Integration of a PMD CamBoard nano onto an AscTec Pelican Quadrotor for Ground Plane Detection

Zoey F. Freeman

University of New South Wales at the Australian Defence Force Academy

Over the past decade, the demand for 3D ranging cameras have become increasingly popular. The 3D ranging technique is based on the Time-of-Flight principle. In this thesis, an experimental investigation of the possibility of utilising the pmd[vision]® CamBoard nano as an effective 3D depth camera was carried out.

Due to the CamBoard nano being optimised for close range gesture control, chessboard calibration techniques were utilised in conjunction with variable integration time to increase the acceptable range to 10 cm – 150 cm. Teamed with the AscTec Pelican Quadrotor, the CamBoard nano accurately estimates the ground plane using Least Squares approximation.

Table of Contents

I. INTRODUCTION ......................................................................................................................... 1
II. ALTITUDE DETERMINATION ................................................................................................. 1
   A. 3D Depth Cameras ............................................................................................................... 1
      i. PMD Camera ..................................................................................................................... 2
         a. Calibration ....................................................................................................................... 2
         b. Integration Time Setup ................................................................................................. 3
         c. Flags – Bad Pixel Removal ............................................................................................ 3
      ii. Ground Plane Detection ............................................................................................... 4
         a. Least Squares Approximation ....................................................................................... 4
   B. AscTec Pelican Quadrotor .................................................................................................. 5
III. INTEGRATION .......................................................................................................................... 5
   A. Robot Operating System .................................................................................................... 6
   B. Mounting ............................................................................................................................. 6
   C. Communication .................................................................................................................. 6
   D. Limitations ......................................................................................................................... 6
IV. VERIFICATION .......................................................................................................................... 7
   A. Algorithm ............................................................................................................................ 7
   B. Static Testing ....................................................................................................................... 8
      i. Indoor Testing .................................................................................................................. 8
   C. Real-time Testing ................................................................................................................. 9
V. RECOMMENDATIONS ............................................................................................................... 10
VI. CONCLUSION ........................................................................................................................ 10

1 PLTOFF, School of Engineering & Information Technology. ZEIT4500
Nomenclature

$\lambda$ Plane coefficients vector
$K_1, K_2, K_3$ Coefficients of $\lambda$
$\varepsilon$ Error residuals
$A$ A matrix for Least Squares

$H$ Height above the plane
$\theta$ Pitch angle
$\phi$ Roll angle

Abbreviations

<table>
<thead>
<tr>
<th>AscTec</th>
<th>Ascending Technologies</th>
<th>PMD</th>
<th>Photonic Mixer Device</th>
</tr>
</thead>
<tbody>
<tr>
<td>API</td>
<td>Application Program Interface</td>
<td>RANSAC</td>
<td>RANdom SAmple Consensus</td>
</tr>
<tr>
<td>FoV</td>
<td>Field of view</td>
<td>RGB</td>
<td>Red, Green, Blue (colour space)</td>
</tr>
<tr>
<td>FPS</td>
<td>Frames per second</td>
<td>RPM</td>
<td>Revolutions Per Minute</td>
</tr>
<tr>
<td>GPS</td>
<td>Global Positioning System</td>
<td>ROS</td>
<td>Robot Operating System</td>
</tr>
<tr>
<td>INS</td>
<td>Inertial Navigation System</td>
<td>IR</td>
<td>Infrared</td>
</tr>
<tr>
<td>LED</td>
<td>Light-emitting diode</td>
<td>SBI</td>
<td>Suppression of Background Illumination</td>
</tr>
<tr>
<td>LiDAR</td>
<td>Light Detection And Ranging</td>
<td>RGB</td>
<td>Red, Green, Blue (colour space)</td>
</tr>
<tr>
<td>MAV</td>
<td>Micro-unmanned Aerial Vehicle</td>
<td>RF</td>
<td>Radio Frequency</td>
</tr>
<tr>
<td>MDK</td>
<td>MATLAB Development Kit</td>
<td>RPM</td>
<td>Revolutions Per Minute</td>
</tr>
<tr>
<td>OS</td>
<td>Operating System</td>
<td>SBI</td>
<td>Suppression of Background Illumination</td>
</tr>
<tr>
<td>PID</td>
<td>Proportional-integral-derivative</td>
<td>ToF</td>
<td>Time-of-flight</td>
</tr>
<tr>
<td>VBFC</td>
<td>Vision based flight control</td>
<td>UAV</td>
<td>Unmanned Aerial Vehicle</td>
</tr>
<tr>
<td>VTOL</td>
<td>Vertical Take-Off and Landing</td>
<td>VBFC</td>
<td>Vision based flight control</td>
</tr>
</tbody>
</table>
I. INTRODUCTION

The use of Unmanned Aerial Vehicles (UAV) has become an increasingly popular research field with special interest for military operations and applications. Tasks that were once performed by humans are being accomplished by autonomous vehicles. Due to the recent success of UAV, researchers are progressively looking into a smaller, more manoeuvrable version, known as Micro-unmanned Aerial Vehicles (MAV), to operate in hard to reach areas, such as in buildings and through tunnels [1]. In these environments, conventional sensor and navigation equipment (such as GPS, IR, INS, and Radar techniques) may be unavailable, resulting in a ‘fully autonomous’ solution to be required [2].

Attributable to the nature of MAV, there are constraints associated with the size, weight, and power requirements, which restrict the choices available in terms of sensors, cameras, and processors. Due to these constraints, the PMDTechnologies GmbH’s pmd[vision]® CamBoard nano has been chosen as the image sensor for this project. This will be integrated into UNSW@ADFA’s Ascending Technologies (AscTec) Pelican quadrotor to provide ground plane determination, leading to hover control for the Pelican.

There have been many papers published on autonomous UAV take-off and landing using Visual Based Flight Control (VBFC) techniques, with particular focus on MAV. Vision-based methods are an ideal alternative, as they require simple hardware, meaning a limited number of cameras. UNSW@ADFA have a large involvement with the development of UAV technology, with particular interest in helicopters (or quadrotors). Although being commercially available (such as the Parrot AR. Drone), the area focuses on fully autonomous systems, namely autonomous take-off and landing [3] [4] [5] [6] [7]. This project seeks to continue on this path providing further research into the use of alternative depth sensors for MAV.

The aim of this project is to develop the pmd[vision]® CamBoard nano for use on the AscTec Pelican quadrotor to determine the altitude. The CamBoard nano provides a small sensor for ground plane detection using Vision Based Flight Control (VBFC) methods.

II. ALTITUDE DETERMINATION

A. 3D Depth Cameras

Time-of-Flight (ToF) cameras are widely used spanning across many different fields. They allow 3D point clouds to be modelled at video frame rates. The most marketable feature of the ToF camera is the ability to maintain a small, compact size while delivering high frame rates and a large Field-of-View (FoV). Laser scanners are still considered to be the most common sensors for 3D mapping due to large fluctuations in accuracy and precision for a ToF camera [8]. However, ToF cameras are amongst the only sensors able to deliver 3D depth information in real time [9].

There are two key measurement principles to determine distance using light: the phase delay between the emitted and reflected signals, and the amplitude of modulated light, used to determine the validity of the calculated depth data [10]. Typical ToF cameras use a symmetric array of LED diodes to mimic a cone of light [11]. Due to this, most ToF cameras are inoperable in sunlight due to the background not being suppressed. In addition, reflective surfaces, like glass, or glossy finishes produce invalid pixel measurements due to the total reflection of infrared components in the light spectrum away from the receiver optics [12].

A ToF camera consists of the following components: [13]

1) **Illumination Unit** – Normally using IR, this brightens the image, by using a modulation frequency up to 100 MHz;

2) **Optics** – The surrounding environment is gathered onto the image sensor using a lens;

3) **Image sensor** – Time taken for the illumination unit to emit and detect the light at each pixel;

4) **Driver electronics** – Controls the illumination unit and image sensor; and

5) **Computation/Interface** – Calculates the distance based on the sensor outputs. This needs to be calibrated based each ToF camera.

There are many ToF cameras available on the market which have been used for research purposes. The most documented is the Microsoft Kinect Sensor. The biggest differences between the CamBoard nano and the Kinect is the size, high frame rate, and large FoV. The Kinect however has a higher resolution and greater range [14].
i. PMD Camera

The Photonic Mixer Device (PMD) is a recent semiconductor sensor created by Prof. R. Schwarte for S-TEC GmbH in 1997. The PMD is able to create a matrix of distances simultaneously where each pixel is effectively its own ranging unit [15]. Like conventional ToF cameras, the CamBoard nano measures distance via the phase delay between the emitted and received light. Ensuring that the path length is the same between the PMD image sensor and object, as well as the object and PMD receiving optic. This is done to ensure that the distance to the object is simply the measured path length divided by two [16].

The pmd[vision]® CamBoard nano is the second most compact Time-of-Flight depth sensor reference design available worldwide. The camera is designed by PMD Technologies who offer a large community for developers [17]. They provide a well-documented SDK with the ability to control all parameters and access to other developers through a forum, programming tools and documentation.

The CamBoard nano is primarily designed for gesture recognition utilising a 90° horizontal x 68° vertical FoV and up to 90 fps, intended to allow natural motions without restrictions at a working range of 5 cm to 50 cm. With dimensions of 37 mm x 30 mm x 25 mm, and a weight of 34 g, the camera uses one LED (OSRAM Dragon SFH4235) and one sensor (pmd PhotonICs® 19k-S3 with SBI) to deliver four kinds of data per pixel. The resolution is 165 x 120, where each pixel can return a depth value, amplitude value, flag value, and the XYZ-Cartesian coordinates [18].

The depth values returned specify the distance along the line of sight to an observed object. This is not the height of the sensor above the plane but the diagonal distance from the sensor to a point on the plane. The amplitude value relates to the signal strength, where it describes the amount of received modulated light. This value is independent of ambient light, as it looks through only the IR LED of the CamBoard nano. This means that the CamBoard nano is able to operate in both high and low light. The flag value returned by the CamBoard nano provides information on the accuracy of that pixel where the following flags are available; invalid, saturated, low signal, and inconsistent. The XYZ-coordinates provide a matrix representation of the coordinates computed from the measured distance aligned pixel wise in a data array \((x_n, y_n, z_n)\) [19]. Unlike previous pmd[vision]® ToF cameras, the CamBoard nano has a set modulation frequency of 30 MHz which cannot be altered to improve the range or to allow the operation of multiple ToF cameras simultaneously.

The CamBoard nano is an active sensor, meaning that it emits its own light via an IR LED. This then reflects off the surface, and back to the image sensor. The time taken and quality of the returned signal is interpreted to produce a depth map. The CamBoard nano is chosen over other range finding devices not only due to the small size and weight being ideal for MAV, but also for the actively illuminated full view different from other devices such as ultrasonic sensors and LiDAR [20].

These devices require a high minimum distance and have lower accuracy due to the single point illumination. The inbuilt suppression of background illumination (SBI) within the sensor allows the CamBoard nano to work in both high and low light environments. It works by emitting intensity-modulated infrared light, using the IR LED, modulated at 30 MHz. In ambient light, each pixel receives both the intensity-modulated IR LED and the constant-intensity ambient light. The SBI occurs within each pixel at the time of measurement and ensures that only the photoelectrons from intensity-modulated light is detected by each pixel [21]. This is a significant distinction between the CamBoard nano and other ToF cameras such as the Microsoft Kinect, which is highly effected by sunlight [22].

a. Calibration

The image sensor contains a small fish-eye lens causing distortion, therefore to acquire an accurate depth map the lens must be calibrated to straighten the image. Calibration also allows for the pixels of the distance image to be converted to 3D coordinates – the actual dimensions of each pixel can be determined from its calculated height. The accuracy of the calibration is paramount in order to calculate the ground plane of the surface.

Due to the mass market production of the CamBoard nano, a detailed individual calibration has not been provided. As such, in order to increase the range and accuracy of the CamBoard nano for this application, it requires further calibration. The method of calibration used was the chessboard approach shown in Figure 2. With the intention of increasing the accurate camera range, the pixel and radial distortion of the image was altered to allow for a larger, although slightly less

---

2 As of June 2013 the CamBoard nano was be superseded by the pmd[vision]® CamBoard pico as the world’s smallest ToF camera [35]

3 To validate the effectiveness of operation in diverse lighting conditions, results were gathered in both direct sunlight and pitch black lighting showing a very small deviation between them.
accurate range. The Metrilus MetriCam SDK\(^4\) toolbox was used to determine appropriate lens parameters in order to optimise the camera calibration file provided with the CamBoard nano [23]. The lens values were then modified using Lens Tool 0.9 [24].

b. Integration Time Setup

The integration time is said to be the ‘most important intrinsic parameter’ of ToF cameras [25]. Integration time is analogous to the exposure time or shutter speed for image sensors (i.e. photography). It describes the time period in which incoming photons are detected for one measurement cycle, in order to derive the phase shift and corresponding distance. The stability of the distance measurements depends on the integration time, as it is increased, the sensitivity of the device also increases. The integration time also refers to the relationship between the strength of the reflected signal and accuracy of the distance measurement.

The integration time can be used to tune the depth measurement received by the CamBoard nano. In particular, the integration time must be adapted according to distances. The integration time can be adjusted by using `pmdSetIntegrationTime` command in the SDK, or in LightVis (real-time visualisation tool for CamBoard nano). The CamBoard nano supports integration times between 12 \(\mu s\) and 2000 \(\mu s\), this was found using the above command in MATLAB, returning an error when out of range.

![Figure 3. Different integration times at a height of 10 cm demonstrating the effect of integration time on surface reflection and proximity. All invalid pixels are set to be black.](image)

Choosing a value too low will yield noisy data due to low received signal. Similarly, an integration time too high can lead to inconsistency of detected depth. In both of these cases, an invalid depth measurement will be given. The disparity in luminance depicts varying intensities of pixels. Inconsistent pixels are due to multiple received signals for a single pixel during the integration time period. The surface material and reflection properties of different materials also have an impact on the integration time setting and overall accuracy of the depth image.

Setting a very high integration time is not always beneficial, and this can be adjusted to suit individual need. Ideally, the lowest possible integration time whilst receiving the depth resolution is desired. The selected integration time should be carefully chosen to maximise FPS for the intended distances whilst limiting the number of invalid pixels and power consumption.

c. Flags – Bad Pixel Removal

Flags contain additional information about the pixel, whether or not it is corrupted and considered to be correct. The value returned by the flag indicates the reason for the invalidity. The classifications for flags are as follows: invalid, saturated, inconsistent, low signal, or SBI active as seen in Table 1 [19]. For all cases mentioned, the invalid flag is also set, meaning that all flags represent an unreliable distance.

Flags were used in this project to discount invalid pixels from the ground plane calculations. The reason for doing this was to reduce the outliers in order to produce the most accurate ground plane using minimal processing power and time. Unfortunately, there are cases where the ratio of invalid to valid pixels is too high, causing a small subset of the 3D point cloud to be used, resulting in potential errors in a varied environment.

\[\text{Figure 4. Visual output of CamBoard nano showing discounted flags, where red pixels represent flag values 9&13}\]

---

\(^4\) This work was supported by MetriCam SDK from Metrilus. It is available at www.metricam.net.
Table 1. Flag values associated with types [19]

<table>
<thead>
<tr>
<th>Flag Type</th>
<th>Description</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>SBI Active</td>
<td>Valid pixel</td>
<td>0</td>
</tr>
<tr>
<td>Invalid</td>
<td>Does not represent a reliable distance</td>
<td>1</td>
</tr>
<tr>
<td>Saturated</td>
<td>Overexposed(^3)</td>
<td>3</td>
</tr>
<tr>
<td>Inconsistent</td>
<td>Data values are inconsistent with each other, can be due to motion artefacts</td>
<td>5</td>
</tr>
<tr>
<td>Low Signal</td>
<td>Not a high enough signal for accurate measurement</td>
<td>9</td>
</tr>
<tr>
<td>Low Signal &amp; Inconsistent</td>
<td>Both cases apply</td>
<td>13</td>
</tr>
</tbody>
</table>

ii. **Ground Plane Detection**

In order to calculate the height of the MAV, the ground plane must first be detected and mapped based on the data received from the CamBoard nano. This is an integral part of the project and forms the basis of future work. A ground plane in this context is defined as having the majority of data points located within a specified tolerance. If the ground plane can be accurately matched, then the altitude of the MAV can be controlled using VBFC techniques. Fitting a plane through a 3D point cloud defines the relative distance and orientation with respect to the camera. There are many approaches to determining a ground plane based solely on visual data and point clouds.

Being able to detect the ground plane not only aids the achievement of altitude control, but also assists in object detection. Although ground plane detection and mapping is seen to be relatively simple, there are many obstructions that can impair the camera from accurately computing the ground plane. Computing the ground plane whilst including these obstructions results in an inaccurate outcome. This means that in order to compute the optimal plane, obstructions, or outliers, must not be considered through calculations. Specific environments require different methods in calculating the ground plane, and therefore the procedure in obtaining feature points differ according to it.

There are several approaches to determine the ground plane, as recorded below:

1) **3D Hough transform method** – finds the equation for various lines (shape detector) within an image [26];
2) **Homography-based methods** – line detection designed specifically for the environment with expected coherent motion patterns [27], [28];
3) **RANSAC method** – used for robust estimation, where points are randomly chosen to construct a plane, while the remained test the estimation [4]. However, this method fails if there are too many outliers; and
4) **Least Squares Approximation** – finds the 3D regression model to reduce the residual error [29].

Many of these methods require a specific environment in order to calculate an accurate ground plane. The above named approaches is not exhaustive and is not limited to those mentioned. From this list, Least Squares approximation will be investigated further.

a. **Least Squares Approximation**

The general equation of a plane in 3D space is calculated using Equation (1), where the coefficients \( K_1, K_2, \) and \( K_3 \) are real integers represented as the coefficient vector, \( \lambda \).

\[
K_1x + K_2y + K_3z = 1 \tag{1}
\]

\[
\lambda = [K_1 \quad K_2 \quad K_3]^T \tag{2}
\]

Once the equation of the plane is calculated, the orientation of the plane, or the roll (\( \phi \)) and pitch (\( \theta \)) of the surface can be found using Equation (3). The roll is defined as the inclination of the plane from the \( x \)-axis, and the pitch is the inclination of the plane from the \( y \)-axis.

\[
\theta = \sin^{-1}(K_1) \quad \phi = \tan^{-1}\left(\frac{K_2}{K_3}\right) \tag{3}
\]

Using the equations from Garratt et al. [29], the ground plane coefficients were calculated to find the Least Squares approximation of the ground plane. The Least Squares approximation algorithm aims to minimise the residual error, \( \varepsilon \), found in Equation (4).

\[
\varepsilon = \sum_{i=1}^{n} (1 - k_1x_i - k_2y_i - k_3z_i)^2 \tag{4}
\]

\(^3\) Saturated pixels are unable to be determined via flags on this device, an invalid pixel is set.
In order to achieve this, the 3D coordinate data is rearranged into the form required by Equation (5), then manipulated to find the coefficient vector, $\lambda$, as in Equation (6).

$$
A = \begin{bmatrix}
    x_1 & y_1 & z_1 \\
    x_2 & y_2 & z_2 \\
    \vdots & \vdots & \vdots \\
    x_n & y_n & z_n
\end{bmatrix}
$$

(5)

$$
\lambda = (A^T A)^{-1} A^T b
$$

(6)

where:

$$
b = [1 \quad 1 \quad \ldots \quad 1]^T
$$

Subsequently the instantaneous height, $H$, is calculated using Equation (7).

$$
H = \frac{1}{K_3}
$$

(7)

B. AscTec Pelican Quadrotor

A quadrotor is a Vertical Take-Off and Landing (VTOL) rotary aircraft that has four rotors, or propellers. The propellers are fixed pitch as opposed to complex variable pitch swash plate mechanisms of conventional helicopters [30]. Control of a quadrotor is achieved by changing the RPM of a single rotor independently of the others, altering both the thrust and torque reaction of that rotor and hence manoeuvring the aircraft.

The Ascending Technologies (AscTec) Pelican Quadrotor is one of many successful quadrotors on the market. The Pelican differs from other available models (such as the Parrot A.R. Drone) as it is designed specifically for research purposes.

The AscTec Pelican has a CoreExpress Intel Atom Processor (referred to as an AtomBoard processor) which uses a Linux Operating System. It is mounted on a carbon fibre plate underneath the AutoPilot Board in the tower design. It is a powerful on-board computer capable of carrying out many complex tasks whilst maintaining a small and light form factor. In the supplied form, the AscTec Pelican is equipped with an inertial sensor suite, a GPS receiver, and a barometric pressure sensor.

Due to the tower design, many different payloads can be mounted, weighing up to 650 g. The maximum payload allows a flight time of 15 minutes. There are two allocated areas for attaching a payload (sensors), they are either on the top angled out towards the horizon, or at the base of the platform, angled downward. The bottom attachment is actively stabilised using a gyroscope [31].

III. INTEGRATION

![Diagram of Pelican's on-board systems integration](image)

Figure 6. Integration of the Pelican's on-board systems to the CamBoard nano

5 of 10
Final Thesis Report 2013, UNSW@ADFA
A. Robot Operating System

The ROS was chosen to control the interaction between the Pelican’s on-board sensors, the CamBoard nano and the flight control systems of the Pelican. The ROS contains the many drivers necessary to provide the link between hardware and the software able to control it. There are also applications for simulation and the analysis of inter-process communication.

The ROS has a large research community with compatibility on a range of platforms and a strong focus on integration and documentation. A package for the CamBoard nano exists which allows for its integration into the ROS framework. The major functions required for testing and fusing sensors are delivered by ROS such as logging, sensor control, feature recognition and mapping.

B. Mounting

The Pelican required modification in order to mount and power the CamBoard nano. The CamBoard nano was mounted on the bottom facing actively stabilised gyroscope always pointing towards the ground, thus making the height detection accurate and the pitch and roll calculations not necessary. When the Pelican is in the initial ground position, the CamBoard nano is approximately 3 cm above the ground.

On some hardware, including the AtomBoard, USB devices are not correctly powered through a USB port. Due to limited power provided, external power, such as an actively powered USB Hub, is sometimes required to ensure proper operation. This was found to be the case when mounting the CamBoard nano to the base of the Pelican. Due to the nature of the Pelican, a permanent solution over an ad hoc one was desired. The final solution was to implement a small 5 V isolated DC-DC converter using the Pelican battery. Unfortunately this solution doubles the weight of the CamBoard nano, thus reducing the benefit of such a lightweight sensor.

C. Communication

A ground station to run commands on the Pelican was configured through a Secure Shell (SSH) connection over Wi-Fi. This allows on-board programs to be run without any peripherals attached directly to the Pelican, thus tests can be conducted whilst the Pelican is flying.

An X-Bee is required to access and control the AutoPilot control board on the AscTec Pelican. The X-Bee is a USB to COM Port connection that allows for a direct RF connection to the AscTec Pelican, but is only available on Windows OS. The control software can be used to manually position the gyroscope to the desired downwards angle for the CamBoard nano.

D. Limitations

The best feature of the CamBoard nano is its small size and weight, however due to this, the CamBoard nano does not have a cooling fan or housing. Consequently, the CamBoard nano can heat up significantly after continuous updates. The sensor heats to approximately 45°C, which is hot to touch, but cool when considering semiconductor operating temperatures.

Semiconductors dependence on temperature causes the system to appear inconsistent whilst the temperature is stabilising. Controlled stabilisation can be achieved with a warm up period of approximately one minute of dummy updates before accurate measurements are gathered. Studies have been carried out on the effect temperature can have on ToF cameras [15]. In order to effectively manage these inaccuracies, the camera must operate during take-off procedures before any control can be implemented.

During camera calibration, a trade-off was made for close range accuracy to achieve a larger range. This was found to be a suitable trade-off as the accuracy of the camera is still within 2 cm over the entire working range (10 cm to 150 cm). As such, the maximum range was increased from 50 cm to 150 cm, and the minimum range was increased from 5 cm to 10 cm in an indoor environment.

In order for the CamBoard nano to calculate the distance to an object, the light is emitted by the IR LED and travels in a straight path to the surface of the object, and is reflected directly back. The incident object’s surface greatly affects the intensity of the reflected signal.

1) Surfaces such as carpet will scatter and absorb light which will cause low signal intensity at the PMD receiver optic, throwing a low signal flag, and variable phase delay causing inaccurate depth readings.

2) If it is a specular (mirror-like/shiny) surface it will completely reflect the light according to the law of reflection away from the PMD receiving optic, which will result in either no or very limited received light.
3) If the ground is a cluttered environment with highly reflective surfaces, they could reflect light to another object, causing multiple reflections. The reflected light directly from the emitter will superimpose with the indirectly reflected light to cause an incorrect depth value, which cannot be correctly determined at all.

4) Very fast moving objects can cause motion blur at the edges, which will result in inconsistent depth values as shown in Figure 8.

The above mentioned limitations associated with the surface parameters introduce operational constraints when flying the quadrotor. In order to allow for an adaptive integration time using a look-up table method, the surface parameters should remain constant throughout the entire flight. Surfaces such as glass, polished metal, and carpet should be avoided. Flight manoeuvres should also be limited in order to reduce the motion blur inconsistencies.

IV. VERIFICATION

A. Algorithm

In order to accurately map and control the altitude of the quadrotor, the instantaneous height must be accurately measured. There are a number of methods to compute the height using the CamBoard nano, these must be evaluated and used to verify the operational conditions of the CamBoard nano. There are two functions in the SDK to find the position of the CamBoard nano above the surface – pmdGetDistances and pmdGet3DCoordinates (functions described in Section II.A.i). A simple measurement taking the distance of the centre pixel could be used to determine the height but this is unreliable due to the possibility that the pixel is flagged as invalid or it is in a cluttered environment.

The method used to accurately model the terrain is the Least Squares ground plane detection method outlined in Section II.A.ii.a. Before Least Squares approximation was implemented, the invalid pixels are filtered using the SDK command pmdGetFlags outlined in Section II.A.i. In order to reduce the interference from nearby objects, the FoV was reduced by applying margins as depicted in Figure 10.

Several iterations were used to ameliorate the fitting of the plane by discounting points that don’t lie within a small exponentially decaying tolerance of the previously calculated plane. This guarantees that if the image is in a slightly cluttered environment it will discount those objects when determining the height. If the object occupies most of the field of view it would dominate the Least Squares determination of the plane.

Algorithm:
for each pixel
store image
check if in bounds
check flags
build A(initial) matrix
for each A(initial) coordinate
create B matrix (A^T A)
calculate Lambda
for each iteration
for each A(initial) coordinate
apply tolerance to calculated ground plane
output A
create B matrix (A^T A)
calculate Lambda

Figure 9. Ground plane determination algorithm utilising the Least Squares method.

Figure 8. Inconsistent flags (white) attributed to motion blur.

Figure 10. FoV of the final image used for ground plane determination. A margin was applied to each edge changing the effective FoV.
B. Static Testing

The ground plane fitting algorithm was run using the ground station over Wi-Fi. At each height, five measurements were made for 11 integration times – $12 \mu s$, $100 \mu s$, $200 \mu s$, $333 \mu s$, $500 \mu s$, $700 \mu s$, $1000 \mu s$, $1200 \mu s$, $1500 \mu s$, $1700 \mu s$, and $2000 \mu s$. For post analysis, the full 3D coordinates for each frame were recorded to on-board storage. As part of the ground plane fitting algorithm, the ground plane is fitted over four iterations to improve the mapping to the plane and to reduce the effect of outliers. The coordinates of the pixels used for the calculation of the ground plane are also recorded to on-board storage. The processing speed when conducting recording is significantly reduced to only 1 fps on average instead of 10.05 fps when only lambda (ground plane coefficients) is recorded.

The variance, including the warm up period, is very low and excluding the warm up period it is only 0.14 cm. This, coupled with a very accurate height measurement, verifies that the measurements from a properly initialised CamBoard nano are reliable. However, as shown in the plot, when a simple low pass filter is applied the noise is reduced, dropping the variance to only 0.005 cm after the warm up. This test was performed on an Ubuntu desktop, achieving 31.9 fps.

To validate the quality of the height determined by the ground fitting algorithm on the CamBoard nano was conducted on different surfaces. The Pelican was suspended above an obstacle free zone using rope at 19 heights to investigate the effects integration time has on the height reading.

The real height was unable to be accurately measured to the edge of the CamBoard nano’s lens using the apparatus available - a flexible tape measure. However, considering the instantaneous height will be used to determine instantaneous speed for speed control, accuracy to within a centimetre will be sufficient.

i. Indoor Testing

The accuracy over the entire range for any singular integration time was not acceptable. An example is shown in Figure 13.R where the integration time was set to $1200 \mu s$ and showed valid heights over the accepted range. Ignoring any major outliers (measurements over 5 m), the mean distribution of absolute error is 2.26 cm with a standard deviation of 4.7 cm, however, this error is mostly attributed to the measurements at 10 cm. This indicates that the height measurements are accurate however could cause issues in a control system as the variation is greater than 2 cm.

If the working range is adjusted to minimise the error for each integration time, the accuracy improves greatly. For instance, the $1200 \mu s$ mean absolute error and standard deviation improves to 1.17 cm and 0.81 cm, respectively when the minimum working range is increased to 20 cm.

It is clear that multiple integration times are necessary to ensure accurate readings across the entire working range. Hence the idea of an adaptive integration time has been considered in order to optimise the range over different surfaces. Figure 13.L shows a MATLAB analysis of the best result possible for the textured linoleum surface. Each point on the best combined model
(green) is made up of a potential different integration time. It was found that the integration time that modelled the real height estimation most often was 700 µs. The overall mean absolute error and standard deviation was 0.56 cm and 0.58 cm respectively over the working range.

C. Real-time Testing

![Graph showing static test results](image)

**Figure 13.** Indoor static test conducted on a textured linoleum surface to measure the accuracy of the implemented altitude calculation algorithm.

L) This plot determine which integration time is most accurate at each height (3 cm – 200 cm). It shows the height measured using the integration time that produces the least error for each run. The median integration time was given to indicate which integration time was most accurate most of the time. There is poor accuracy when the Pelican is on the ground and improves at 10 cm. This test confirms the notion that the integration time is required to vary as the height changes. This also shows that operating bounds of the system is approximately 10 cm – 150 cm (accepted range). However, it is important to note that this plot could not be replicated in real time and the accuracy at height below 15 cm is inconsistent requiring further investigation.

R) This plot represents a single fixed integration time of 1200 µs. The working range depicted for this integration time is 20 cm to 150 cm with the mean distribution of absolute error of 1.17 cm and standard deviation of 0.81 cm.

Following static testing, a confirmatory test was conducted to ensure the CamBoard nano could return an instantaneous height while operating effectively in the desired environment. The Pelican was flown over the same textured surface as the static testing phase, whilst operated in manual⁶ mode for the duration of the test. The Pelican was flown to simulate the flight profile used in the eventual take-off, landing, and altitude control using the ground plane algorithm.

![Graph showing real-time flight test results](image)

**Figure 14.** Instantaneous altitude of the Pelican during remote controlled flight at a fixed integration time of 700 µs.

The restriction of minimum detectable height is apparent during take-off and landing. The first valid height on take-off is 18.5 cm and the final valid height on landing was 11 cm. A greater range is required to allow for autonomous take-off and landing. This could be done through an adaptive integration time or increase the mount height of the camera above the ground, which is preferable however would require redesigning the undercarriage of the Pelican. Another potential solution is a hard coded landing and take-off procedure once the minimum height is reached – if the height decreases, then suddenly spikes; it can be assumed the aircraft is low enough to conduct an auto descent. This could be successful, but could cause the Pelican to attempt an unwanted auto landing, potentially causing damage.

This preliminary real time testing operates at an average 10.05 fps. The verification of the determined height could not be completed due to the absence of a sonar sensor to compare the results to. For this test, there were no other sensors fused with the ground plane determination in order to improve the quality of the determined height.

---

⁶ In manual mode, the Pelican is controlled directly from the pilots input.
V. RECOMMENDATIONS

Due to the low processing speed of the Atom Processor, the frame rate did not meet the original expectations, therefore it was recommended and approved to be upgraded to a more powerful processor such as the AscTec MasterMind operating an Intel Core i7. The only compromise that must be made for this improvement is increased power consumption leading to reduced battery life. An experiment was done on a 64-bit Intel Quad-Core i3 CPU at 2.93 GHz x 4. This experiment concluded that the additional processing power will increase the frame rate by up to two and a half times. However, the frame rate has not been tested utilising the ROS package.

The problem with mounting the CamBoard nano to the Pelican detailed in Section III.B can be revisited to install a smaller and lighter DC-DC converter.

The AtomBoard processor used on the Pelican can be configured to communicate with the CamBoard nano in ROS to run the algorithm in real-time. The velocity and acceleration from the instantaneous height can then be calculated and smoothed by fusing with the on-board inertial sensors. Once the measurements are working in real time, the Pelican can then be controlled to hover in the z-plane using a PD controller.

ROS implementation on other robot platforms the school of engineering currently possess, such as the Parrot AR Drone and the Pioneer 3-AT. ROS provides a platform for greater interoperability of sensors and real-time control utilising open source detection and mapping algorithms.

Recently, Leap Motion released another version of a small ToF camera similar to the Microsoft Kinect used for gesture control [32]. It is far cheaper than the CamBoard nano whilst still providing a fully featured SDK with appropriate gesture control.

VI. CONCLUSION

The pmd[vision]® CamBoard nano is the most compact Time-of-flight image sensor currently on the market. Its primary function is for close range gesture control, and is optimised as such. Consequently, the application of this project using a CamBoard nano is limited to short ranges, experimentally between 10 cm and 150 cm. This however is considered to be ideal for autonomous take-off, landing and hover for a Micro-unmanned Aerial Vehicles as discussed in this summary report.
References


Ref. 1

Final Thesis Report 2013, UNSW@ADFA


