{r,p}}(\mathbf{\Theta}^* - \mathbf{\Theta}) + K_{d_{r,p}}(\dot{\mathbf{\Theta}}^* - \dot{\mathbf{\Theta}}) \tag{6}$$

  where $K_{p_{r,p}} = \left[\begin{array}{cc} K_{p_r} & K_{p_p} \end{array} \right]^T \in \mathbb{R}^2$ and $K_{d_{r,p}} = \left[\begin{array}{cc} K_{d_r} & K_{d_p} \end{array} \right]^T \in \mathbb{R}^2$ are the control gains.
- Controller designed such that $\theta_p \to \theta_p^*$ and $\theta_r \to \theta_r^*$ almost immediately

Waypoint navigation

In the $\{E\}$ frame,

$$\mathbf{f}^E = \mathbf{R}_V^E \mathbf{f}^V \tag{7}$$

and

$$\begin{aligned} f_x^V &= T \sin\theta_p \approx T\theta_p & (8) \\ f_y^V &= T \sin\theta_r \cos\theta_p \approx T\theta_r. & (9) \end{aligned}$$

for small θ_x and θ_y

If we can vary θ_p and θ_r independently and instantaneously, then motion in the $x_E y_E-$ plane can be modelled as a double integrator.

Waypoint navigation

- We vary θ_p and θ_r independently and quickly such that change in angle is much faster than translational motion
- As θ_p and θ_r change, the vertical component of T reduces by a factor of the cosine of θ_p and θ_r
- But θ_p and θ_r are small and altitude control loop is fast

Hence the quadrotor can be modelled as a double integrator

$$\dot{\mathbf{p}}^E = \mathbf{v}^E, \qquad \dot{\mathbf{v}}^E = \mathbf{f}^E \tag{10}$$

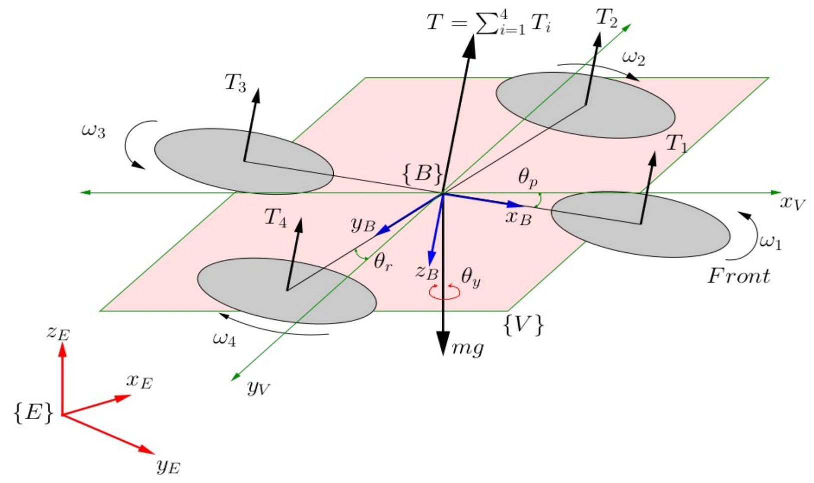
where

- $\mathbf{p}^E = \begin{bmatrix} p_x^E & p_y^E \end{bmatrix}^T \in \mathbb{R}^2$ is the position in $\{E\}$
- $\mathbf{v}^E = \begin{bmatrix} v_x^E & v_y^E \end{bmatrix}^T \in \mathbb{R}^2$ is the velocity in $\{E\}$
- $\mathbf{f}^E = \begin{bmatrix} f_x^E & f_y^E \end{bmatrix}^T \in \mathbb{R}^2$ is the acceleration input in $\{E\}$

Consensus law

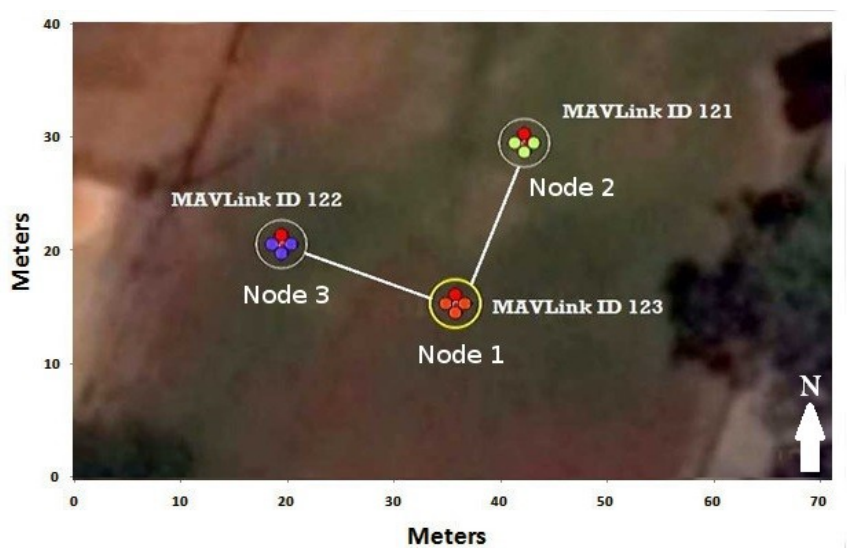
- If, for all $\mathbf{p}_i^E(0)$ and $\mathbf{v}_i^E(0)$ and all $i, j = 1, ..., n$, $\|\mathbf{p}_i^E(t) - \mathbf{p}_j^E(t)\| \to 0$ and $\mathbf{v}_i^E \to 0$ as $t \to \infty$ then consensus achieved
- Information exchange modelled as undirected graph $\mathcal{G}_n := (\mathcal{V}, \mathcal{E})$ where $\mathcal{V} = \{1, ..., n\}$ is the set of nodes and $\mathcal{E} \subseteq (\mathcal{V} \times \mathcal{V})$ is the set of edges
- Node $\equiv$ quadrotor, edge $\equiv$ available communication channel
- Set of neighbours, $\mathcal{N}_i := \{j \in \mathcal{V} : (i, j) \in \mathcal{E}\}$.
- Laplacian matrix $\mathcal{L}_n$ of a graph $\mathcal{G}_n$ is given by $\mathcal{L}_n = [l_{ij}] \in \mathbb{R}^{n \times n}; l_{ij} = -a_{ij}, i \neq j, l_{ii} = \sum_{j=1, j \neq i}^{n} a_{ij}$.

Quadrotor team

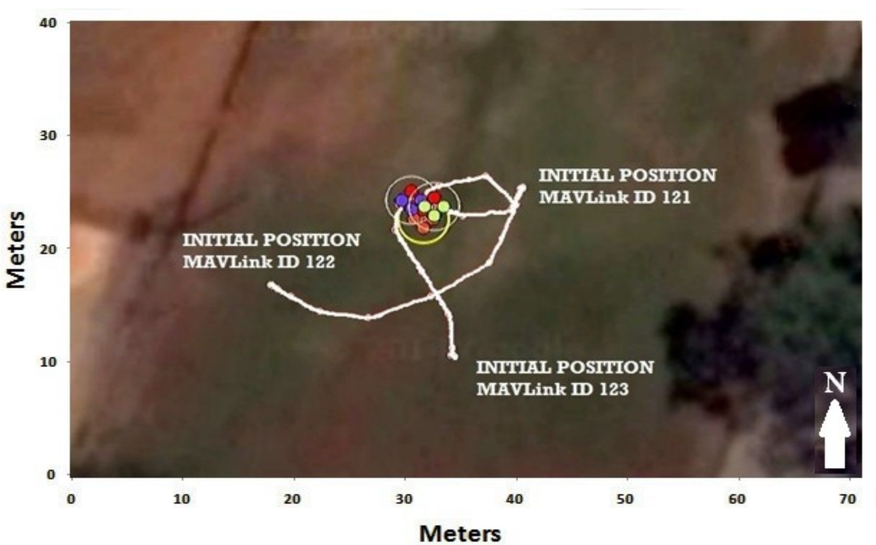


Each quadrotor obeys the control law

$$\mathbf{f}_i^E = \sum_{j \in \mathcal{N}_i} a_{ij}(\mathbf{p}_j^E - \mathbf{p}_i^E) - \beta \mathbf{v}_i^E, \qquad i = 1, \ldots, n$$



Initial states



Consensus!

Decentralized cooperative control of quadrotor teams

Consensus law

Theorem

Given a system

$$\dot{\mathbf{p}}^E = \mathbf{v}^E, \qquad \dot{\mathbf{v}}^E = \mathbf{f}^E \tag{12}$$

The control law [1],

$$\mathbf{f}_i^E = \sum_{j \in \mathcal{N}_i} a_{ij}(\mathbf{p}_j^E - \mathbf{p}_i^E) - \beta \mathbf{v}_i^E, \qquad i = 1, ..., n$$

achieves consensus asymptotically iff $\mathcal{G}_n$ is connected

As a result

- $\mathbf{p}(t) \to (\beta \mathbf{1}_n \mathbf{1}_n^T \otimes I_2)\mathbf{p}(0) + (\mathbf{1}_n \mathbf{1}_n^T \otimes I_2)\mathbf{v}(0)$
- $\mathbf{v}(t) \to 0$ as $t \to \infty$

Hence $\|\mathbf{p}_i^E(t) - \mathbf{p}_j^E(t)\| \to 0$ and $\mathbf{v}_i^E \to 0$ as $t \to \infty$ for all $i, j = 1, ..., n$

[1] Proof similar to W. Ren, R. Beard, Distributed consensus in multi-vehicle cooperative control, Springer, 2008

Experiment overview

- Each quadrotor shares its position information, $\mathbf{p}^E = \begin{bmatrix} p_x^E & p_y^E \end{bmatrix}^T$ which is measured by the on-board GPS receiver with its neighbours
- Efficient communication mechanism between agents vital for the successful demonstration of the consensus law

Run 1

- Zigbee protocol in broadcast mode
- Reliable data transmission and reception hampered by data collisions

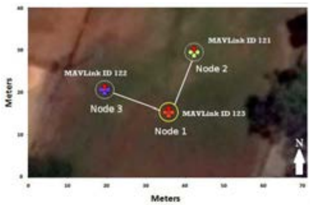
Run 2

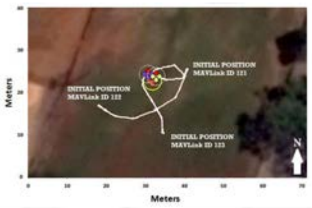
- Indigenously developed protocol with time-slotting for improved efficiency
- Can deal with node failure

Run 1

Zigbee protocol in broadcast mode

Consensus Video

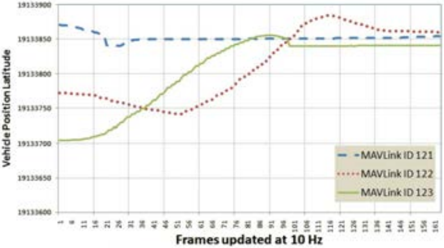


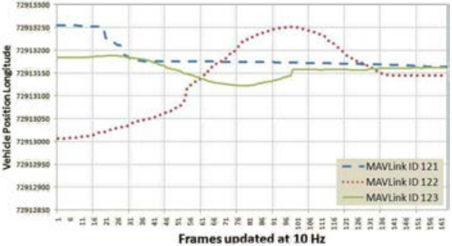


Run 1: Position plots

Zigbee protocol in broadcast mode

Latitude and Longitude tracking

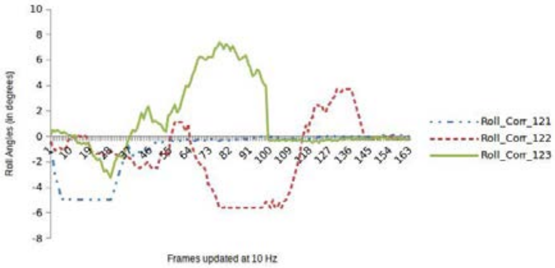


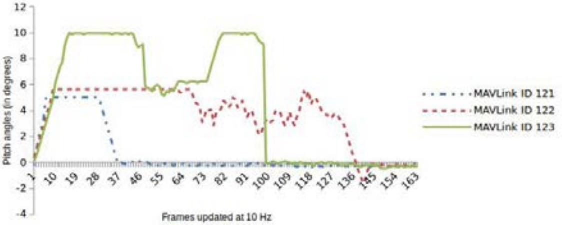


Run 1: Angle correction plots

Zigbee protocol in broadcast mode

Roll and pitch angle correction

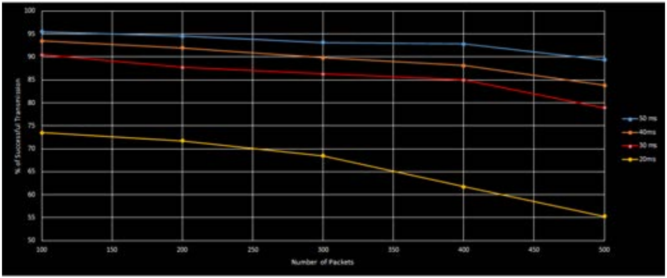


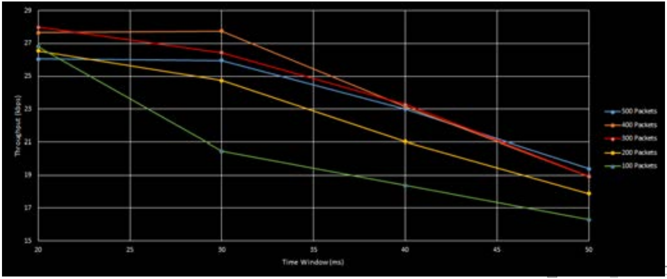


Run 1: Communication protocol efficiency

Zigbee protocol in broadcast mode

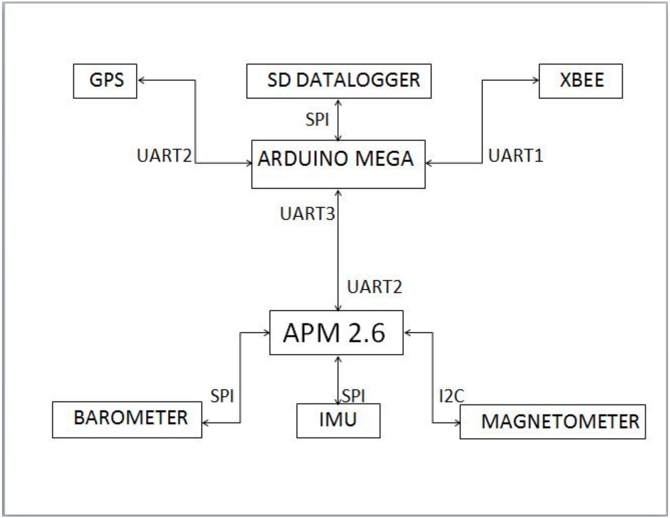
% Successful transmission and and Throughput





Run 2: Hardware Architecture

Indigenously developed protocol



Run 2: Features

Indigenously developed protocol

Time synchronization Correction for clock drift

Slotting into frames Every node assigned time slot to send its location packet

Link break control To remove dependency on master node for synchronization

Run 2: Consensus Plot

Indigenously developed protocol

Figure: Three nodes reaching consensus successfully

Run 2: Relative position error, Node 1

Indigenously developed protocol

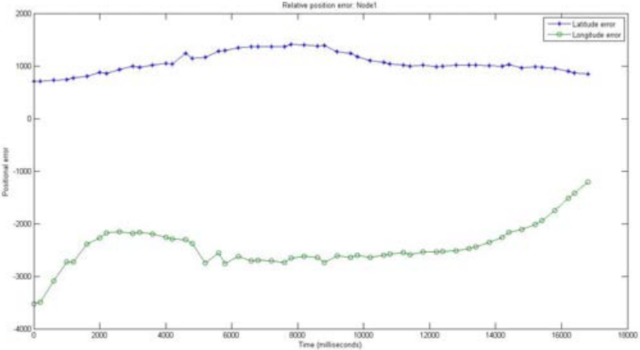


Figure: Relative position error for Node 1

Run 2: Relative position error, Node 2

Indigenously developed protocol

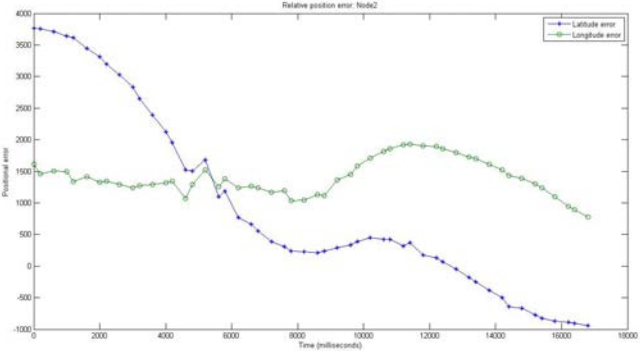


Figure: Relative position error for Node 2

Run 2: Relative position error, Node 3

Indigenously developed protocol

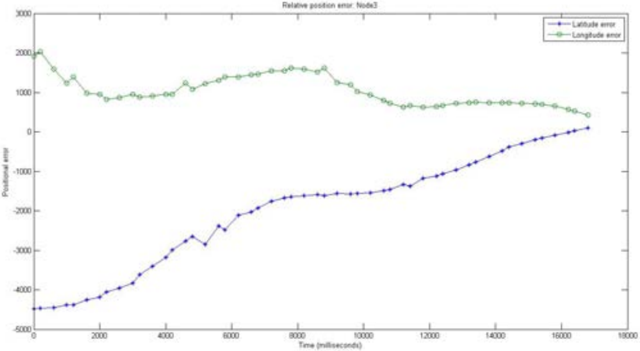
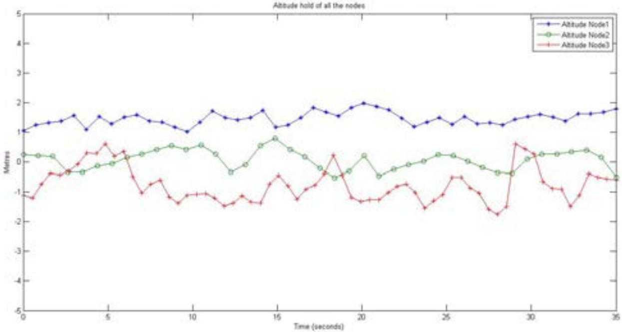


Figure: Relative position error for Node 3

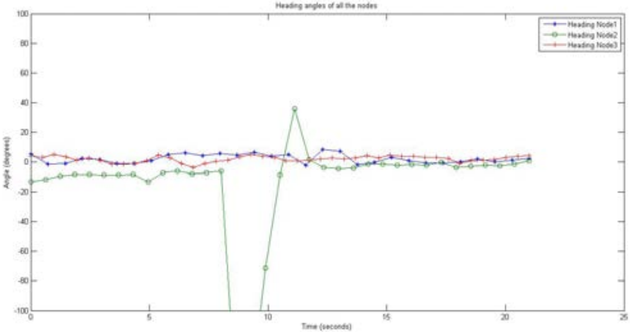
Run 2: Altitude Hold, all nodes

Indigenously developed protocol



Run 2: Heading Hold, all nodes

Indigenously developed protocol



Conclusion

Experimentally verified a decentralized consensus law wherein

- On-board controllers take navigation decisions by communication with its neighbours $\implies$ decentralized!
- Justification for approximating the quadrotor as two independent double integrators acting along the x and $y-$ axes of motion
- Outdoor environment $\implies$ inherent GPS errors. However, the quadrotors still successfully managed to reach consensus.

Future Work

Extension to a bigger system:

- Repeat the experiment with five physical quadrotors using our newly developed communication protocol.
- Implementing our existing consensus theory in the Robot Operation System (ROS) environment on the virtual quadrotors.
- Integrating the physical and virtual quadrotors to perform in a common environment.
- Develop algorithms to
  - Help choose the "ideal" leader(s) so that the whole group of agents is driven to consensus faster or with minimum fuel expenditure

Future Work

Implementation of time-optimal consensus laws: Monash Swarm Robotics Laboratory

- Indoor quadrotor testbed; No external disturbances
- Motion sensor cameras: precise localization of the agents
- Aim to test agressive time-optimal consensus laws in this indoor environment.

Thank you :)

Questions?