Meal Delivery with Unmanned Aerial vehicles

Akshit Jain
jaina488@gmail.com

Anil Loya
me12b1002@iith.ac.in

February 15, 2020

Abstract

Meal delivery is growing 300% faster than dine-in traffic. Current meal delivery models involves a person delivering a meal on a bike or car. The weight overhead (vehicle + person) and lack of autonomy leads to energy ineffeciency and high delivery cost. Additionally, use of ground vehicle contributes to congestion on roads. We propose hybrid Vertical take off and landing(VTOL) unmanned aerial vehicle(UAV) for completely autonomous operations. We propose rooftop landing for safety, hybrid fixed wing platform for effecient forward flight, frequent charge model to reduce battery weight and visual navigation, mapping and obstacle avoidance to reduce sensor weight and reliability. Design of UAV and cost analysis is presented for proposed model. We also survey current state of aerial platforms, battery technologies, sensors, perception algorithms prevalent to general aerial use cases.

Contents

1 Introduction

1.1 Motivation

In logistics a special class of goods exists which contributes majorly to its total delivery cost not due to its weight but due to its time criticality.

A life threatening example includes Medical supplies eg. red blood cells (RBCs), Plasma, Cryoprecipitate, platelets, vaccines, diagnostic samples [1]. These can be required during life threatening scenarios such as combat casualty [2], Hemorrhage during childbirth [3] and Vaccines during disasters. Time criticality dictates the requirements for closer last mile hubs. Additionally Blood products have very low shelf life($\sim$42 days) [4] and special storage conditions eg. -18 ℃ for Cryoprecipitate [4] which leads to inefficient distribution and wastage. Companies like Zipline[5], Wingcopter[6], DHL[7] send medicines and vaccines to remote villages where road transport is too slow.

Much less life threatening but wider use case is of meal delivery. Online meal ordering is growing 300% faster than dine-in traffic [8] since 2014 in USA. In India, food delivery platforms UberEats, Zomato and Swiggy combined serves on avg. $\sim$4 millions orders daily [9]. Online food delivery market is projected to reach $134.5 billion by 2023 [10].

Current model of delivery involves a person delivering a meal(usually singular) on a ground vehicle (bike or car). Energy inefficiecy is evident as overhead of $\sim$ 180 Kg (person + bike) needs to undergo trip to deliver relatively small meal, say pizza $\sim$ 0.6 Kg [11]. Major cost is contributed by human workforce due to absence of autonomy. Additionally, use of ground vehicles contributes to traffic congestion. Radius of operation(distance b/w user and restaurants) for restaurants remains small ($\sim$5km) constraint put by delivery time($\sim$35 mins) [12].

Above inefficiencies thus motivates delivery industry (Google Wing [13], Amazon Prime Air[14], Uber Elevate [15]) to adopt UAVs for last mile logistics of light goods.

1.2 Applications of UAVs

brief on UAV use cases in diff. industries. an image and line about motivation. eg. companies.

- aerial photography
- power line inspection

- traffic surveylence
- mine inspection
- oil pipes inspection
- wind turbine blade inspection
- agriculture (dusting and pesticide spraying)
- search and rescue

2 Problem Statement

Assume, in an urban environment, a food order is placed by user to a restaurant by some means (App, call, or scheduled). Once the order is received and prepared, it is dispatched from pickup location(restaurant) which will eventually reach customer.

Let,

$$
\begin{aligned}
t_{pl} &: \text{time of order } \mathbf{pl}\text{acement} \\
t_{pr} &: \text{time order was } \mathbf{pr}\text{epared} \\
t_{pk} &: \text{time of order } \mathbf{p}\text{ic}\mathbf{k}\text{up} \\
t_{dr} &: \text{time of order } \mathbf{dr}\text{op(received by user)}
\end{aligned}
$$

Needless to say,

$$
\begin{gathered}
t_j \in \mathbb{R}^+ \forall t_j \in \{t_{pl}, t_{pr}, t_{pk}, t_{dr}\} \\
\text{and} \\
0 \leqslant t_{pl} < t_{pr} \leqslant t_{pk} < t_{dr}
\end{gathered}
$$

Using boldface $\mathbf{t}_x$ to represent time span between two events,

$$
\begin{aligned}
\mathbf{t}_{tot} &: \mathbf{tot}\text{al time from order to drop} \\
&= t_{dr} - t_{pl} \\
\mathbf{t}_d &: \text{time taken after it was ready for } \mathbf{d}\text{elivery} \\
&= t_{dr} - t_{pr}
\end{aligned}
$$

Usually there is a constraint,

$$\mathbf{t}_{tot} < \mathbf{T}_{wait} \tag{1}$$

where, $\mathbf{T}_{wait}$ is the maximum time user can wait after placing an order. eg. it won't be acceptable if your food arrives an hour after you order. Less typically, there can also be constraint,

$$\mathbf{t}_{d} < \mathbf{T}_{stale} \tag{2}$$

where $\mathbf{T}_{stale}$ is the time after preparation, the item looses its integrity. eg. a pizza looses its integrity after, say 30 mins if reheating is not availiable. If we had single order, our goal would be to minimize the cost of delivery (we will define this cost later), while respecting constraints (1) and (2). Note that, we can only control $\mathbf{t}_d$ as t_{pr} is within restaurant's control. So, if $t_{pr} - t_{pl} > \mathbf{T}_{wait}$, then solution is trivially infeasible.

Concentrating on single delivery for the scope of this report, our aim will be to minimize cost $\boldsymbol{\theta}_{\text{mission}}$ incurred to take parcel from pickup point x_p to drop point x_d satisfying constraints put forth by vehicle V and $\mathbf{T}_{\text{wait}}$.

$$\begin{aligned}
&\min_{\alpha \in S} \boldsymbol{\theta}_{\text{mission}}(\alpha) \\
&\alpha \doteq \{x_p, x_d, V\} \\
&S \doteq S(V, T_{\text{wait}})
\end{aligned}$$

In next section we will see, how $\boldsymbol{\theta}_{\text{mission}}$ or more generally $\boldsymbol{\theta}_{\boldsymbol{a,b}}$ to move from point x_a to x_b, depends on vehicle selection.

3 Manned Ground Delivery: Establishing Baseline

To any research problem, it is important to define metric and establish baseline of the current approach in order to quantify performance of proposed approach. In this section, we will formalize existing meal delivery model and compute baseline cost.

In Manned ground delivery(MGD), package needs to be transported from point x_a (restaurant) to x_b (user). Here,

$$x_\alpha \doteq (x_\alpha, y_\alpha, z_\alpha) | x_\alpha \in \Omega_d \subset \mathbb{R}^3 \forall \alpha \in \{a, b\}$$

where Ω_d is reachable subspace by MGD, say appartments $\cup$ restaurants. Lets denote ground vehicle by $V_g \in \{\text{bike}, \text{car}, \text{scooter} \dots\}$ and $\Omega_g \subset \mathbb{R}^2$ be the subspace(roads) reachable by V_g. Let, $x_\alpha^g \in \Omega_g$ be the nearest point to $x_\alpha \in \Omega_d$, say parking spot. MGD takes a path:

$$p_{ab} \doteq x_a \longrightarrow x_a^g \longrightarrow x_b^g \longrightarrow x_b | p : t \in \mathbb{R}^+ \mapsto \Omega_d$$

Edges $x_\alpha \longleftrightarrow x_\alpha^g$ are traversed by human where stairs, elevators can also be used. Optimum path $p_{ab}^g \doteq x_a^g \longrightarrow x_b^g$ on Ω_g is found (usually third party service eg. google maps) and navigated to minimize time, since time taken primarily dictates cost as we will see later.

3.1 Modelling delivery cost

We will model monetary cost, θ_g of our MGD as this is a must condition to prove economic viability ie. $\theta_a \leqslant \theta_g$ where θ_a is monetary cost for Unmanned Aerial Delivery(UAD). Assuming there exists a feasible path, we model monetary cost $\theta_\alpha(x_a, x_b)$ as:

$$\begin{aligned}
\theta_\alpha(x_a, x_b) &= \theta^E + \theta^H + \theta^V \\
\theta^E &: \text{fuel cost} \\
&= \frac{\overbrace{d(p_{ab})}^{\text{path distance}}}{\underbrace{\eta(V_\alpha)}_{\text{mileage}}} \times \underbrace{c_{\text{fuel}}}_{\text{cost per vol.}} \\
\theta^H &: \text{Human resource cost} \\
&= \frac{\overbrace{c_{\text{human}}}^{\text{monthly salary}}}{30 \times \underbrace{t_d}_{\text{daily working hour}} \times 3600} \times \underbrace{t(p_{ab})}_{\text{path time}} \\
\theta^V &: \text{vehicle asset cost} \\
&= \frac{c_{\text{vehicle}}}{d_{\text{life}}} \times d(p_{ab})
\end{aligned}$$

Above model is very rough and inaccurate for eg. orders traffic varies through out day(peaks during lunch and dinner) but this is compensated by jointly using resource for e-commerce, vehicle is used as joint ownership model, software development cost is not considered. For more accurate model refer [16].But since we have estimate of avg. cost per delivery $\theta_g \sim$

₹ 65 from [16][17], We can use above model to estimate contribution of each component. Above model gives estimate [18] $\theta^H \sim$ ₹ 46.5, $\theta^E \sim$ ₹ 6.5, $\theta^V \sim$ ₹ 8 giving $\theta_g =$ ₹ 61.

As it can be seen, major contribution is θ^H ie. 75%, this should make it obvious that an Unmanned Ground vehicle(UGV) or more popularly self-driving cars presents itself as superior solution. But complete automation(no human intervention) on ground continues to be a challenging problem which requires detecting traffic signals, construction signs, people, moving vehicles and motion planning among them. Additionally UGV are time constrained during peak traffic as almost all human traffic is confined to Ω_g. Relatively, autonomy in aerial vehicles is less challenging due to absence of most of above and at the same time enjoys euclidean distance over manhattan distance at much higher speeds.

4 Unmanned Aerial Delivery

Similar to section 3, here we model UAD. Now instead, we have aerial vehicle $V_a \in \{\text{UAVs}\}$ and $\Omega_g \subset \mathbb{R}^3$ as aerial vehicles are not confined to roads. We will restrict Ω_g to outdoors. Formally, assume flat ground model with x-y plane parallel to ground and z axis pointing upwards. Let,

$$\begin{aligned} z^r(\mathbf{x}) &= \max(\mathbf{e_z}.\mathbf{x}_\alpha | \mathbf{x}_\alpha \in \overline{\Omega}_{\text{air}}, (\mathbf{x}_\alpha - \mathbf{x}) \times \mathbf{e_z} = \mathbf{0}) \\ \overline{\Omega}_{\text{air}} &: \text{subspace unoccupied by air} \\ &= \mathbb{R}^3 \setminus \Omega_{\text{air}} \\ \mathbf{e_z} &: \text{unit vector in z-axis} \end{aligned}$$

Informally, $z^r(\mathbf{x})$ gives roof height at point $\mathbf{x}$. Second condition restricts $\mathbf{x}_\alpha$ on line $\perp$ to x-y plane passing through $\mathbf{x}$. This lets us define

$$\begin{aligned} \Omega_a &= \Omega_{\text{outdoor}} \setminus \Omega_{\text{nofly}} \\ \Omega_{\text{outdoor}} &= \{\mathbf{x} | \mathbf{x}.\mathbf{e_z} > z^r(\mathbf{x})\} \\ \Omega_{\text{nofly}} &: \text{no fly zones} \end{aligned}$$

Also, let $\Omega_l \subset \Omega_a$ be surface where V_a can land. Our UAD model will thus take a path:

$$p_{ab} \doteq x_a \longrightarrow x_a^a \longrightarrow x_b^a \longrightarrow x_b | p : t \in \mathbb{R}^+ \mapsto \Omega_a$$

Note,parcel will be placed by restaurant on its rooftop $x_a^a \in \Omega_l$ from where our V_a can land and pickup. Similarly, parcel will be dropped at point x_b^a, nearest and reachable from x_b from where user can pickup.

We will try to find optimum path $x_a^a \longrightarrow x_b^a$ which minimizes energy use and avoids obstacles(other buildings). In order to find cost of a path, we will need first design the vehicle and model its energy consumption. We will restrict this exploration to VTOL, since we are constrained by relatively high min. take off angle as fence height to roof width ratio is high for urban rooftops. For design and analysis, we introduce reader to basic aeronautics concepts in next section relevant to our analysis.

5 Introduction to Aeronautics

Since, we assume that reader may not necessarily have Aeronautics/ Mechanical background, we will try to introduce to some basic concepts required to follow subsequent sections. For a detailed introduction, we recommend excellent texts[19], [20].

Let us look at the simplest case of level unaccelerated flight. Drawing similarity from ground body. TODO: insert figure In order to balance all forces, we need to oppose gravity and oppose aerodynamic drag similar to friction in ground bodies. Force opposing gravity ie. weight(F_g) is called lift(F_L) and drag(F_D) is compensated by generated thrust(F_T).

Now, let us take case of earliest flying body we have seen ie. paper planes. To study aerodynamic forces a paper palne, we can assume flat plate theory[21].

$$
\begin{aligned}
F_a &: \text{aerodynamic force} \\
&= \frac{dp}{dt} = \frac{dm}{dt} v_n \\
&= v_n \frac{\rho A dx}{dt} = v_n(\rho A v_\infty) \\
&= \rho A v_\infty^2 \sin(\theta) \\
F_L &= F_a \cos(\theta) = \rho A v_\infty^2 \sin(\theta)\cos(\theta) \qquad (3) \\
F_D &= F_a \sin(\theta) = \rho A v_\infty^2 \sin^2(\theta) \qquad (4)
\end{aligned}
$$

Here,

v_∞ : $airspeed$

v_n : speed normal to surface

p : air momentum

A : area of plate

θ : angle of attack

Note, even though this model makes too many oversimplifying assumptions(imcompressible, always seperated flow), it works surprisingly well to model[22] and perch flat plate glider on wire[23]. The glider experiences angle of attack as high as 90°- 120°which are well beyond in stall regime(angle at which flow becomes seperated), yet data fits nicely with additional gaussian radial basis function.

In practice though, we always use aerofoil shape over flat plate since aerofoil shape keeps flow attached(laminar) at higher angles and gives higher lift to drag ratio. Lift and drag for wing by lifting line theory is modelled as:

$$
\begin{aligned}
F_L &= q_\infty S C_l \\
F_D &= q_\infty S C_d \\
&\text{where,} \\
q_\infty &\doteq \frac{\rho v_\infty^2}{2} \\
C_l &: \text{lift coeffecient} \\
&\doteq C_l(\alpha, v_\infty, \mathrm{Re}) \\
C_d &: \text{drag coeffecient} \\
&= C_{d0} + \frac{C_l^2}{\pi e A R} \\
\alpha &: \text{angle of attack} \\
n_{\mathrm{Re}} &: \text{Reynolds number} \\
C_{d0} &: \text{parasitic drag} \\
e &: \text{oswald effeciency factor} \\
AR &: \text{aspect ratio of wing}
\end{aligned}
\tag{5}
$$

TODO: insert plot for cl , cd vs alpha

Drag forces, are always produced as byproduct of lift as can be seen from (3) and (4). This drag is called (lift)**induced drag**, while drag due to skin friction is called **parasitic drag**.Now that we know how to balance vetical forces ie. gravity F_g with lift F_L, lets see how to generate thrust F_T to oppose drag force F_D. Thrust is generated by propulsion sytems:

- Jet propulsion

- Rotory propeller

Immediately question aries, which one is more effecient? From Newtons's second law, rate of momentum change gives force T (eq. 3.1 [20]):

$$T = \dot{m}(v_e - v_\infty)$$
$$\dot{m} : \text{mass flow}$$
$$v_e : \text{exit speed of air}$$

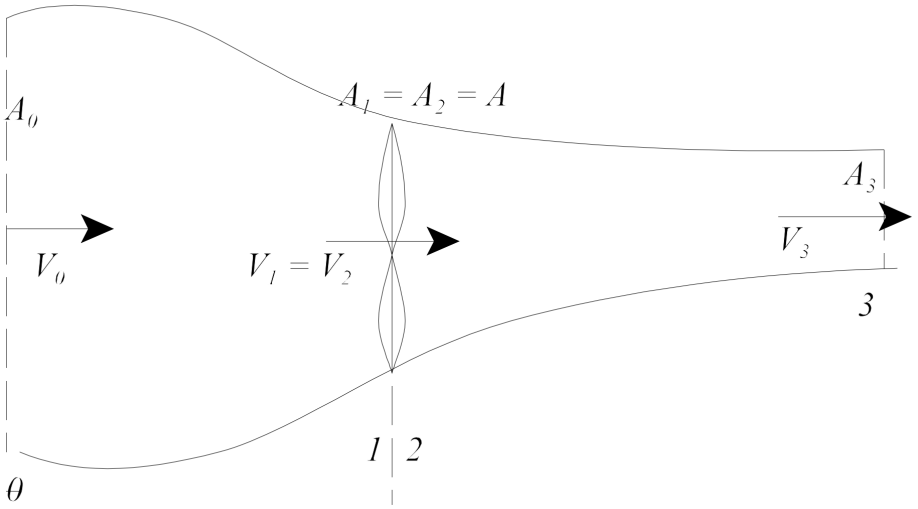
Since, kinetic energy imparted air, is wasted once exited. This gives an intution about effeciency that moving large mass to small speeds(rotory propeller) is more effecient over moving small mass to large speed. But there is a thrust effeciency tradeoff. Formally, effeciency(η_p) is given by(eq. 3.8 [20]).

$$\eta_p = \frac{2}{1 + v_e/v_\infty}$$

So back to our question, which to use? This depends on speed for which vehicle is designed to fly. At low speeds rotory propellers are more effecient but effeiency decreases with increasing airspeed while jet propulsion effeciency is constant w.r.t airspeed and is hence more effecient at high speed.

5.1 Propellers

We can think propeller to be nothing but just a rotating wing. Since wings generate lift at high airspeeds, this relative speed is generated by instead rotating propellers at high speed. We can model F_{prop} by integrating F_l and F_d of each section over blade length. This method is known as Blade Element Theory(BET). But for sake of simplicity, we will use momentum theory [24].



Applying Bernoulli's equation:

$$P_\infty + \frac{1}{2}\rho v_0^2 = P_1 + \frac{1}{2}\rho v_1^2 \tag{6}$$

$$P_\infty + \frac{1}{2}\rho v_3^2 = P_2 + \frac{1}{2}\rho v_2^2 \tag{7}$$

deducting (7) from (6) and note $v_1 = v_2$,

$$P_2 - P_1 = \frac{1}{2}\rho(v_3^2 - v_0^2) \tag{8}$$

Thrust by momentum change and pressure difference across propeller times area swept gives:

$$T = \rho A_3 v_3 (v_3 - v_0) \tag{9}$$

$$T = A(P_2 - P_1) \tag{10}$$

By mass conservation, $Av_1 = A3v_3$ in (9). equating (9) and (10) and together with (8) gives:

$$v_1 = \frac{v_0 + v_3}{2} \tag{11}$$

(11) states that speed through propeller is average of entry and exit. Overall momentum theory gives thrust(complete proof ref. to [24]):

$$T = P_i^{2/3}(2\rho A)^{1/3} \tag{12}$$

P_i : Induced power (engine power)

Now, that we have some rudimentary understanding of theory, we can begin with UAV design in next section.

6 UAV Design

In this section, we will design structure and components of our UAV which requires estimating wing sizing, power loading, energy consumption for mission, battery sizing and vechile weight. We refer to texts [25] and [20]. Introductory course[26] might also be helpful to reader.

We consider the delivery model where we partner with restaurants to place charging hubs on roofs. This model has advantage that total energy req. for mission reduces by saving VTOL routine for charging as this routine is most energy intensive. Battery will be recharged during order pickup time. Prime motivation to optimize for energy is that battery will contribute majorly to total weight as we will see later and power consumption propotional to W^3 (TODO: cite relevant section). A light weight UAV is more agile, safe and energy effecient.

For battery sizing, we will estimate ana upper bound for worst case mission scenario $z^r(x_{\text{pickup}})$, $z^r(x_{\text{drop}})$, $z^r(x_{\text{charge}})$ are all 0. Just next to each point x_{pickup}, x_{drop} and x_{charge} stands tallest building $z^r_{\max} \in \Omega_l$ which dictates the cruise height to $h_{\text{cruise}} = z^r_{\max}$. For compact notation, we will denote pickup, drop and next pickup/charge as simply a, b ,c. Further, all our analysis will lie in Ω_a and $x_\alpha \doteq x^a_\alpha \in \Omega_l \forall \alpha \in \{a, b, c\}$. Let,

$$
\begin{aligned}
E_{\text{mission}} &= E_{ac} = E_{ab} + E_{bc} \\
E_{ab} &= E_{\text{climb}}(h_{\text{cruise}} - h_a) \\
&\quad + E_{\text{cruise}}(d_{ab}) \\
&\quad + E_{\text{descent}}(h_{\text{cruise}} - h_b)
\end{aligned}
$$

since, $h_a = h_b = 0$

$$
\begin{aligned}
E_{\text{mission}} &= 2E_{\text{climb}}(h^{\max}) + 2E_{\text{descent}}(h^m ax) + E_{\text{cruise}}(d^{\max}_{ab} + d^{\max}_{bc}) \\
&< 4E_{\text{climb}}(h^{\max}) + E_{\text{cruise}}(d^{\max}_{ac})
\end{aligned}
$$

Note, for now, in above model we have not considered energy req. during transistion from vertical climb to cruise. This will need to be modeled in order to hold validity of upper bound.

6.1 Climb Energy

Modelling E_{climb},

$$\begin{aligned} F_T &= F_G + F_D \\ &= m_{\text{to}} g + q_\infty S_{\text{wet}} \end{aligned}$$

where,

$$q_\infty \doteq \frac{1}{2}\rho v_{\text{climb}}^2$$

S_{wet} : wetted surface area in top view

m_{to} : maximum take off mass

F_T is provided by all our propellers combined. Thrust by each propeller:

$$\begin{aligned} F_{\text{prop}} &= \frac{F_T}{N_{\text{prop}}} \\ P_i &= \sqrt{\frac{F_{\text{prop}}^3}{2\rho S_{\text{prop}}}} \\ P_{\text{shaft}} &= \frac{P_i}{\eta_{\text{prop}}} \\ P_{\text{climb}} &= \frac{P_{\text{shaft}}}{\eta_{\text{motor}}} \end{aligned}$$

Where,

N_{prop} : number of propellers

S_{prop} : propeller disk area

P_i : induced power from (12)

η_{prop} : propeller effeciency

η_{motor} : transmission effeciency

This finally gives climb energy:

$$E_{\text{climb}} = N_{\text{prop}} P_{\text{climb}} t_{\text{climb}}$$

Where, for constant climb rate v_{climb} :

$$t_{\text{climb}} = \frac{h^{max}}{v_{\text{climb}}}$$

We will limit our v_{climb} to 1 m/s, though commercial drones(Wingcopter) have $v_{\text{climb}} \leqslant 6m/s$, as we donot yet have understanding of heat dissipation.

Note that E_{climb} will have a minima at higher v_{climb}, satisfying constraint $P_{\text{climb}} < P^{\max}_{\text{discharge}}$. But operating at max discharge rate of battery reduces battery cycle life as irrevesible reactions happen at higher tempratures.

6.2 Cruise Energy

At level unaccelerated flight during cruise,

$$\begin{aligned} F_T &= F_D \\ F_L &= F_G \\ \text{From (5),} & \\ \frac{F_L}{F_D} &= \frac{C_l}{C_d} \\ \text{at cruise,} & \\ \pi_{\text{cruise}} &= (\frac{C_l}{C_d})_{\max} \\ &= \frac{1}{2}\sqrt{\frac{\pi e AR}{C_{D0}}} \\ F_T &= \frac{F_g}{\pi_{\text{cruise}}} \end{aligned} \tag{13}$$

For cruise energy, similar analysis can be carried out as carried out as in 6.1 by replacing F_T from (13). One thing, we will need is v_{cruise}. At cruise, induced drag equals parasitic drag(eq. 5.34, pg.297 [19])

$$\begin{aligned} C_{d0} &= KC_l^2 \\ \text{where,} & \\ K &= \frac{1}{\pi e AR} \end{aligned} \tag{14}$$

$$q_\infty c_l S_{\text{wing}} = F_G$$

$$v_{\text{cruise}} = \sqrt{\frac{2F_g}{\rho_\infty C_l S_{\text{wing}}}}$$

where from (14),

$$C_l = \sqrt{\frac{C_{d0}}{K}}$$

We will design our wing chord to get $v_{\text{cruise}} = 60$ km/h. We limit this speed, since this will be dictated by our obstacle avoidance system.

6.2.1 On estimate of glide ratio

An accurate estimate of glide ratio(π_{cruise})[27], will depend on final structure as parasitic drag (C_{d0}) depends on wetted surface area and surface material characeristics(coeffecient of friction). π_{cruise} can be as high as $\sim$ 70 for gliders, $\sim$ 60 for sail planes and as low as 4 for helicopters. For our estimates, we can assume $\pi_{\text{cruise}} = 11$ which is equivalent for cessna 172.

6.3 Transistion Energy

Too lengthy, refer to [28]

6.4 Maximum Takeoff Weight

$$m_{to} = m_{\text{battery}} + m_{\text{payload}} + m_{\text{empty}}$$
$$m_{\text{empty}} = m_{\text{structural}} + m_{\text{motors}} + m_{\text{avionics}}$$

Since, m_{empty} will depend on frame material chosen(Nylon, Carbon fibre) and load analysis. For this we will refer to software package SUAVE[29]. For now, we use emperical estimate of m_{empty} of VTOL of similar payload capacity(Wingcopter).

6.5 Propeller Sizing

We use emperical relation(eq. 8.71 [20])

$$D_{prop} = 10.7 P_{\text{shaft}}^{\frac{1}{4}}$$

subsectionBattery sizing

$$m_{\text{battery}} = \frac{E_{\text{mission}}}{\rho_E}$$

where,

ρ_E : energy density of battery

6.6 Analysis

Above numerical computations are carried out in [30]

m_{payload}	1.5 kg
m_{empty}	3.6 kg
ρ_E(Li-ion)	240 Wh/kg
ρ_E(Li-Ti)	96 Wh/kg
Wing length	1.8 m
Wing chord	0.39 m
Tail length	0.58 m
Tail chord	0.33 m
v_{climb}	1 m/s
h_{climb}	150 m
no. of vertical props	4
no. of horizontal props	1
Propeller Diameter	0.385 m
η_{prop}	0.8
η_{shaft}	0.8
c_{d0}	0.017
e_{oswald}	0.8
π_{cruise}	10
v_{cruise}	60 km/h
d_{ab}	6 km
d_{bc}	2km
$d_{cruise}(d_{ab} + d_{bc})$	8km
$P_{\text{avionics}}(2P_{\text{GPU}})$	15 W

Running line search to get optimum battery mass yeilds:

Battery type	m_{total}(kg)	m_{battery}(kg)
Li-ion	5.56	0.47
Li-Ti	6.6	1.5

Energy distribution in case of Li-ion being:

Component	Energy(kJ)
$\sum$ VTOL	373.5
Forward flight	18.6
Compute	16.2

It is interesting to note, how power hungry VTOL phase is! This is for battery sizing worst case climb height for all 4 phases at very low speed of 1 m/s. In actual operation this will reduce drastically.

We also calculate **worst case** mission energy cost:

Battery-Type	Cost(₹)
Li-ion	1.1
Li-Ti	1.5

7 Control and Motion Planning

Dynamics of a dynamical system can be given by firt order form:

$$\begin{aligned} &\dot{\mathbf{x}} = f(\mathbf{x}, \mathbf{u}) \\ &\text{where,} \\ &\quad \mathbf{x} : \text{state of system} \\ &\quad \mathbf{u} : \text{control input} \end{aligned} \tag{15}$$

We will try to answer 2 question:

Motion Planning or trajectory generation: Given set of way points and some constraints (obstacles, input saturations), Find a faesible trajectory for our system to follow. Formally (sec. 1.1.1 [31]), find state trajecotry $x : [t_a, t_b] \mapsto \mathbb{R}^n$ and corresponding input trajectory $u : [t_a, t_b] \mapsto \mathbb{R}^m$, such that our dynamical system (15) is transferred from initial state, $x(t_a) = \mathbf{x_a}$) to admissible final state $x(t_b) \in \Omega_f \subset \mathbb{R}^n$ and such that corresponding state traj $x(\circ)$ satisfies constraints $x(t) \in \Omega_x \subset \mathbb{R}^n, u(t) \in \Omega_u \subset \mathbb{R}^m \forall t \in [t_a, t_b]$. Sidenote: An optimal control problem would additionall minimize some cost $J(\mathbf{u})$.

Tracking controller: In and ideal world, our dynamical model would be completely accurate and there will be no noise/disturbances, we would execute our above input trajectory and reach admissible final state Ω_f. This is called **open loop control**. But world is not ideal, and in practice we have modelling uncertainites and disturbances. In order to track our trajectory, we have to constantly make corrective inputs by taking feedback of the current state. This is called **closed loop control**

Above 2 problems are studied extensively in field of control theory. Control of our vehicle can be thought as two seperate controllers, One for VTOL phase when forces and moments are primarily generated by diffrential thrust. During VTOL phase lifting surfaces have negligible control effects and can be ignored. While during forward flight, we shut down vertical facing propellers and all all forces, moments are generated by lifting surfaces(wings, elevons).

In this section, we study control of quadrotors and slightly more complex but interesting case of VTOL fixed wing tailsitters.

7.0.1 Quadrotors

Quadrotors are specific case of more general multi-rotor UAVs. Quadrotors have become primarily popular, as it has contains no complex or high stress moving parts unlike swash plates in helicopters. Additionaly conventional wisdom says, atleast 4 coplanar actuators are needed for good controllability. Since, our UAV has 4 top facing propellers, study of control of quadrotors is relevant. We primarily refer to [32] and [33].

7.0.2 Dynamics

Newtons equation of motion governing accelertaion of center of mass(COM) is:

$$m\ddot{\mathbf{x}}^{\mathcal{I}} = {}^{\mathcal{I}}R_{\mathcal{B}}\mathbf{f}_{\text{tot}}^{\mathcal{B}} + m\mathbf{g}^{\mathcal{I}}$$

where,

m : mass of quadrotor

$\mathcal{I}$: Intertial(world) frame

$\mathcal{B}$: Body frame

$\mathbf{x}^{\mathcal{I}}$: position vector of COM in frame $\mathcal{I}$

${}^{\mathcal{I}}R_{\mathcal{B}}$: Rotation matrix from $\mathcal{B}$ to $\mathcal{I}$

$\mathbf{f}_{\text{tot}}$: Thrust exerted (by propellers)

$$\mathbf{g}^{\mathcal{I}} : -\begin{pmatrix} 0 \\ 0 \\ g \end{pmatrix}^{\mathcal{I}}$$

We use notation, $\begin{pmatrix} p \\ q \\ r \end{pmatrix}^{\mathcal{F}}$ to denote vector in basis components $pe_x^{\mathcal{F}} + pe_y^{\mathcal{F}} + pe_z^{\mathcal{F}}$ of frame $\mathcal{F}$.

Further, euler equation of motion gives:

$$\tau^{\mathcal{B}}_{\text{tot}} = I\ddot{\theta}_{\mathcal{BI}} + \dot{\theta}_{\mathcal{BI}} \times I\dot{\theta}_{\mathcal{BI}}$$

I : inertia matrix in body frame

$\dot{\theta}_{\mathcal{BI}}$: angular velocity of $\mathcal{B}$ w.r.t $\mathcal{I}$

$\ddot{\theta}$: angular acc.

Force is generated by thrust producing deices ie. propellers driven by indivisually controlled motors. Force by single propeller is modelled as:

$$\mathbf{f}_{\text{prop}} = \kappa_f \omega_p^2 \mathbf{e}_n$$

where,

κ_f : force constant of propellers

ω_p : rpm of props

$\mathbf{e}_n$: unit vector normal to plane of rotation

Here, all the complexity is hidden by κ_f, which is a function of propeller diameter, blades airfoil section profile along span and angle of attack with rotating plane. In theory, κ_f will vary with advance ratio J and air speed v_∞ (sec. 9.2 [19]).

We will choose a fixed pitch propeller, optimized for chosen climb and cruise speed and ignore variation in effeciency due to air speed$v\infty$ as our flight speed envelope is small.

Finding κ_f may look like a complex task. But can be found quite simply by experimentally varying ω_p from 0 to ω_p^max and measuring Thrust respectively using load cells(weighing scale). This data is in turn fit as quadratic model to get κ_p.

Drag by each prop blade and thrust produced results to moments in body frame:

$$\tau^{\mathcal{B}}_{\text{prop}} = \text{sgn}(\omega_p)\kappa_m \mathbf{e}_z^{\mathcal{B}} + \mathbf{l}^{\mathcal{B}} \times \mathbf{f}^{\mathcal{B}}_{\text{prop}}$$

where,

sgn : ± 1 depending on direction of rotation

$\mathbf{l}$: position of prop w.r.t COM

Summing up indivisual forces and moments gives:

$$
\begin{aligned}
\mathbf{f}_{\text{tot}} &= \sum_{i=1}^{N=4} \mathbf{f}_{\text{prop},i} \\
\tau_{\text{tot}} &= \sum \tau_{\text{prop},i} \\
&= \kappa_m (\sum_{i=1}^{N=4} \text{sgn}(\omega_i)\omega_i^2)\mathbf{e_z} + \sum \mathbf{l}_{p,i} \times \kappa_f \omega_{p,i}^2 \mathbf{e}_z
\end{aligned}
$$

Commonly, props are chosen to rotate in opposite directions alternatively([34] [35] [33]), thus $\text{sgn}(\omega_{p,i})$ can be replaced by:

$$\text{sgn}(\omega_{p,i}) = (-1)^i$$

and $\mathbf{l}_{p,i}$ are placed such that they align with body axes $x^{\mathcal{B}}$ and $y^{\mathcal{B}}$ giving:

$$
\begin{bmatrix} \mathbf{l_1} & \mathbf{l_2} & \mathbf{l_3} & \mathbf{l_4} \end{bmatrix} = \begin{bmatrix} \begin{pmatrix} L \\ 0 \\ 0 \end{pmatrix} & \begin{pmatrix} 0 \\ L \\ 0 \end{pmatrix} & \begin{pmatrix} -L \\ 0 \\ 0 \end{pmatrix} & \begin{pmatrix} 0 \\ -L \\ 0 \end{pmatrix} \end{bmatrix}
$$

Having defined dynamics of quadrotor, we begin to introduce technique in non-linear control theory callled diffrential flatness in next section.

7.1 Diffrential Flatness

Roughly speaking, a system is diffrentially flat if, we can find a set of ouputs, equal to number of inputs s.t all states and inputs can be determined from these outputs without integration[36].

Formally, the system is diff. flat if we can find outputs $y \in \mathbb{R}^m$ of form:

$$y = y(x, u, \dot{u}, \dots u^{(p)})$$

such that,

$$
\begin{aligned}
x &= x(y, \dot{y}, \dots y^{(q)}) \\
u &= u(y, \dot{y}, \dots y^{(q)})
\end{aligned}
$$

7.2 Quadrotor: Diffrential flatness

Trajectory generation for quadrotor in presence of obstacles and tracking this trajectory becomes easy as quadrotor can be shown to be diff. flat [32] in outputs:

$$\mathbf{y} = \begin{pmatrix} \mathbf{x}_b^{\mathcal{I}} \\ \psi \end{pmatrix}$$
$$y : [t_a, t_b] \mapsto \mathbb{R}^3 \times SO(2)$$
$$\mathbf{x}_b^{\mathcal{I}} : \text{positiong of quadrotor in } \mathcal{I} \text{ frame}$$
$$\psi : \text{yaw angle}$$

7.2.1 Tracking controller

Tracking controller is simple PD controller in SO(3) to avoid singularities. PD control on flat outputs is more stable to parameter changes than on entire state [37].

$$\begin{aligned} \mathbf{e}_p &= \mathbf{x} - \mathbf{x}_T \\ \mathbf{e}_v &= \dot{\mathbf{x}} - \dot{\mathbf{x}}_T \\ \mathbf{f}_{\text{tot}} &= \mathbf{f}_T - K_p \mathbf{e}_p - K_v \mathbf{e}_v \end{aligned}$$
where,
$$K_\alpha : \text{gain matrices}$$

R_T is simply as function of flat outputs, which gives error in SO(3) [38]:

$$\begin{aligned} \mathbf{e}_R &= \frac{1}{2}(R_T^T R_{\text{est}} - R_{\text{est}}^T R_T) \\ \mathbf{e}_\omega &= \omega_{\mathcal{BW},\text{est}} - \omega_{\mathcal{BW},T} \\ \tau_{\text{tot}} &= -K_r \mathbf{e}_R - K_\omega \mathbf{e}_\omega \end{aligned}$$

$\begin{pmatrix} \|\mathbf{f}_{\text{tot}}\| \\ \tau_{\text{tot}} \end{pmatrix}$ will give indivisual inputs $\begin{pmatrix} u_1 \\ u_2 \\ u_3 \\ u_4 \end{pmatrix}$ by simply multiplying with inverse of a contant matrix A^{-1} [32].

7.2.2 Trajectory generation

For trajectory generation in presence of obstacles, refer to [32].

7.3 Aircraft: Diffrential flatness

This is of our interest as we want to control and generate trajectory our hybrid VTOl in forward flight too. Fixed wing Aircraft has been shown to be diff. flat. For Trajectory generation and controller design we refer to (Chapter 14: Automatic Flight Control Systems [39])

7.4 Tailsitter

TODO: add here. Refer [40]

8 Visual Perception

Perception is a genral term which refers to interpretation of sensory information in order to represent and understand environment.

We will first state our high level capabilities required for mission, which will inturn guide the scope of perception required.

1. User should be given abitlity to choose exact delivery location based on his reachability and convenience. The choice is restricted and subset of our reachable set(recall Ω_l from 4).
2. Digital Eelvation Map(buildings, trees, mountains) needs to be known in advance in order to plan trajectory, choosing landing sites.
3. Capability to percieve and avoid unseen obstacles(Parked cars, objects, antenas etc.)
4. Ability to accurately estimate ($\sim$ cm order) 3D position even in GPS denied environment.

1 requires providing uer with intutive map. We choose RGB-D map of service area, zoomed down to users GPS/current location. How do we compute this RGB-D map, will be discussed in later sections. 2 and 3 can be solved by mapping environment with sensors such as Lidar, Radar or camera sensors. We rule out Radar, Lidar due to prohibitive weight, high cost and non solid state nature.

Visual sensors(cameras), on the other hand have seen tremendous advancement in weight, resolution, power consumption and cost attributing to smartphone revolution. Cameras are general purpose sensor which, given the right intelligence and computation power both of which have been enabled by Deep models in Computer Vision community and low power GPUs (Jetson Family [41]). Commercial drones such as, Skydio 2 [42] have demonstrated robust obstacle avoidance and tracking to enable autonomous FPV video recording.

For 3, bio inspired **event cameras** seems to be promising solution [43] [44], since it measures intensity gradient rather than constant interval images, hence is not prone to motion blur. We have not explored this, but lies in our future work.

4 is usually solved as subproblem while solving mapping for 2 in visual simultaneous localization and mapping(vSLAM).

Specifically in coming sections, we will discuss about visual odometry(VO), visual localisation and mapping(vSALM), depth estimation and structure from motion(SfM) or 3D reconstruction.

All of the above emerge from common set of fundamentals, collectively studied under field of visual perception. For introduction to this field, we refer to excellent text [45] and course[46].

8.1 Modelling camera, image projection

Image projection is simply:

8.2 Camera calibration

Camera calibration matrix K, can be found by either spec sheet from manufacturer or general point correspondences(manually selected or automatic features). Implemetation in software packages(Matlab[47], opencv[48]) is available.

8.3 Feature extractors

All of subsequent sections will use feature points in order to find correspondences in different scenes. A feature point has location in image and description of neighbourhood patch. Desirable properties of feature points include scale, rotation and illumination invariant. For real time extraction, hand crafted ORB features are used while more robust SIFT features are used offline. Recently, learned local features are gaining popularity due to

higher recall and robustness. For comparision and benchmark of features extractors refer to [49] [50].

8.4 Visual localization and mapping

vSLAM consists of baiscally three modules:

1. Initialization
2. Tracking
3. Mapping

Initialization , initializes a coordiate frame for pose estimation and 3d reconstruction. First features are computed in current image frame and then correspondences are found with refrence image frame. Parallely Homography matrix H(for planar scenes), with 4 point correspondences(sec 5.3 [45]) and Essential matrix E (non-planar scene), is computed using eight point algorithm (sec. 5.2.1 [45]). Matching is done inside RANSAC scheme (sec 11.2.2 [45]) as matches can be noisy.

Tracking For continous tracking, constant velocity model can be assumed to guide where to look for next correspondences w.r.t current frame. While if tracking is lost(say dark spots, sudden gust) wider search is performed. Essential matrix E (rotation+translation) with epipolar constraint as already discussed gives relative pose.

Mapping 3D points can be recovered by triangulating pints from covisible frames. New points are added by adding current unmatched points. Since these can grow rapidly, only those points which are matched in future frames are kept and rest are discarded.

Loop Closure Since, our map will accumulate drift error according to distance of camera movement over time. This drift error can be eleminated if we again see previous scene in our path(loop). The correction is then propogated to all points in this loop path.

For informal survey on vSLAM methods, refer [51]. Popular implemetations include VINS-mono [52][53], ORB-SLAM [54] [55], OpenVslam[56][57].

8.5 Structure from motion and depth estimation

SfM is the process of reconstructing 3D structure from series of images from different view points. Similar techniques of point triangulation, bundle adjustment as discussed in 8.4. But at same time, SfM models assumes images from multiple uncalibrated images and additionaly texture mapping is also performed. SfM models are not designed to work in real time and are carried out offline with GPUs. refer [58], COLMAP[59]

Depth estimation ie. RGB-D map of entire scene is required in real time for obstacle avoidance in our case. This can either be done with parallax in multi view stereo(MVS) cameras or from single RGB image from monocular camera using Deep models. For training data generation [60] uses SfM pipeline to get sparse depth map of scene. Then training loss is defined only on these sparse regions. The model is then able to predict dense depth of entire scene. Deep models are also used by commercial drones[42] for robust obstacle avoidance.

9 Software Packages

In this section we list relevant software packages four our development.

9.1 ROS

9.2 Gazebo

9.3 PyBullet

9.4 Drake

9.5 Rotors

9.6 px4

9.7 SUAVE

9.8 OpenVSP

9.9 mapbox

References

[1] "Using drones for medical payload delivery in papua new guinea: Case study," link.

[2] C. K. Gilmore, M. Chaykowsky, and B. Thomas, "Autonomous unmanned aerial vehicles for blood delivery," 2019, link.

[3] J. W. Rosen, "Zipline's ambitious medical drone delivery in africa," 2017, link.

[4] "Blood transfusion services: Handling, storage and returns," link.

[5] "How rwanda built a drone delivery service - youtube," https://www.youtube.com/watch?v=jEbRVNxL44c, (Accessed on 02/05/2020).

[6] "Wingcopter vaccine delivery in vanuatu - youtube," https://www.youtube.com/watch?v=T1hVFivopD8, (Accessed on 02/05/2020).

[7] "Deliver future: Parcelcopter 4.0 — delivering vital medicines by drone - youtube," https://www.youtube.com/watch?v=id00S4L0P5A, (Accessed on 02/05/2020).

[8] N. R. News, "Restaurant takeout and delivery are taking a bite out of dine-in traffic," link.

[9] "Swiggy: Zomato, swiggy, ubereats reduce discounts as food delivery market grows cold," https://economictimes.indiatimes.com/small-biz/startups/newsbuzz/zomato-swiggy-ubereats-reduce-discounts-as-food-delivery-market-grows-cold/articleshow/72286430.cms, Nov 2019, (Accessed on 02/05/2020).

[10] "Online food delivery - worldwide — statista market forecast," https://www.statista.com/outlook/374/100/online-food-delivery/worldwide, (Accessed on 02/05/2020).

[11] "Domino's nutritional chart per serving," https://www.bd.com/resource.aspx?IDX=4261, (Accessed on 02/05/2020).

[12] "Swiggy expands restaurants' reach to remote locations - times of india," https://timesofindia.indiatimes.com/companies/swiggy-expands-restaurants-reach-to-remote-locations/articleshow/61647301.cms, (Accessed on 02/05/2020).

[13] "Wing launches america's first commercial drone delivery service to homes in christiansburg, virginia - youtube," https://www.youtube.com/watch?v=wCTKwkYzVzo, (Accessed on 02/05/2020).

[14] "Prime air," https://www.aboutamazon.co.uk/innovation/prime-air, (Accessed on 02/05/2020).

[15] "Uber air: Delivering uber eats with drones — uber elevate — uber - youtube," https://www.youtube.com/watch?v=0yMv16p8FO8, (Accessed on 02/05/2020).

[16] D. goyal, "Our unit economics for food delivery in india," https://www.zomato.com/blog/our-unit-economics-for-food-delivery-in-india, June 2016, (Accessed on 02/06/2020).

[17] "Zomato currently makes losses of rs. 25 on each order on delivery costs alone - officechai," https://officechai.com/startups/zomato-currently-makes-losses-rs-25-order-delivery-costs-alone/, April 2019, (Accessed on 02/06/2020).

[18] "Delivery cost estimate - colab," https://colab.research.google.com/drive/1P5YvU2cM2_rJBUAb0sPsPvH70FoY293b.

[19] J. Anderson, *Introduction to Flight*, ser. McGraw-Hill series in aeronautical and aerospace engineering. McGraw-Hill Higher Education, 2005. [Online]. Available: https://books.google.co.in/books?id=pW8U-UddgDMC

[20] ——, *Aircraft Performance and Design*, ser. McGraw-Hill aerospace science /technology series. McGraw-Hill, 1999. [Online]. Available: https://books.google.co.in/books?id=ct9kQgAACAAJ

[21] J. Tangler and D. Kocurek, "Wind turbine post-stall airfoil performance characteristics guidelines for blade-element momentum methods," in *43rd AIAA Aerospace Sciences Meeting and Exhibit*, 2005, p. 591.

[22] R. Cory and R. Tedrake, "Experiments in fixed-wing uav perching," in *AIAA Guidance, Navigation and Control Conference and Exhibit*, 2008, p. 7256.

[23] J. Moore, R. Cory, and R. Tedrake, "Robust post-stall perching with a simple fixed-wing glider using lqr-trees," *Bioinspiration & biomimetics*, vol. 9, no. 2, p. 025013, 2014.

[24] F. H. Lutze, "Lecture aoe 3104 vehicle performance," http://www.dept.aoe.vt.edu/~lutze/AOE3104/thrustmodels.pdf, 2014.

[25] R. Beard and T. McLain, *Small Unmanned Aircraft: Theory and Practice*. Princeton University Press, 2012. [Online]. Available: https://books.google.co.in/books?id=YqQtjhPUaNEC

[26] S. Sardela, "Design of fixed wing aerial vehicles - nptel," 2018. [Online]. Available: https://nptel.ac.in/courses/101/104/101104073/

[27] "Lift-to-drag ratio - wikipedia," https://en.wikipedia.org/wiki/Lift-to-drag_ratio, (Accessed on 02/08/2020).

[28] A. Kamal and A. Ramirez-Serrano, "Design methodology for hybrid (vtol+ fixed wing) unmanned aerial vehicles," *Aeronaut. Aerosp. Open Access J*, vol. 2, pp. 165–176, 2018.

[29] T. W. Lukaczyk, A. D. Wendorff, M. Colonno, T. D. Economon, J. J. Alonso, T. H. Orra, and C. Ilario, "Suave: an open-source environment for multi-fidelity conceptual vehicle design," in *16th AIAA/ISSMO Multidisciplinary Analysis and Optimization Conference*, 2015, p. 3087.

[30] A. Jain, "Vtol_design.ipynb - google colab," https://colab.research.google.com/drive/1d29BxWIPj-Z5CBLk3A9gKSGaA93J-tg-, 1 2020, (Accessed on 02/08/2020).

[31] H. Geering, *Optimal Control with Engineering Applications*. Springer Berlin Heidelberg, 2007. [Online]. Available: https://books.google.co.in/books?id=oLEk009WAEEC

[32] D. Mellinger and V. Kumar, "Minimum snap trajectory generation and control for quadrotors," in *2011 IEEE International Conference on Robotics and Automation*. IEEE, 2011, pp. 2520–2525.

[33] T. Lee, M. Leok, and N. H. McClamroch, "Geometric tracking control of a quadrotor uav on se (3)," in *49th IEEE conference on decision and control (CDC)*. IEEE, 2010, pp. 5420–5425.

[34] A. Tayebi and S. McGilvray, "Attitude stabilization of a vtol quadrotor aircraft," *IEEE Transactions on control systems technology*, vol. 14, no. 3, pp. 562–571, 2006.

[35] P. Castillo, R. Lozano, and A. Dzul, "Stabilization of a mini-rotorcraft having four rotors," in *2004 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)(IEEE Cat. No. 04CH37566)*, vol. 3. IEEE, 2004, pp. 2693–2698.

[36] R. M. Murray, M. Rathinam, and W. Sluis, "Differential flatness of mechanical control systems: A catalog of prototype systems," in *ASME international mechanical engineering congress and exposition*. Citeseer, 1995.

[37] J. Levine, "An introduction to differentially flat systems — jean levine - youtube," https://www.youtube.com/watch?v=KYdZQBVxYf0, May 2016, (Accessed on 02/08/2020).

[38] F. Bullo and A. Lewis, *Geometric Control of Mechanical Systems: Modeling, Analysis, and Design for Simple Mechanical Control Systems*, ser. Texts in Applied Mathematics. Springer New York, 2004. [Online]. Available: https://books.google.co.in/books?id=jg7VumxOoe4C

[39] J. Levine, *Analysis and Control of Nonlinear Systems: A Flatness-based Approach*, ser. Mathematical Engineering. Springer Berlin Heidelberg, 2009. [Online]. Available: https://books.google.co.in/books?id=HHRslb_8970C

[40] R. Ritz and R. D'Andrea, "A global controller for flying wing tailsitter vehicles," in *2017 IEEE international conference on robotics and automation (ICRA)*. IEEE, 2017, pp. 2731–2738.

[41] "High performance ai at the edge — nvidia jetson tx2," https://www.nvidia.com/en-us/autonomous-machines/embedded-systems/jetson-tx2/, (Accessed on 02/08/2020).

[42] "Skydio 2: The drone you've been waiting for. – skydio, inc." https://www.skydio.com/, (Accessed on 02/08/2020).

[43] "Rapid, dynamic obstacle avoidance with an event-based camera - youtube," https://www.youtube.com/watch?v=sbJAi6SXOQw&t=20s, (Accessed on 02/09/2020).

[44] D. Falanga, S. Kim, and D. Scaramuzza, "How fast is too fast? the role of perception latency in high-speed sense and avoid," *IEEE Robotics and Automation Letters*, vol. 4, no. 2, pp. 1884–1891, 2019.

[45] Y. Ma, S. Soatto, J. Kosecká, and S. Sastry, *An Invitation to 3-D Vision: From Images to Geometric Models*, ser. Interdisciplinary Applied Mathematics. Springer New York, 2012. [Online]. Available: https://books.google.co.in/books?id=FwfSBwAAQBAJ

[46] K. D. Jianbo Shi, "Robotics: Perception - university of pennsylvania — coursera," https://www.coursera.org/learn/robotics-perception/home/info, 2018, (Accessed on 02/09/2020).

[47] "Single camera calibrator app - matlab & simulink," https://www.mathworks.com/help/vision/ug/single-camera-calibrator-app.html, (Accessed on 02/09/2020).

[48] "Opencv: Camera calibration with opencv," https://docs.opencv.org/3.4.9/d4/d94/tutorial_camera_calibration.html, (Accessed on 02/09/2020).

[49] V. Balntas, K. Lenc, A. Vedaldi, and K. Mikolajczyk, "Hpatches: A benchmark and evaluation of handcrafted and learned local descriptors," in *CVPR*, 2017.

[50] J. L. Schönberger, H. Hardmeier, T. Sattler, and M. Pollefeys, "Comparative Evaluation of Hand-Crafted and Learned Local Features," in *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[51] T. Taketomi, H. Uchiyama, and S. Ikeda, "Visual slam algorithms: a survey from 2010 to 2016," *IPSJ Transactions on Computer Vision and Applications*, vol. 9, no. 1, p. 16, 2017. [Online]. Available: https://doi.org/10.1186/s41074-017-0027-2

[52] "Vins-mono: A robust and versatile monocular visual-inertial state estimator," https://github.com/HKUST-Aerial-Robotics/VINS-Mono, (Accessed on 02/09/2020).

[53] T. Qin, P. Li, and S. Shen, "Vins-mono: A robust and versatile monocular visual-inertial state estimator," *IEEE Transactions on Robotics*, vol. 34, no. 4, pp. 1004–1020, 2018.

[54] "Orb_slam2: Real-time slam for monocular, stereo and rgb-d cameras, with loop detection and relocalization capabilities," https://github.com/raulmur/ORB_SLAM2, (Accessed on 02/09/2020).

[55] R. Mur-Artal, J. M. M. Montiel, and J. D. Tardos, "Orb-slam: a versatile and accurate monocular slam system," *IEEE transactions on robotics*, vol. 31, no. 5, pp. 1147–1163, 2015.

[56] "xdspacelab/openvslam: Openvslam: A versatile visual slam framework," https://github.com/xdspacelab/openvslam, (Accessed on 02/09/2020).

[57] S. Sumikura, M. Shibuya, and K. Sakurada, "OpenVSLAM: A Versatile Visual SLAM Framework," in *Proceedings of the 27th ACM International Conference on Multimedia*, ser. MM '19. New York, NY, USA: ACM, 2019, pp. 2292–2295. [Online]. Available: http://doi.acm.org/10.1145/3343031.3350539

[58] J. L. Schönberger and J.-M. Frahm, "Structure-from-motion revisited," in *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016. [Online]. Available: https://www.cv-foundation.org/openaccess/content_cvpr_2016/papers/Schonberger_Structure-From-Motion_Revisited_CVPR_2016_paper.pdf

[59] "colmap/colmap: Colmap - structure-from-motion and multi-view stereo," https://github.com/colmap/colmap, (Accessed on 02/09/2020).

[60] Z. Li, T. Dekel, F. Cole, R. Tucker, N. Snavely, C. Liu, and W. T. Freeman, "Learning the depths of moving people by watching frozen people," *CoRR*, vol. abs/1904.11111, 2019. [Online]. Available: http://arxiv.org/abs/1904.11111