Control of Hover for a Micro-Air-Vehicle using a Visual Snapshot

Matthew A. Garratt, Andrew J. Lambert, Hamid Teimoori
University of New South Wales, Australia
m.garratt,a.lambert,h.teimoori@adfa.edu.au

Abstract

The problem of developing a reliable system for sensing and controlling the hover of a Micro Air Vehicle (MAV) using visual snapshots is considered. A new algorithm is proposed that uses a stored image of the ground, a snapshot taken of the ground directly under the MAV, as a visual anchor point. The absolute translation of the aircraft and its velocity are then calculated by comparing the subsequent frames with the stored image and fed into the position controller. For controller design and testing purposes, we analytically derive a complete model of a small size helicopter with no stabilizing bar (flybar). The simulation results for 2D and 3D versions of the snapshot algorithm confirm the effectiveness of the proposed algorithm.

1 Introduction

The objective of this work is to develop a reliable system for sensing and controlling the hover of a Micro Air Vehicle (MAV). The work will be suited to both rotary wing and flapping wing implementations. The system would require only visual and inertial sensing, breaking the reliance on technologies such as GPS which can be selectively disabled or jammed and which may be unavailable indoors and in cluttered environments. Range measuring technologies are required for control of height in hover but state-of-the-art options for sensing height such as laser range-finders or radar are simply impractical for MAVs owing to the physical barriers to miniaturisation that exist. Additional advantages of visual guidance are that it is passive, small and low cost.

2 Background

Use of vision for controlling the hover of a rotorcraft has been demonstrated by a number of researchers. One of the earliest demonstrations [O.Amidi *et al.*, 1999] used a combination of image feature tracking and stereo vision to control the hover of a 67kg Yamaha R-50 helicopter. However the reliance on stereo vision to find the range to features makes the miniaturisation of the equipment for use on a MAV difficult as stereo vision requires a minimum disparity between cameras.

Structured environments have been used successfully for hover control (e.g. [L.Mejias *et al.*, 2006; C.S.Sharp *et al.*, 2001; O.Shakernia *et al.*, 2002]. For example, an autonomous helicopter was landed on a helipad using vision and inertial information in [S.Saripalli *et al.*, 2003] but the system relied on a pattern of polygons painted on the helipad so that the helicopter could align itself with the pattern. In this work, we target natural landscapes rather than artificially structured ones, so these techniques are not suitable.

Determination of altitude by visual means for height control in a hovering rotorcraft is also a challenge. In [Cherian *et al.*, 2009], altitude estimation is attempted by analysing the texture in a single downwards looking camera, however this technique was found to be impractical in environments of low texture and only works at low-altitudes. Given ground speed from a non-visual sensor such as GPS, optic flow can be used to estimate range. This has been used for the problem of terrain following [Garratt and Chahl, 2008], centering in a natural canyon [Griffiths *et al.*, 2006] and obstacle avoidance [Zingg *et al.*, 2010; Beyeler *et al.*, 2007]. However for control of hover, optic flow cannot be used to control height as it does not produce range unless there is significant motion which is counter to the objective of a near perfect hover.

Optic flow or image motion can been used to estimate velocity given a measure of range from another sensor such as stereo vision or a laser rangefinder [Garratt and Chahl, 2007; Corke, 2004], however, optic flow only provides a velocity damping effect and horizontal drift will occur. There is some evidence that bees make use of a stored 2D snapshot to locate themselves [Cartwright and Collet, 1992]. Cartwright and Collett [Cartwright and Collet, 1983] suggest that the direction in which the bees move at any moment is governed by the discrepancy between the snapshot and its current retinal image. A new algorithm is therefore proposed that uses a stored image of the ground, a snapshot, so to speak, taken of the ground directly under the MAV. This is different to previous approaches involving optic flow, in that, as well as damping the motion using feedback from optic flow, we propose feedback of absolute displacement measured from comparison to a single datum image taken at our reference location. By comparing subsequent frames with this snapshot, it should be possible to calculate absolute translation from the datum. A 2D version of the snapshot algorithm would need to make use of a secondary means of determining height above the ground, such as stereo vision. A 3D version of the snapshot hover should be possible based on calculating the expansion and contraction of the image as the MAV climbed and descended.

One of the limitations of a snapshot hover, would be that a measure of scale is still required to bootstrap the algorithm. With a 2D hover, the height is assumed to be known by some other means. With a 3D snapshot algorithm, this will not be the case. A number of suggestions for bootstrapping are proposed. Firstly, for hover after a vertical take off, it should be possible to integrate the vertical velocity to obtain an initial height estimate to start the snapshot working. The vertical velocity may be estimated from the inertial sensors, optic loom and from the open loop vertical dynamics of the MAV. A transition to hover from forward flight may also provide the means to obtain an initial estimate of scale using the height known from optic flow during the transition to hover. For a practical implementation, sufficient overlap between the snapshot and the current image is necessary. This means that the MAV could not stray more than about 20 percent of the MAV height away from where the snapshot was taken.

In [Blosch *et al.*, 2010] and [Ahrens *et al.*, 2009], visual based SLAM algorithms for a quadrotor with a single camera are demonstrated which give impressive results for hover and exploratory flight. However, these techniques are computationally expensive and currently rely on external processing to support the computational load. In our case, we do not use feature tracking, but rather we use the Image Interpolation Algorithm (I^2A) [Srinivasan, 1994]. We have demonstrated (I^2A) working on a 1024×768 pixel image to calculate average translatory flow and image loom at 50 frames per second using an FPGA [Garratt *et al.*, 2009]. The FPGA solution allows for a single chip processing unit with associated low-weight, making it highly suitable for integration into a MAV. Our preliminary design efforts confirm that the same FPGA could be used to simultaneously calculate optic flow, optic loom, and snapshot displacement from a stored image.

3 MAV Simulation and Control

This section describes our simulation test-bed which is used to examine the feasibility of using visual snapshot to control the position of a small helicopter. A closed loop micro helicopter simulation has been developed to test this, which is based on a radio-controlled "Revolutor H-610" helicopter [Keyence Corporation, 2011]. The H-610 helicopter has a small body, weighing only $0.1kg$ and measuring $0.29m$ in length. A two blade main rotor produces the thrust force and a small tail rotor is used to counteract the main rotor torque and provide yaw control. A hinge-less hub is used to connect the blades of the main rotor to their pivot. It does not have a stabiliser bar (fly-bar), which is one of the main reasons for choosing the H-610 as our subject, since the delayed response caused by the fly-bar makes precise control of the helicopter more difficult as the bandwidth of the plant is reduced.

Since the helicopter model is too small to have a structure to change collective pitch and control the thrust force, it uses the main rotor angular velocity for controlling the thrust and changing the altitude. Cyclic variation of blade pitch is required to make the main rotor thrust vector tilt so that control of pitching, rolling and horizontal force can be achieved. Cyclic pitch is achieved by making rapid and once per revolution variations to the driving torque applied to the main rotor. The system uses the combination of an encoder disk that rotates with the shaft and a matching photo sensor to detect the azimuth angle of the rotor blades (i.e. the rotation angle measured clockwise relative to the negative longitudinal axis of the helicopter). In order to change the blade pitch, the motor power output is increased/decreased when the rotor blades come to a certain position as viewed from the body using the signal provided by the photo sensor. This change in torque is transmitted to the rotor blade through a connecting rod which is offset from the center of rotation. As the torque changes, the connection rod deforms and imparts a change in pitch in the rotor blade by twisting the blade along its length. Using this technique, cyclic pitch control is applied to the helicopter [Keyence Corporation, 2011].

Table 1 summarizes some of the main physical parameters for the Revolutor H-610 used in our simulations. For this helicopter, the body mass, fuselage dimensions and the main and tail rotors diameters are the only available parameters. Hence, other necessary parameters such as the moments of inertia: I_{xx}, I_{yy}, I_b are approximated from basic engineering principles using the published specifications and the available geometry shown in Fig. 1.

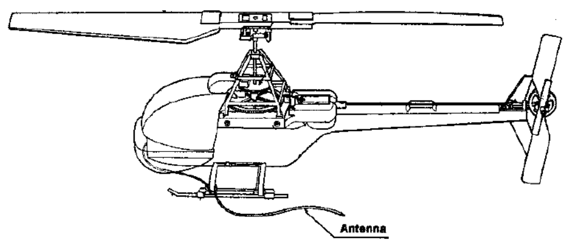

Figure 1: Revolutor H-610 helicopter model

Table 1: Helicopter physical parameters

Parameters	Description
$m = 0.1\ kg$	helicopter mass
$I_{xx} = 1.62 \times 10^{-5}\ kg.\ m^2$	rolling moment of inertia
$I_{yy} = 8 \times 10^{-5}\ kg.\ m^2$	pitching moment of inertia
$I_b = 1.86 \times 10^{-5}\ kg.\ m^2$	blade moment of inertia
$b = 2$	number of blades
$R = 0.175\ m$	main rotor radius
$c = 0.021\ m$	main rotor blade chord
$\alpha = 5\ rad^{-1}$	main rotor lift curve slope
$h_{mr} = 0.068\ m$	vertical location of main rotor

3.1 Helicopter Model

The simulation of the H-610 couples full 6DOF rigid body dynamics equation with a first order model of the main rotor flapping dynamics. The simulation is based on a similar model for an electrically propelled 8kg helicopter at [Garratt, 2007] with modifications to account for the different pitch control scheme and the lack of a flybar. The model includes nonlinear rigid body dynamics, main rotor and tail rotor flapping, controller and servo dynamics. A simplified block diagram of the system is shown in Fig. 2. In this model (x, y, z) denotes the position of the helicopter center of gravity, (p, q, r) and (u, v, w) are the helicopter angular and linear velocities in body frame, respectively. Helicopter attitude is expressed by the Euler angles (ϕ, θ, ψ) for roll, pitch and yaw respectively. In the following section, we provide a brief description of the simulation model.

3.2 Rigid-body equations

The helicopter dynamics is described by conventional six degrees of freedom rigid body equation of motion, in which the forces and moments that are generated by the rotor and other aerodynamically generated forces are considered as external forces/moments and expressed relative to the body center of gravity. Due to the limit of space, the representation of rigid body equations of motion is omitted here, see e.g. [Garratt, 2007].

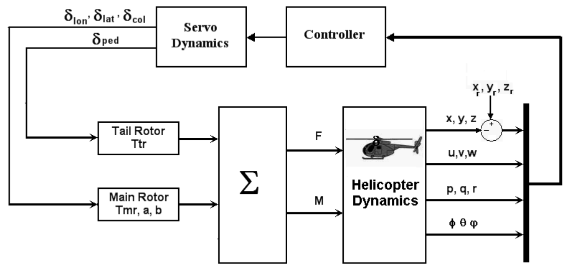

Figure 2: Block diagram of the controlled helicopter dynamics.

3.3 Main and Tail Rotors Forces and Moments

For rotor flapping representation, we consider the tip-path-plane (TPP) model, which is the path described by the helicopter blade tips during their rotation. In this representation, the lateral and longitudinal flapping dynamics can be expressed by first order equations:

$$\begin{aligned} \dot{a} &= -q - \frac{a}{\tau_{mr}} + \frac{1}{\tau_{mr}}(A_u u + A_{lon}\delta_{lon}) \\ \dot{b} &= -p - \frac{b}{\tau_{mr}} + \frac{1}{\tau_{mr}}(B_v v + B_{lat}\delta_{lat}), \end{aligned} \tag{1}$$

where a and b are the rotor flapping in the longitudinal and lateral directions, respectively. The parameter τ_{mr} is the main rotor flapping time-constant, δ_{lon} and δ_{lat} are the lateral and longitudinal cyclic control inputs (pilot stick or control system outputs) and A_{lon} and B_{lat} are effective steady-state lateral and longitudinal gains from the cyclic inputs to the main rotor flapping angles. In (1), $A_u = \frac{\partial a}{\partial u}$ and $B_v = \frac{\partial b}{\partial v}$ are constants and represent the lateral and longitudinal dihedral derivatives. The dihedral effect is the change of TPP tilt due to lateral and longitudinal velocities (this makes the helicopter roll away from sideslip) [Padfield, 2007]. In our study, due to their importance, these two parameters are augmented to the original equations of flapping dynamics presented in [Mettler, 2003; Garratt, 2007]. Since the rotor is symmetric, we consider $A_u = -B_v$.

In order to include the effects of rotor flapping on the fuselage rigid body dynamics, we express the forces and moments produced by the rotor in terms of the rotor flapping. The forces and moments generated by the main and tail rotors, in complete form, are represented by [Prouty, 1990]:

$$\begin{bmatrix} X \\ Y \\ Z \end{bmatrix} = \begin{bmatrix} -T_{mr}a \\ T_{mr}b + T_{tr} \\ -T_{mr} \end{bmatrix} \tag{2}$$

$$\begin{bmatrix} L \\ M \\ N \end{bmatrix} = \begin{bmatrix} Y.h_{mr} + T_{tr}.h_{tr} \\ X.h_{mr} \\ X.y_{mr} + Y.l_{mr} + T_t.l_{tr} + Q_{mr} \end{bmatrix}, \tag{3}$$

where, (X, Y, Z, L, M, N) are the external forces and moments and the torque Q_{mr} is the aerodynamic drag due to the rotation of the main rotor. In (3), Q_{mr} is approximated as the sum of the induced torque due to generated thrust, and the torque due to the profile drag on the blades [Padfield, 2007].

3.4 Servo Actuator Dynamics

Servo dynamics set constraints in designing controllers due to delay and rate limitation. The servo model considered is a simple first order transfer function with a pure time delay T_d, expressed as:

$$G_s = \frac{e^{-T_d s}}{\tau_a s + 1}, \tag{4}$$

where τ_a is the actuator time constant representing the speed limits and the bandwidth of the actuator model. In our simulation, the overall pure time delay, including the servo, computational and sensor delays, is considered as $T_d = 40ms$ in the control loop, which is simulated as transport delay in Simulink model. Furthermore, the servo time constant is set to $\tau_a = 0.029s$.

3.5 Controller Design

The importance of a reliable and effective control strategy is the same as an accurate positioning estimation system. Helicopters are highly nonlinear and have open-loop unstable dynamics as well as significant cross-coupling between their control channels which makes their control a challenging task. Despite this, the classical PID controllers can be quite effective for stable hovering or low-speed flight conditions [Mettler *et al.*, 2000; Castillo *et al.*, 2005; Amidi, 1996].

In order to stabilize and control the helicopter system, high bandwidth low latency attitude data, provided by inertial measurements, are used in attitude control loop, which is also known as inner-loop. The high bandwidth attitude information will improve the controller performance against high latency, unstable and highly coupled properties of helicopter. The desired helicopter attitude signals are assigned by the position control loop (outer-loop), which uses the helicopter position and velocity information from the vision system as the feedback and generates the necessary roll and pitch angles in order for the helicopter to follow a trajectory or control its position. To design the controllers, the helicopter dynamics is linearized, using the small perturbation theory, by considering the aircraft dynamical motion as small deviations about a steady equilibrium flight condition (trim condition). In trim condition none of the state variables change with time and there is no translational or rotational acceleration. We assume the perturbations around the equilibrium point are small, therefore, the simplified results are not applicable to large amplitude motions such as spinning or where highly nonlinear effects are involved such as stall.

3.6 The State-space Model of the Helicopter

Having derived the linearized six degrees of freedom rigid body equations of motion, and considering TPP equations of motion (1) and the forces and moments produced by the main and tail rotors ((2) and (3)), the simplified parameterized state space model for decoupled lateral and longitudinal dynamics are obtained

$$\begin{bmatrix} \dot{u} \\ \dot{q} \\ \dot{\theta} \\ \dot{a} \end{bmatrix} = \begin{bmatrix} X_u & X_q & -g & X_a \\ M_u & M_q & 0 & M_a \\ 0 & 1 & 0 & 0 \\ \frac{A_u}{\tau_{mr}} & -1 & 0 & \frac{-1}{\tau_{mr}} \end{bmatrix} \begin{bmatrix} u \\ q \\ \theta \\ a \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ 0 \\ \frac{A_{lon}}{\tau_{mr}} \end{bmatrix} \delta_{lon} \tag{5}$$

$$\begin{bmatrix} \dot{v} \\ \dot{p} \\ \dot{\phi} \\ \dot{b} \end{bmatrix} = \begin{bmatrix} Y_v & Y_p & g & Y_b \\ L_v & L_p & 0 & L_b \\ 0 & 1 & 0 & 0 \\ \frac{B_v}{\tau_{mr}} & -1 & 0 & \frac{-1}{\tau_{mr}} \end{bmatrix} \begin{bmatrix} v \\ p \\ \varphi \\ b \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ 0 \\ \frac{B_{lat}}{\tau_{mr}} \end{bmatrix} \delta_{lat}, \tag{6}$$

where, $X_u = \frac{1}{m}\frac{\partial X}{\partial u}$, $M_u = \frac{1}{I_{yy}}\frac{\partial M}{\partial u}, \ldots$ are the force and moment derivatives normalized by the mass of the aircraft or the respective moment of inertia. Using (2) and (3) and the helicopter parameters given in Table 1, the numerical representation of longitudinal and lateral dynamics (5) and (6) is derived as

$$\begin{bmatrix} \dot{u} \\ \dot{q} \\ \dot{\theta} \\ \dot{a} \end{bmatrix} = \begin{bmatrix} 0 & 0 & -9.81 & -9.81 \\ 0 & 0 & 0 & 834 \\ 0 & 1 & 0 & 0 \\ 0.468 & -1 & 0 & -122.4 \end{bmatrix} \begin{bmatrix} u \\ q \\ \theta \\ a \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ 0 \\ 122.4 \end{bmatrix} \delta_{lon} \tag{7}$$

$$\begin{bmatrix} \dot{v} \\ \dot{p} \\ \dot{\phi} \\ \dot{b} \end{bmatrix} = \begin{bmatrix} 0 & 0 & 9.81 & 9.81 \\ 0 & 0 & 0 & 4117 \\ 0 & 1 & 0 & 0 \\ -0.468 & -1 & 0 & -122.4 \end{bmatrix} \begin{bmatrix} v \\ p \\ \varphi \\ b \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ 0 \\ 122.4 \end{bmatrix} \delta_{lat}, \tag{8}$$

which contain unstable complex poles. We use the classical PID controllers to stabilize the helicopter dynamics in lateral and longitudinal channels. The PID controllers' gains are tuned automatically to minimize the Integral of Time Absolute Error (ITAE) performance index, which is mathematically given by $ITAE = \int_0^\infty t|e|\,dt$ where, t is the time and e is the error which is calculated as the difference between the desired and measured outputs when the system is excited by a step input. A Simulink block diagram can be established as shown in Fig. 3 for the

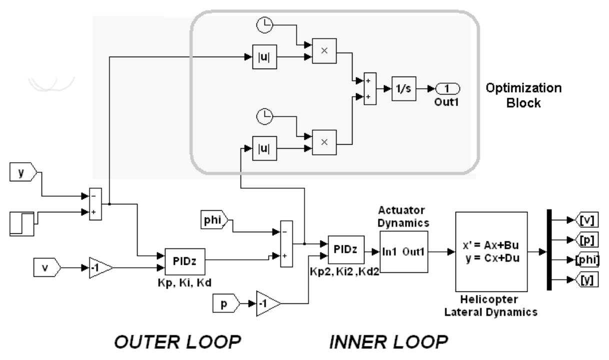

Figure 3: Optimal gain tuning of attitude and position PID controllers for lateral dynamics.

lateral channel, where the ITAE criterion can be evaluated as shown (a similar model is established for the longitudinal channel, as well). A function of Matlab optimization toolbox (fminsearch) is called to calculate the minimum of the objective function.

4 Simulation of Visual Snapshot

OpenGL software is used to rapidly draw the simulated view from a downwards looking camera onboard the UAV. Figure 4 shows the top level view of the complete version of our guidance simulator, involving the vision subsystems, implemented in Simulink. The vision subsystems contain blocks for generating the simulated view from a downwards looking camera. The image stream from each simulated view is fed into optic flow processing block which calculates a vector flow field for each image frame. The OpenGL code is written in C++ and compiled to provide a dynamic link library executable (.dll) file that will run under the Windows operating system. The .dll library routines are called from inside the Simulink program. The routines are called every 0.02 seconds of simulation time to simulate a camera speed of 50 frames per second, noting we have already been able to demonstrate 50 frames per second computation of optic flow on an FPGA based image processing engine [Garratt *et al.*, 2009]. The .dll function calls are passed input variables corresponding to the UAV position and attitude calculated in the simulation. The OpenGL executable program outputs the images as a 400×400 pixel 24-bit color image. The color image is converted into an array of 8-bit intensity values by averaging the intensities from the red, green and blue color channels. The images are displayed as the simulation proceeds.

For the purposes of this work, a planar grass environment is simulated. Photographs of a real grass field were taken with a digital camera and then converted into a 6x6 grid of bitmap texture tiles, with each tile consisting of 256x256 pixels. By mirroring the 36 tiles in both x and y directions, an endless grass environment was generated which does not have visible boundaries where the tile pattern repeats. Figure 5 shows the view from a downwards looking cameras when the helicopter is flying above the grass tiles.

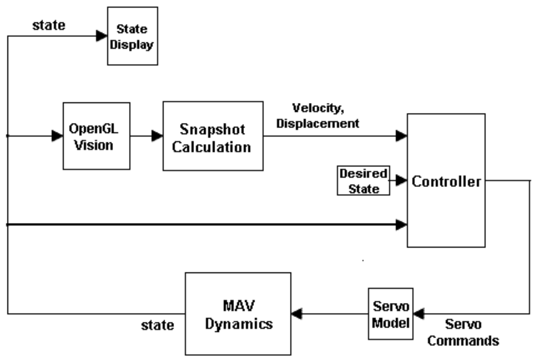

Figure 4: SIMULINK Simulation of Closed Loop MAV with Visual Feedback.

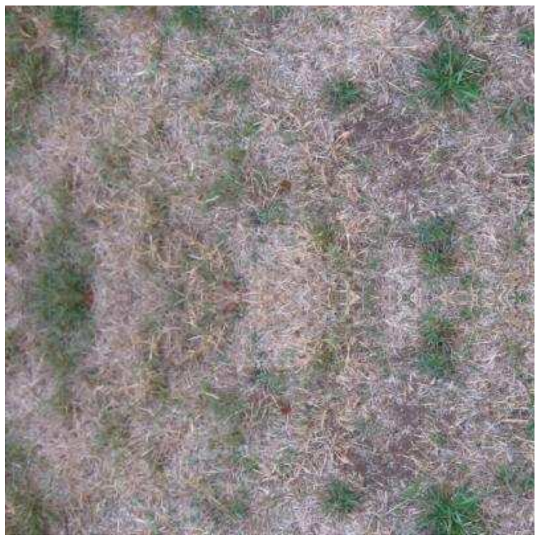

Figure 5: Simulated Image from Downwards Looking Camera.

4.1 Image Processing

The stream of pixel data from the OpenGL graphics blocks passes to a subsystem that calculates the image motion and image displacement. Both image processing tasks are completed using the Image Interpolation Algorithm (I^2A) [Srinivasan, 1994] to compute optic flow. Various versions of the I^2A exist for combinations of rotations, shear and translation. However, for small rotations between frames, it is possible to only treat the cases of translation as the other image transformations can be reconstructed based on the distribution of vectors in the flow field. A detailed derivation of the I^2A

algorithms can be found in [Srinivasan, 1994].

The simple I^2A algorithm involving only lateral and longitudinal translations is implemented in the C language and called by Simulink using an s-function. A graphical mask is applied to the Simulink s-function which allows the size of patches and the reference shifts to be varied. Flow estimates are calculated at every pixel in the image.

Simulink blocks have also been developed to pre-filter the images passed to the blocks which calculate image motion. These blocks also have graphical masks that allow the filter parameters to be conveniently changed. Additional blocks to subsample the optic flow results and to calculate average translations and image loom have also been created.

The snapshot calculations are carried out using a modified I^2A block that allows an arbitrary stored frame to be used as the reference image. For snapshot calculations, a large 128x128 pixel patch with a 32 pixel reference shift was found to be most suitable. The image was heavily pre-filtered using a 32x32 boxcar filter prior to the snapshot calculation being used. The output of the snapshot was subsampled to 16x16 vectors which were then averaged to give the translatory displacements.

5 2D Snapshot Simulation

The scenario chosen to test the closed loop hover was a transition from low speed forward flight to hover. The H-610 was setup with an initial velocity of 2m/s at an altitude of 1m. The height is controlled using a PID controller and an unspecified height sensor (e.g. sonar as used to control height in the AR Drone quadrocopter [Bills and Yosinski, 2010]). Yaw is controlled by another PID controller with feedback from an unspecified heading sensor such as a magnetometer. No attempt is made to simulate height or yaw sensor errors for the 2D snapshot simulation, as this is not the focus of the 2D snapshot. After t=0 seconds, optic flow damping [Garratt and Chahl, 2007] is activated to slow the horizontal speed of the helicopter. This damping is implemented as velocity feedback with no position feedback. After 3 seconds, once the speed has been substantially arrested, a trigger on the snapshot block is set which causes a single snapshot reference image to be taken. From this point onwards, the snapshot block outputs translatory displacement in pixels based on the shift between the reference snapshot and the current image. The translatory displacement is used for position feedback from this point onwards whilst the velocity feedback (optic flow damping) continues. The simulation continues for 10 seconds.

Once the simulation begins, the height sensor is not used to set the scale of the optic flow. Rather a fixed scale corresponding to the initial altitude of 1m is used. The optic flow is converted from pixels per frame to radians per second based on the known geometry of the camera. This optic flow is then converted into a velocity estimate by multiplying by the assumed height. For the purposes of this simulation, no attempt is made to compensate the optic flow measurements to remove the effect of pitch rate and roll rate, although this could be done in the future assuming the presence of inertial sensors. The displacement in pixels from the snapshot block is also converted into a displacement in meters by multiplying by the assumed height. No attempt is made to compensate the displacement measurement by subtracting the effect of attitude changes (pitch and roll). The effects of yaw and loom are also not compensated for. The raw optic flow and snapshot displacement are used as the only feedback signals in a position outer loop controlling the helicopter position. An attitude control inner loop is used to stabilise the helicopter and receives attitude commands from the outer loop as per the control paradigm described in section 1.

Atmospheric gusts and turbulence corresponding to 5 m/s mean wind speed at ground height are simulated using a Dryden wind gust profile. Prior to generation of the OpenGL imagery, the state of the helicopter is augmented with motions equivalent to 2g horizontal and 3g vertical vibration acting at the main rotor speed (294 RPM).

Figures 6 shows the optic flow and snapshot displacement estimated during the simulation. The position and velocity of the helicopter during the simulation, are depicted in Fig. 7. Figures 8 and 9 show a comparison between the visual results and the exact results corresponding to the state of the helicopter. The discrepancies between the results can be attributed to the lack of compensation for the attitudinal changes of the helicopter, effects of changing altitude and errors arising from calculation of the optic flow using the I^2A. Overall, the correlation between the visual results and the state of the helicopter are very good, given that the displacements are very small, in the order of only a few centimeters.

The effect of the optic flow damping results in a very rapid arrest of velocity which brings the helicopter from 2m/s to less than 1m/s in the first 1 second. The helicopter fluctuates vertically by about 10cm during the entire manoeuvre and about 2cm during the snapshot phase. The heading of the helicopter varies less than 4 degrees during the 10 second simulation.

6 3D Snapshot Hover Simulation

For the 3D snapshot simulation, the height of the helicopter is also determined from the output of the snapshot block. In this case, the loom of the image is used to calculate the vertical displacement. Image *loom* corre-

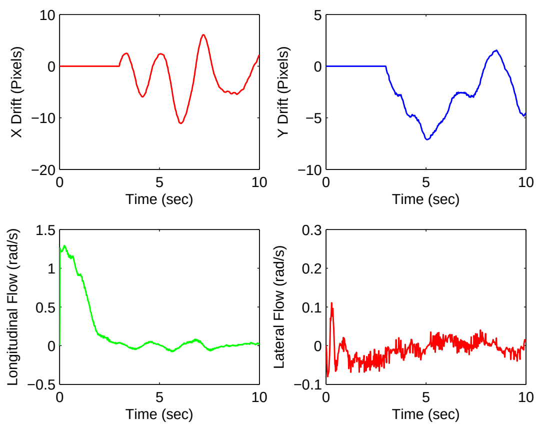

Figure 6: 2D Snapshot Hover: Optic flow and Snapshot Displacement (displacement measured from snapshot activation at $t = 3sec$).

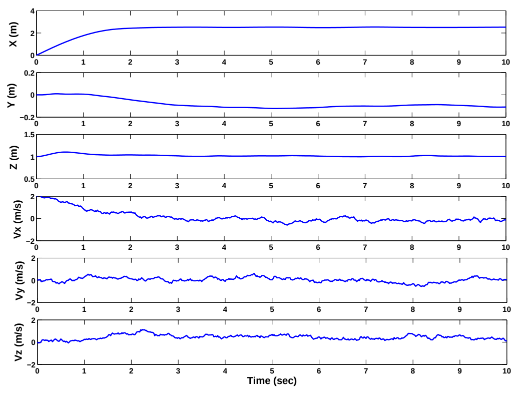

Figure 7: 2D Snapshot Hover: Position and Velocity Results.

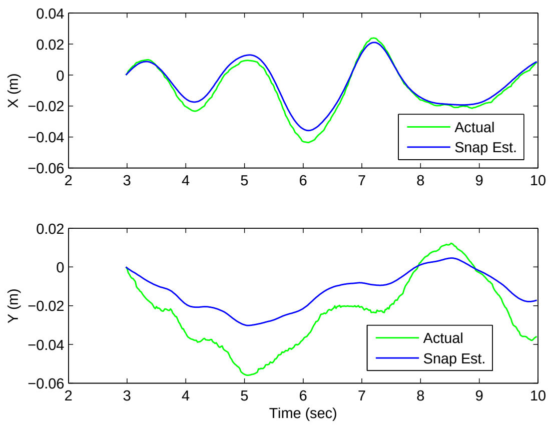

Figure 8: 2D Snapshot Hover: Comparison of actual helicopter position with estimated displacement from snapshot.

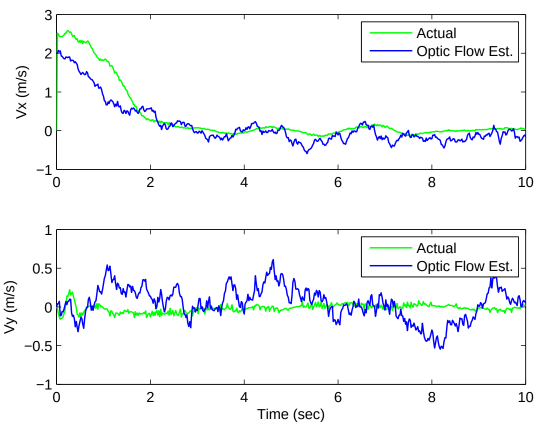

Figure 9: 2D Snapshot Hover: Comparison of actual helicopter velocity with estimate from optic flow.

sponds to image expansion or contraction due to motion perpendicular to the image plane of the camera. We have implemented a method for extracting the loom value by subtracting the sum of the flow vectors on the left-hand side of the image from the sum of the flow vectors on the right-hand side of the image. This is done by making use of equation (9), which is derived in [Garratt and Cheung, 2009], to calculate the image loom from the measured optic flow field. The loom term $L = \dot{Z}/Z$ represents the time to contact.

$$L = \frac{\dot{Z}}{Z} = \left[\sum_{left} Q_u(x,y) - \sum_{right} Q_u(x,y)\right] \times \left[2\sum_{right} U\right]^{-1} \tag{9}$$

where $Q_u(x,y)$ are the horizontal components of the optic flow field defined at corresponding horizontal locations (U,V) in the image plane.

Given a previous range estimate (Z), the loom calculated by comparing consecutive frames can be used to estimate the vertical velocity $w = LZ$. The estimate of w can then be used to damp out vertical motions of the helicopter and provide stable control of height. If the height of the helicopter is known when the snapshot is taken, the loom between the snapshot frame and current frame (L_{snap}) can also be used to estimate the change in height using equation 10.

$$\Delta Z = Z L_{snap} \tag{10}$$

The scenario for this simulation was hover at 1m height above ground. The helicopter is assumed to have zero velocity at the start of the simulation. A snapshot is taken at t=0.04 seconds and all consecutive frames are compared with this datum frame to provide absolute displacement. The initial height of the helicopter at t=0sec, (Z_0) is used to provide the scale for all future position and velocity estimates. Horizontal (X and Y) displacement is controlled in the same manner as for the 2D snapshot simulation. Vertical displacement is controlled using the ΔZ estimate from equation 10 and the w computed using a combination of loom and vertical acceleration. The frame-to-frame loom measurement is quite noisy and use of integrated accelerometers assists with smoothing out the noise.

A state prediction and correction cycle is used to estimate the height above terrain Z and vertical velocity w. First, at each inertial sensor sample time ΔT, the relative vertical velocity estimate is updated using the vertical accelerometer measurement a_z as per Equation (11). A measurement of the vertical speed is determined from the loom calculation. The height estimate is updated using the relative vertical velocity estimate and corrected using the range from snapshot loom (L_{snap}). The prediction and correction cycle is outlined below.

Predicting the relative vertical velocity estimate $\hat{w}$, where g is the acceleration due to gravity:

$$\hat{w}_k^- = \hat{w}_{k-1}^+ + (a_z + g)\Delta T \tag{11}$$

Updating the height estimate Z from the velocity estimate $\hat{w}$:

$$\hat{Z}_k^- = \hat{Z}_{k-1}^+ + \hat{w}_{k-1}^- \Delta T \tag{12}$$

Calculating the loom range $\mathcal{R}$ from the snapshot loom L_{snap} and the initial height of the helicopter Z_0 when the snapshot was taken:

$$\mathcal{R}_k = Z_0 L_{snap} \tag{13}$$

Conditioning the relative velocity estimate from the frame-to-frame loom where the constant α is a filtering parameter between 0 and 1:

$$\hat{w}_k^+ = (1-\alpha)\,\hat{w}_{k-1}^- + \alpha L Z_0 \tag{14}$$

Conditioning the height estimate from the loom range where the constant β is a filtering parameter between 0 and 1:

$$\hat{Z}_k^+ = (1-\beta)\,\hat{Z}_{k-1}^+ + \beta \mathcal{R}_k \tag{15}$$

The filter gains α and β were chosen as a compromise between correcting drift in the velocity and position estimates, and smoothing out the noise introduced by the loom measurement. Values of 0.1 were found to be suitable for both in simulation. Figure 10 shows the height and vertical velocity estimates during the 3D snapshot hover.

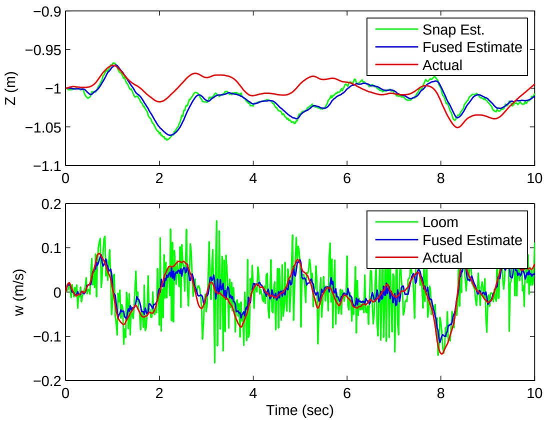

Figure 10: H610 3D Snapshot Hover: Height and vertical velocity estimates.

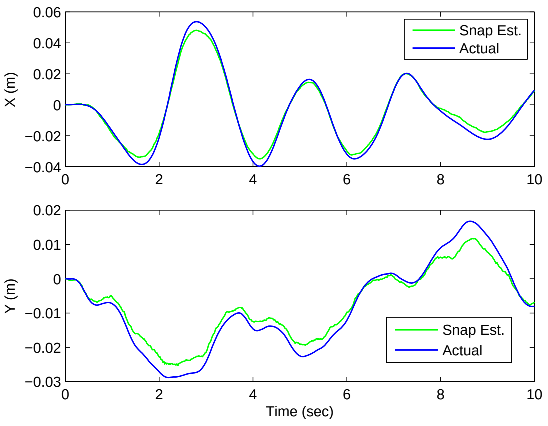

Figure 11: H-610 3D Snapshot Hover: Horizontal Position.

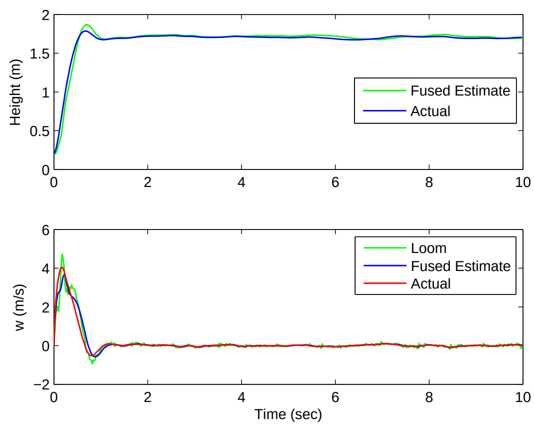

Figure 12: Bootstrapped take-off:Height and velocity estimates.

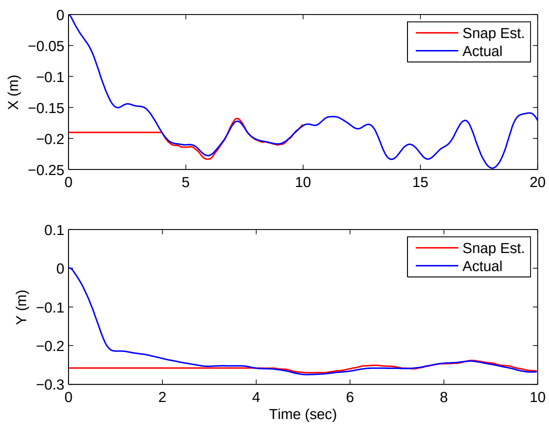

Figure 13: Bootstrapped take-off: Horizontal Position.

7 Bootstrapped take-off and Snapshot Hover Simulation

All of the snapshot schemes require an initial estimate of height from some source when the snapshot is taken. In this simulation, an attempt is made to bootstrap the height estimate from integrated optic flow loom information during the take-off phase. Given that the height of the camera is known when the helicopter is on the ground, the change in height at each frame can be calculated using equation 10. By integrating the change in height over time, an estimate of the total height can be provided. As the integration proceeds, the error between the actual height and the calculated height will increase due to accumulated offsets, however, provided the use of this estimate is only used for a short time, the height estimate may be satisfactory. At the end of the take-off manoeuver, a snapshot is taken, and the integrated z-estimate at that instant is used from that point on as the scaling factor in all future displacement and velocity calculations.

8 Conclusion

We have shown that for both small and medium sized unmanned helicopters, the use of visual snapshot techniques should allow the control of take-off, hover and the transition from low speed forward flight to hover. Through the use of 3D snapshot, it should be possible to control a small hovering vehicle visually without a separate height sensor or the use of bulky stereo vision.

9 Future Work

In future work, we intend to test the snapshot in flight using our Vario XLC gas-turbine helicopter. This helicopter has been instrumented with a laser-rangefinder for determining height (useful for benchmarking results and testing the 2D version of the algorithm), differential GPS and a PC104 flight computer with frame grabber for computing the snapshot parameters in real-time.

Acknowledgment

This work was carried out with funding provided by the Australian Defence Science and Technology Organisation.

References

[Ahrens *et al.*, 2009] S. Ahrens, D. Levine, G. Andrews, and J.P. How. Vision-based guidance and control of a hovering vehicle in unknown, GPS-denied environments. In *Robotics and Automation, 2009. ICRA'09. IEEE International Conference on*, pages 2643–2648. IEEE, 2009.

[Amidi, 1996] O. Amidi. *An Autonomous Vision-Guided Helicopter*. PhD thesis, Dept. of Electrical and Computer Engineering, Carnegie Mellon University, Pittsburgh, 1996.

[Beyeler *et al.*, 2007] A. Beyeler, J. Zufferey, and D. Floreano. 3D Vision-based Navigation for Indoor Microflyers. In *Proceedings of the 2007 IEEE International Conference on Robotics and Automation*, pages 1336–1341, Rome, Italy, 10-14 April 2007.

[Bills and Yosinski, 2010] C. Bills and J. Yosinski. MAV stabilization using machine learning and onboard sensors. Technical Report CS6780, Cornell University, 2010.

[Blosch *et al.*, 2010] M. Blosch, S. Weiss, D. Scaramuzza, and R. Siegwart. Vision based MAV navigation in unknown and unstructured environments. In *Robotics and Automation (ICRA), 2010 IEEE International Conference on*, pages 21–28. IEEE, 2010.

[Cartwright and Collet, 1983] B. A. Cartwright and T. S. Collet. How honey bees use landmarks to guide their return to a food source. *J.Comp Physiol*, 151:521–543, 1983.

[Cartwright and Collet, 1992] B. A. Cartwright and T. S. Collet. Landmark learning in bees. *Nature*, 295:560, Feb 1992.

[Castillo *et al.*, 2005] C. Castillo, W. Alvis, M. Castillo-Effen, K. Valavanis, and W. Moreno. Small scale helicopter analysis and controller design for non-aggressive flights. *IEEE International Conference on SMC, Hawaii*, October 2005.

[Cherian *et al.*, 2009] A. Cherian, J. Andersh, V. Morellas, N. Papanikolopoulos, and B. Mettler. Autonomous altitude estimation of a UAV using a single onboard camera. In *Intelligent Robots and Systems, 2009. IROS 2009. IEEE/RSJ International Conference on*, pages 3900–3905. IEEE, 2009.

[Corke, 2004] P. Corke. An inertial and visual sensing system for a small autonomous helicopter. *Journal of Robotic Systems*, 21(2):43–51, 2004.

[C.S.Sharp *et al.*, 2001] C.S.Sharp, O.Shakernia, and S.S.Sastry. A vision system for landing an unmanned aerial vehicle. In *IEEE International Conference on Robotics and Automation*, 2001.

[Garratt and Chahl, 2007] M. A. Garratt and J. S. Chahl. An optic flow damped hover controller for an autonomous helicopter. In *Proceedings of the 22nd International UAV Systems Conference*, Bristol, UK, 16-18 April 2007.

[Garratt and Chahl, 2008] M. A. Garratt and J. S. Chahl. Vision-based terrain following for an unmanned rotorcraft. *Journal of Field Robotics*, 25(4-5):284–301, 2008.

[Garratt and Cheung, 2009] M. A. Garratt and A. Cheung. Obstacle avoidance in cluttered environments using optic flow. In *Proceedings of the Australian Conference on Robotics and Automation*, Sydney, Australia, 2-4 December 2009.

[Garratt *et al.*, 2009] M. A. Garratt, A.Lambert, and T.Guillemette. FPGA implementation of an optic flow sensor using the image interpolation algorithm. In *Proceedings of the Australian Conference on Robotics and Automation*, Sydney, Australia, 2-4 December 2009.

[Garratt, 2007] M. A. Garratt. *Biologically Inspired Vision and Control for an Autonomous Helicopter*. PhD thesis, Research School of Biological Sciences, Australian National University, Canberra, Australia, 2007.

[Griffiths *et al.*, 2006] S. Griffiths, J. Saunders, A. Curtis, B. Barber, T. McLain, and R. Beard. Maximizing miniature aerial vehicles: Obstacle and terrain avoidance for MAVs. *IEEE Robotics and Automation Magazine*, pages 34–43, 2006.

[Keyence Corporation, 2011] Keyence Corporation. Revolutor H-610 operating instructions. `http://hobby.keyence.co.jp/english/manual.html`, 2011. accessed 1 Nov 2011.

[L.Mejias *et al.*, 2006] L.Mejias, S.Saripalli, P.Campoy, and G.S.Sukhatme. Visual servoing of an autonomous helicopter in urban areas using feature tracking. *Journal of Field Robotics*, 23:185–199, 2006.

[Mettler *et al.*, 2000] B. Mettler, M. B. Tischler, T. Kanade, and W. Messner. Attitude control optimization for a small-scale unmanned helicopter. *AIAA Guidance, Navigation and Control Conference*, pages 40–59, 2000.

[Mettler, 2003] B. Mettler. *Identification Modeling and Characteristics of Miniature Rotorcraft.* Kluwer Academic Publishers, Norwell, MA, September 2003.

[O.Amidi *et al.*, 1999] O.Amidi, T.Kanade, and K.Fujita. A visual odometer for autonomous helicopter flight. *Robotics and Autonomous Systems*, 28(2):185–193, 1999.

[O.Shakernia *et al.*, 2002] O.Shakernia, R.Vidal, C.S. Sharp, Y. Ma, and S.Sastry. Multiple view motion estimation and control for landing an unmanned aerial vehicle. In *Proceedings of the IEEE International Conference on Robotics and Automation,2002.*, volume 3, pages 2793–2798, Nov 2002.

[Padfield, 2007] G. D. Padfield. *Helicopter Flight Dynamics.* Blackwell, 2007.

[Prouty, 1990] R. W. Prouty. *Helicopter Performance, Stability, and Control.* Robert E. Krieger Publishing Company, 1990.

[Srinivasan, 1994] M. V. Srinivasan. An image-interpolation technique for the computation of optic flow and egomotion. *Biological Cybernetics*, 71(5):401–415, 1994.

[S.Saripalli *et al.*, 2003] S.Saripalli, J.F.Montgomery, and G.S.Sukhatme. Visually guided landing of an unmanned aerial vehicle. *IEEE Transactions on Robotics and Automation*, 19(3):371–380, June 2003.

[Zingg *et al.*, 2010] S. Zingg, D. Scaramuzza, S. Weiss, and R. Siegwart. MAV navigation through indoor corridors using optical flow. In *Robotics and Automation (ICRA), 2010 IEEE International Conference on*, pages 3361–3368. IEEE, 2010.