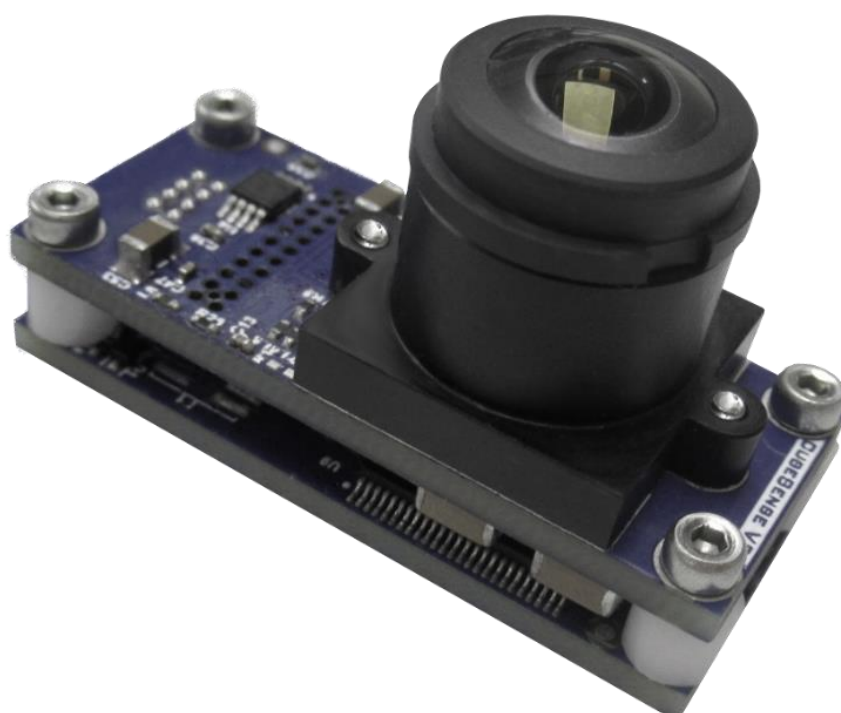




# CUBESENSE V3

AN INTEGRATED SUN AND NADIR SENSOR MODULE



## USER MANUAL

### Contact Us

Phone (0027) 79 945 9957  
E-mail [info@cubespace.co.za](mailto:info@cubespace.co.za)  
Web [www.cubespace.co.za](http://www.cubespace.co.za)  
Facebook /CubeSpaceADCS  
Twitter @CubeSpace\_ADCS


### Physical Address

**CubeSpace**  
Hammanhand Road  
Stellenbosch  
7600  
South Africa

# Table of Contents

<b>Document Approved By .....</b>	<b>2</b>
<b>List of Acronyms/Abbreviations .....</b>	<b>2</b>
<b>1. Introduction.....</b>	<b>3</b>
<b>2. Getting Started .....</b>	<b>4</b>
2.1 Unpacking the CubeSense package .....	4
2.2 Before getting started.....	4
2.3 Ground Support Program (GSP).....	4
2.4 Hardware setup .....	6
<b>3. Usage .....</b>	<b>8</b>
3.1 Identification.....	8
3.2 Managing memory contents.....	9
3.3 Doing detection.....	10
3.4 Interpreting detection result .....	12
3.5 Capturing and downloading images.....	13
3.6 Typical use.....	15
<b>4. Important Usage Considerations .....</b>	<b>17</b>
4.1 Detection thresholding.....	17
4.2 Image exposure settings.....	17
4.3 SRAM over-current protection .....	18
4.4 Nadir Detection Adjustment.....	18
4.5 Sensor masking .....	18
<b>5. Documentation History .....</b>	<b>21</b>

## Document Approved By

Document Approved By	
Document	CubeSense V3 User Manual
Version	1.11
Date Modified	6 January 2020
Approved By:	Douw Steyn
Signature	

## List of Acronyms/Abbreviations

CMOS	Complementary metal-oxide semiconductor
ADCS	Attitude and Determination Control System
FPGA	Field Programmable Gate Array
I <sup>2</sup> C	Inter- Integrated Circuit
MCU	Microcontroller Unit
OBC	Onboard Computer
SRAM	Static Random Access Memory
UART	Universal Asynchronous Receiver/Transmitter
COTS	Commercially Off-the-shelf
PCB	Printed circuit board

# 1. Introduction

The CubeSense module is an integrated sun or nadir sensor for CubeSat attitude sensing. It makes use of a CMOS camera that is configured for either sun sensing or for horizon detection. If configured as a sun sensor, a neutral density filter is included in the optics to ensure that only the sun will be visible in the image. The camera has wide field-of-view optics (200 degrees) for increased operating range.

The primary output of the sensor is the measured sun/nadir vector in the sensor's coordinate frame. The measured vectors are output as angles relative to the camera bore-sight. If the CubeSense is configured as a nadir sensor then it can also be used as a camera to capture and download 1024x1024 pixel greyscale images.

This document is only applicable to CubeSense V3.



The unit contains a variety of static sensitive devices. The appropriate electrostatic protection measures must thus be implemented. **The unit must never be handled without proper grounding.**



It is recommended that the unit be handled in a clean environment. A clean room of ISO class 8 or better or an appropriate laminar flow workbench is recommended.



The unit should be **kept free of moisture or liquids**. Liquids and moisture could have corrosive effects on the electronics and electronic joints which may lead to degradation and loss of reliability of the circuits.



The unit must be handled with care and **dropping or bumping the unit should be completely avoided.**



The camera **lens should be kept clean** and free of any dirt that may obstruct the images captured by the camera. Dust should be removed with a cloth. If required, the lens may be cleaned using ethanol and appropriate lens cleaning equipment, but unnecessary cleaning of the lens should be avoided.



The optics are fitted with a **dust cap which should be removed before flight.**



The position of the lens relative to the image sensor is of extreme importance for accurate detection. **Any external force on the lens or lens holder should be completely avoided.**



**Please read Section 4(Important Usage Considerations) very carefully.**

## 2. Getting Started

The Getting Started guide will show the simple steps to get CubeSense up and running. CubeSense is provided with a simple test application to allow the user to gain experience with the available functions as well as test the hardware.

### 2.1 Unpacking the CubeSense package

The received Peli-Case contains the following items:

- CubeSense unit(s)
- Support PCB
- UART-to-USB cable
- CubeSpace flash drive

The included items will allow the user to interface with a CubeSense unit without the need for other hardware.

### 2.2 Before getting started

The following additional items are required before the user can get started with the CubeSense:

- Multi-meter or oscilloscope
- Ground Support Program (see Section 2.3)
- Computer with an open USB port (running Windows 7 or later)

### 2.3 Ground Support Program (GSP)

#### 2.3.1 What is the GSP?

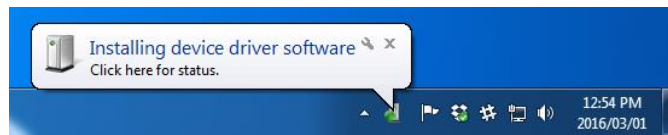
The GSP allows the user to interface with a CubeSense via the UART connection. No additional software or hardware (except the items mentioned in Section 2.2) is required, which means that the CubeSense can act as a standalone module.

The software enables the user to request telemetry and to send telecommands from/to a CubeSense.

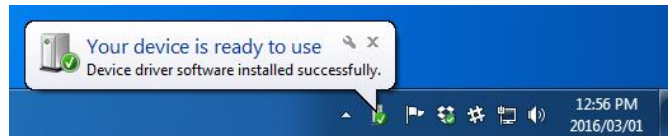
#### 2.3.2 Connecting to the UART cable

Follow the instructions below to verify the connection between CubeSupport and the UART cable:

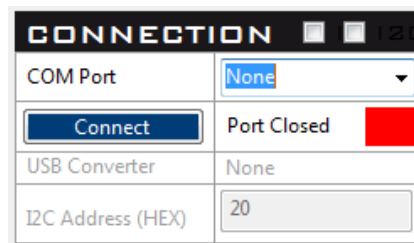
- Plug the UART-to-USB cable into the computer's open USB port.
- The computer should detect the cable and install the drivers by itself.



- After a couple of minutes, a message indicating the successful installation of the drivers should appear.

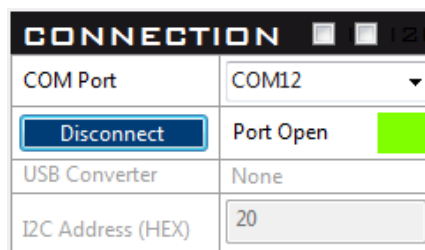


- Browse to the folder containing the GSP and launch the application by running *CubeSenseV3\_GSP\_TestingApplication.exe*.
- Select the COM port of the UART cable and click on *Connect*



Check the COM port assigned to the cable by browsing to *Device Manager* and noting the number shown under *Ports (COM & LPT)* for *USB Serial Port (COM x)*. The number given by **x** will be used to connect to CubeSense.

- The connection status should turn green if the connection was **successful**, as illustrated below.

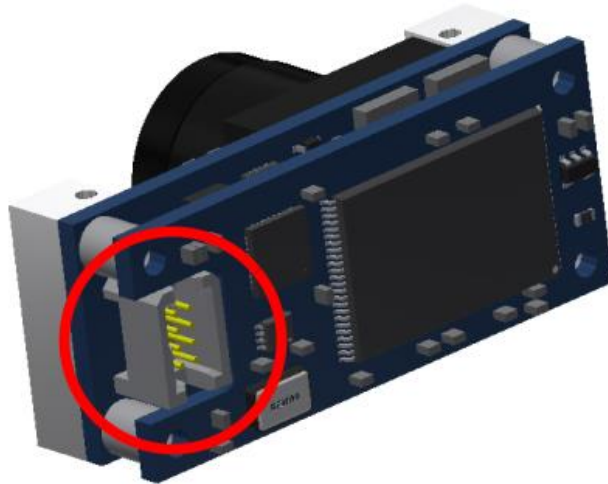


- Disconnect from the UART-to-USB cable by clicking on the *Disconnect* button.

The Health Check document will allow the user to become familiarized with the GSP interface as well as perform the incoming health check to ensure that the module arrives in full working condition.

## 2.4 Hardware setup

Electrical  
Interface



**Figure 1 – CubeSense electrical connections**

### 2.4.1 Power

The CubeSense module draws its power from the USB port if the Support PCB is used. If the module is not used with the Support PCB, the power could also be provided by a stable lab power supply with a 3.3V output. The current limit of the supply should be set to approximately 300mA. Although the CubeSense circuitry has some protection, it is still good practice to have this current limit. The standard connector populated on the CubeSense module is a *Samtec TFC-104-01-F-D*. The 3V3 power is located at pin 1 as shown in Table 1.

**Table 1 – Pinout of CubeSense Connector\***

Pin	Name	Description
1	3.3V	Input voltage of 3.3V
2	SDA <sup>1</sup>	SDA line for I2C communication
3	SCL <sup>1</sup>	SCL line for I2C communication
4	GND	Ground pin (both ground pins must be used)
5	RX	RX line for UART communication
6	TX	TX line for UART communication
7	EN	Enable line that turns the CubeSense on or off (active high)
8	GND	Ground pin (both ground pins must be used)

<sup>1</sup> There are no pull-up resistors on the I2C bus. Any pull-up resistor should be implemented by the master.

### 2.4.2 Communication

There are multiple options for communicating with CubeSense. The primary communication method for ground testing is UART. The location of the UART TX and RX pins are shown in Table 1.

For the ground tests:

- Plug the CubeSense module into the support PCB (header is marked as CubeStar)
- Plug the supplied UART-to-USB cable into the support PCB (the header is labelled as "UART" on the PCB). Plug the UART-to-USB cable into the computer (if it is not plugged in already).
- Open the GSP (if it is not open already).
- Connect to the appropriate COM port
- A successful connection should allow you to request the status and observe the Runtime increasing as shown here:

STATUS		READ
Runtime	7s460ms	
Firmware Version	V3.0	
Serial Number	CS1923	
TCMD Counter	0	
TLM Counter	3	
TCMD Buffer Overrun	<input type="checkbox"/>	
I2C TLM Read Error	<input type="checkbox"/>	
Last TCMD ID	0	
TCMD Processed	<input checked="" type="checkbox"/>	
TCMD Error	No Error	
Reset Communications		
Reset MCU	Reset Cameras	

- Refer to the Health Check document for further tests that can be performed.
- If the connection is not successful, check the following:
  - ✓ Verify that the CubeSense is plugged into the support PCB.
  - ✓ Verify that the UART connection cables are plugged in correctly.

In case of continuous connection issues, probe pin 1 (marked with line) of the CubeSense/CubeStar header on the support PCB and verify that there are 3.3V present.

### 2.4.3 Enabling/Disabling

CubeSense has its own overcurrent protected switches which switch on/off the power that is supplied to the board. At production, these switches can be configured to be default on or off. If the support PCB is used the Enable line will be always on when power is applied. When CubeSense is used without the CubeSupport board, the user can switch on/off CubeSense by applying 3.3V to the CubeSense Enable pin. The Enable pin is active high. The location of this pin is shown in Table 1.



## 3. Usage

### 3.1 Identification

Each CubeSense module is programmed with a serial number, and details about its firmware. To check these details about a CubeSense module, TLM no 0 and 1 can be requested. The TLMs starts at address 128, so these addresses will be 128 and 129. The content of these TLM frames are as follows:

**Table 2 – Status TLM**

<b>Telemetry frame ID</b>	0				
<b>Name</b>	Status				
<b>Frame length (bytes)</b>	8				
<b>Channels</b>	<b>Byte No</b>	<b>Length (bytes)</b>	<b>Channel</b>	<b>Data type</b>	<b>Detail</b>
	0	1	Node type	Unsigned 8-bit	Identification of type of CubeComponent Node
	1	1	Interface version	Unsigned 8-bit	Interface definition version
	2	1	Firmware version (major)	Unsigned 8-bit	
	3	1	Firmware version (minor)	Unsigned 8-bit	
	4	2	Runtime (seconds)	Unsigned 16-bit	Number of seconds since processor start-up
	6	2	Runtime (milliseconds)	Unsigned 16-bit	Number of milliseconds (after the integer second) since processor start-up

**Byte 0:** The node identification is used within CubeSpace ADCS bundles to identify which modules are connected. CubeSense has Node Type = 2

**Byte 1:** The interface definition indicates which version of the software interface CubeSense implements. This should match the 'Node Definition' version number that is being used.

**Byte 2 & 3:** Byte 2 is the major version and Byte 3 is the minor version of the CubeSense module. If Byte 2 = 1 and Byte 3 = 0, the firmware version is V1.0

**Byte 4-5:** Bytes 4-5 forms an unsigned 16-bit integer that indicates the elapsed time since start-up, in seconds.

**Byte 6-7:** Bytes 6-7 forms an unsigned 16-bit integer that indicates the elapsed time since the previous second, in milliseconds.

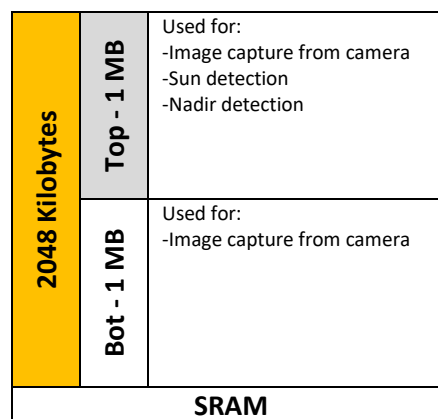
**Table 3 – Serial number TLM**

Telemetry frame ID	1				
Name	Serial number				
Frame length (bytes)	2				
Channels	Byte No	Length (bytes)	Channel	Data type	Detail
	0	2	Serial number	Unsigned 16-bit	Number that defines the serial of CubeSense

**Byte 0-5:** Each CubeSense has a unique serial number.

### 3.2 Managing memory contents

The CubeSense memory module is divided into two halves, top and bottom. This is illustrated in Figure 2.



**Figure 2 – SRAM layout**

Each 1 MB (1048576-byte) half can store one CubeSense image. The main functions CubeSense can execute are:

1. Capture an image to one of the halves of the SRAM
2. Perform detection on the image stored in the **top** half the SRAM
3. Download the image stored in one of the halves of the SRAM.

Only one of these functions can be executed at any given time. If the command is given to execute multiple functions, they will be executed sequentially. It is important to note which part of which SRAM will be influenced by a given instruction, since this will overwrite the

contents that were stored in that position. It is the user's responsibility to choose the correct part of the SRAM to write to and to avoid overwriting data that should be stored for use at a later stage.

### 3.3 Doing detection

Even though CubeSense has the capability to capture and download images, the main application of the module is to do Sun or Nadir detection. While doing detection the user is not required to download or manage images. When a detection command is received by CubeSense, the module automatically captures a new image as soon as possible, executes the applicable type of detection on that image, and saves the result in memory. There are two ways of issuing a detection command:

1. **Send TC 20 for a Capture and Detect:**

TC 20 should be sent without any parameters

2. **Request TLM 22 for the last detection result, and automatically trigger new capture and detection:**

When requesting the latest detection result through these TLMs, a new detection is automatically scheduled. This is the method of operating CubeSense that requires the least communications, and is the recommended method of operating the module.

The result of a detection can be read by either reading the result of the previous detection that was done (TLM 20) or reading the result while also scheduling a new detection (TLM 22) as discussed above. All of these telemetries have the following structure:

**Table 4 – Auto-trigger TLMs**

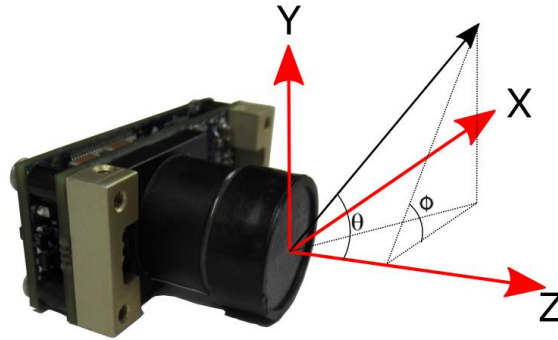
Telemetry frame ID	20 and 22				
Name	Detection result & Trigger				
Frame length (bytes)	6				
Channels	Byte No	Length (bytes)	Channel	Data type	Detail
	0	2	$\alpha$	Signed 16-bit	$\alpha$ angle in centi-degrees (range = -100 to 100 degrees)
	2	2	$\beta$	Signed 16-bit	$\beta$ angle in centi-degrees (range = -100 to 100 degrees)
	4	1	Capture Result	Unsigned 8-bit	0 = start-up 1 = capture pending 2 = successfully captured 4 = camera timeout

					5 = SRAM overcurrent
	5	1	Detection result	Unsigned 8-bit	0 = start-up 1 = no detection scheduled 2 = detection pending 3 = Nadir error – too many detected edges 4 = Nadir error – not enough detected edges 5 = Nadir error – Bad fit 6 = Sun error – Sun not found 7 = Successful detection

The first step after reading this TLM is to check the detection and capture-results. The description of each result is detailed in the table above. ***Only when the capture result is "Successfully captured" and the detection result is "Successful detection" should the result in bytes 0-3 be used as valid detection results.***

### 3.4 Interpreting detection result

The sensor's axes are defined as follows:



**Figure 3 – Camera axes definitions**

The measured angles  $(\alpha, \beta)$  that the sensor output are virtual angles that can be transformed to angles or vectors that can be used in the satellite attitude determination. The direction vector can be calculated using the following procedure:

First calculate  $\theta$  and  $\phi$  using the following formula:

$$\theta = \sqrt{\left(\frac{\alpha}{100}\right)^2 + \left(\frac{\beta}{100}\right)^2}$$

$$\phi = \text{atan4}(\beta, \alpha)$$

Then, the direction vector is:

$$\begin{pmatrix} X \\ Y \\ Z \end{pmatrix} = \begin{pmatrix} \sin(\theta) \cos(\phi) \\ \sin(\theta) \sin(\phi) \\ \cos(\theta) \end{pmatrix}$$

In finding the  $\alpha$  and  $\beta$  detection values, the sensor makes use of a calibrated  $\alpha/\beta$  "bore-sight" location on the imaging sensor. This pixel location is the intersection of the optical axis of the lens with the imaging sensor. This value is unique for every sensor and can be changed using a "Set boresight" telecommand. After a reset, the boresight value reverts to the production default.



**This boresight value is programmed into the firmware of CubeSense at time of production and is calibrated to each individual lens. It is advised to not change this value.**

### 3.5 Capturing and downloading images

CubeSense provides the option of downloading an image that is located in any of the SRAM locations. This can be done for images of the earth or of space, or to verify the exposure settings of the sensor to tweak its performance. Images can be captured to the top or bottom half of the SRAM. The module can therefore hold 2 images at any given time. It is however important to note that doing detection will overwrite the top image in the SRAM.

TC 21 is used to capture an image to a specified SRAM location:

**Table 5 – Capture Image TC**

<b>Telecommand ID</b>	21				
<b>Name</b>	Image capture				
<b>Parameters length (bytes)</b>	1				
Fields	Offset	Length	Field	Data type	Detail
	0	1	SRAM Location selection	Unsigned 8-bit	0 = Top Half 1 = Bottom Half

After sending the command to capture the image, TLM 20 can be requested to see if the image was successfully captured. Byte 4 ("Capture Result") in this TLM can be used to read the status of the capture.

Once the required image is captured, it will be stored as long as the CubeSense module is powered, or it is overwritten by another image.

Downloading an image from CubeSense to an OBC or PC is a lengthy process (a few minutes on highest resolution). It is important to note that while CubeSense is busy downloading an image, no other image captures or detection can be done. Once another command is sent to CubeSense, its electronics will reset into a new mode, and the progress of the download will be lost. In such a case the download will have to be restarted. Because of this reason, it's advised to hold all downloads for eclipse, when no detection is possible.

A full resolution CubeSense image is 1024x1024 bytes. Smaller versions of this image can however also be downloaded in cases where limited time is available for communication. To set up a new image download, TC 64 is called:

**Table 6 – Initialize image download TC**

<b>Telecommand ID</b>	64
<b>Name</b>	Initialize image download
<b>Parameters length (bytes)</b>	2

Fields	Offset	Length	Field	Data type	Detail
	0	1	SRAM location	Unsigned 8-bit	0 = Top 1 = Bot
	1	1	Size selection	Unsigned 8-bit	0 = 1024x1024 (8192 frames) 1 = 512x512 (2048 frames) 2 = 256x256 (512 frames) 3 = 128 x 128 (128 frames) 4 = 64 x 64 (32 frames)

Images are downloaded in 128-byte packets. When TC 64 is received by CubeSense, the first 128 bytes of the image is loaded into CubeSense's memory. Once this TC has been sent to CubeSense, TLM 65 can be polled to check when the image frame has successfully been loaded.

**Table 7 – Image frame info TLM**

<b>Telemetry frame ID</b>	65				
<b>Name</b>	Image frame info				
<b>Frame length (bytes)</b>	3				
<b>Channels</b>	<b>Offset</b>	<b>Length</b>	<b>Channel</b>	<b>Data type</b>	<b>Detail</b>
	0	2	Image frame number	Unsigned 16-bit	Number of current frame loaded into download buffer
	2	1	checksum	Unsigned 8-bit	XOR checksum of frame loaded into download buffer

Once the frame number is set to 0, the image frame can be requested. To do this, TLM 64 can be requested. If required, the integrity of the image frame can be verified by a XOR (Exclusively-OR) of all of the bytes in the frame. This final value can be compared to Byte 2 of TLM 65 that was read earlier. If the values do not match, the frame can be downloaded again by requesting TLM 65 again.

Once the frame has been successfully downloaded, the next frame can be loaded into CubeSense. This is done by sending TC 65. **It is important to note that CubeSense expects the next frame to be one more than the current frame.** For example, if the current frame is 10, the next parameter sent with TC 65 should be 11. If this is not correct, CubeSense will simply ignore the TC.

**Table 8 – Advance Image download TC**

<b>Telecommand ID</b>	65				
<b>Name</b>	Advance image download				
<b>Parameters length (bytes)</b>	2				
<b>Fields</b>	<b>Offset</b>	<b>Length</b>	<b>Field</b>	<b>Data type</b>	<b>Detail</b>
	0	2	Next frame number	Unsigned 16-bit	Number of next frame to be loaded

After this TC has been sent, TLM 65 can once more be polled until the 'frame number'-channel is equal to the appropriate image frame. These steps can be repeated until all frames have been downloaded.

The data for an image with dimensions X by X can be packed into a bitmap in the following manner:

Byte 1	Byte 2	Byte 3	.	128	.	X-2	X-1	X
X + 1	.	.	.	.	.	.	.	.
2X + 1	.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.	X*X

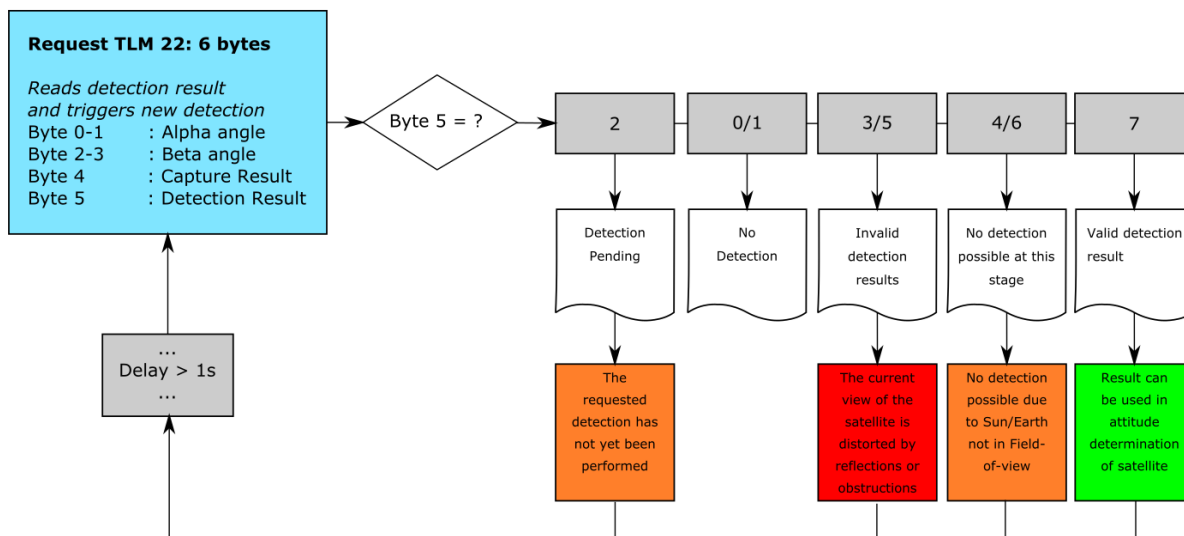
**Figure 4 – Image bitmap example**

### 3.6 Typical use

The way in which CubeSense is normally used is to periodically request the result of the last detection, and to trigger the next detection.

The procedure for doing detection is outlined in Figure 5.





**Figure 5 – Flow diagram for typical use**

## 4. Important Usage Considerations

### 4.1 Detection thresholding

Sun/Nadir-sensors need to differentiate the sun/earth from the background of space in the images that they take. In both cases this is done by comparing the brightness of the pixels in the image to each other, in order to locate the sun and the edges of the earth. These algorithms require a threshold and sensitivity value to execute. By setting the threshold high enough, unwanted artefacts in the image taken by the camera are discarded, and only the relevant celestial body is identified.

The threshold is set to a default value optimized for robust measurements. If the user wishes to change the threshold values, telecommand 40 can be used. It is however recommended that the default values be used.

### 4.2 Image exposure settings

The exposure times for the image sensors are key to their successful operation. The default exposure settings for the sensors are set to values for robust operation and it is recommended that the client do not alter these values. If, however the client wishes to use the Nadir sensor as a greyscale imager, the exposure can be set by using telecommand 43. For correct detection, the exposure should be set to acquire images similar to Figure 6.

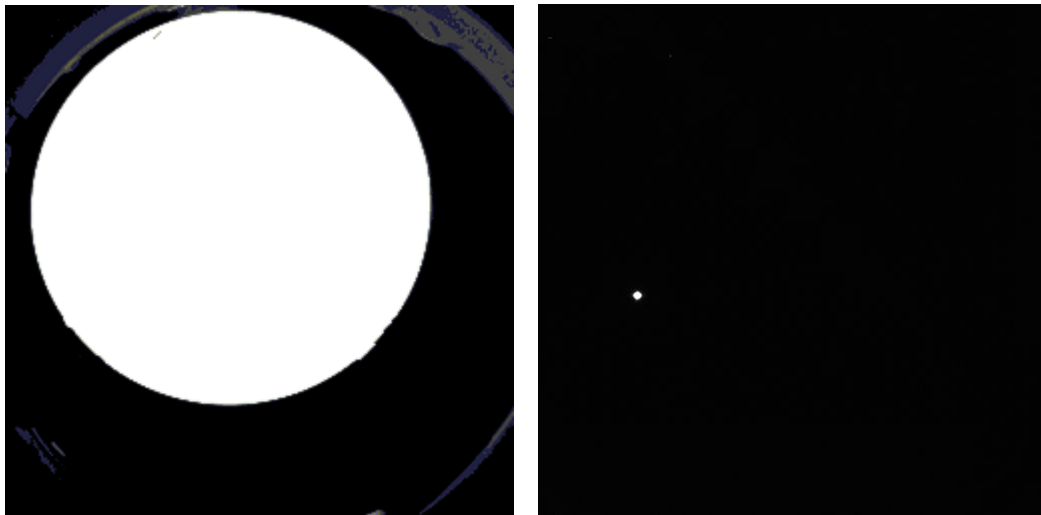


Figure 6 – Images of earth (left) and sun (right) from orbit

### 4.3 SRAM over-current protection

When high energy radiation causes a latch-up inside the SRAM, it will result in a high current draw that may permanently damage the SRAM. CubeSense is fitted with a current sensor to monitor the SRAM. In the case that the SRAM experiences an over-current, power to the SRAM is turned off and an internal status flag is set in the CubeSense. Subsequent usage of the sensor will result in an error response indicating an SRAM over-current. The "Power"-telemetry frame (ID = 26) can be used to read the SRAM current or to check for SRAM over-current. After an over-current has occurred, the user can clear the over-current flag, to re-enable the use of the SRAM by using the "Clear SRAM overcurrent flag" Telecommand (ID = 11). In the case of permanent SRAM failure, CubeSense will no longer be able to provide detections or image downloads.

### 4.4 Nadir Detection Adjustment

Using a false Nadir detection in the satellite EKF can cause instability. The Nadir algorithm of CubeSense utilizes simple logic to reject such false detections and displaying a Bad Fit error. There are four adjustable parameters that can be used to reject false nadir detections – the *Max Deviation Percentage*, *Max Bad Edges*, *Max Radius* and *Min Radius* parameters.

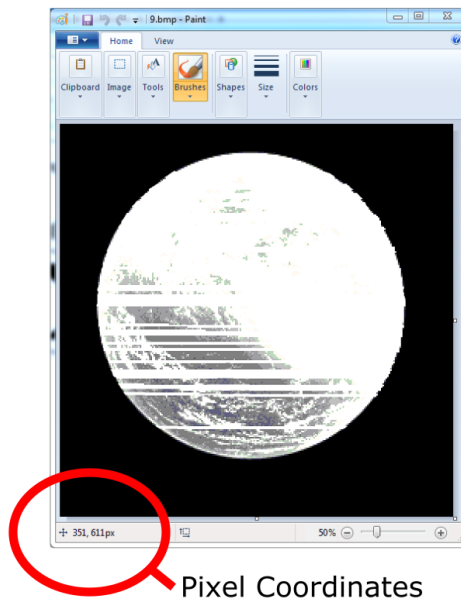
The Nadir sensor algorithm attempts to fit a circle to a number of detected edge points. The *Min-* and *Max Radius* parameters (telecommand 15) can be used to specify the valid range of the circle radius. This will depend on the height of the satellite in its orbit. The *Max Deviation Percentage* (telecommand 14) is the tolerance that is allowed for all detected points to lie on or close to the fitted circle. If there are more than *Max Bad Edges* such points, a detection error is reported. Do not adjust any of these parameters without first consulting CubeSpace.

### 4.5 Sensor masking

The Nadir sensor may in some configurations have antennas or other deployable structures in its field of view. For these cases, a mask on these images is required to avoid false detections. The sun sensor's detection algorithm does not require any masking to function, but in cases where the object in its FOV is reflective, it may provide false detections. A mask should then be placed over the reflective area to avoid this.

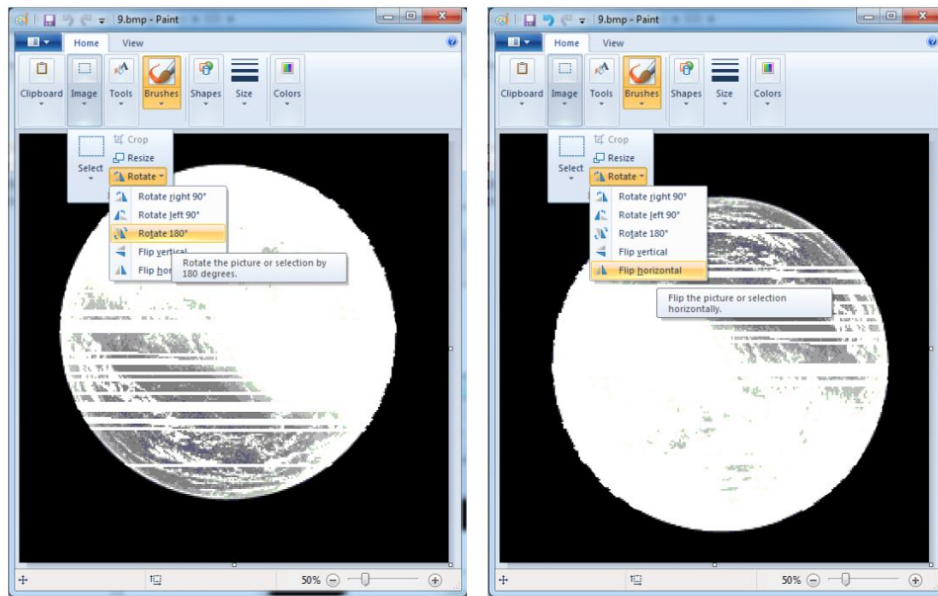
To mask an object, the coordinates of a square can be sent to CubeSense via telecommand 52. Every pixel that falls within this square will be ignored by the detection algorithms. To mask more complex objects in the FOV, multiple squares (up to 5) can be defined. To find the coordinates that should be entered, a full resolution bitmap image should be downloaded from CubeSense via the GSP program. Open the image with *Microsoft Paint* and hold the mouse over the top-left corner of the object to mask and note the minimum X and Y pixel values as shown in Figure 7. Then hold the mouse over the bottom-right corner and note the

maximum X and Y pixel values. These two coordinates define the masking square. Take great care to ensure correct placement of these masking squares. Testing should also be performed to verify correct masking. See the Node Definition for correct telemetry and telecommands to perform masking.



**Figure 7 – Object masking**

When the CubeSense is used with the within a CubeADCS running CubeACP, the image will first need to be transformed before reading the pixel coordinates. This is done by opening the image in *Microsoft Paint*, rotating the image 180°, flipping the image horizontally and then, reading the minimum X and Y pixel values at the top-left and the maximum X and Y at the bottom-right. These steps are shown in the figure below.



**Figure 8 – ACP image transformation**

## 5. Documentation History

Version	Author	Pages	Date	Description of Change
1.0	MK	ALL	15/02/2016	First draft
1.1	DS	ALL	29/05/2016	Some updates
1.2	GJVV	ALL	02/08/2016	Template and formatting update
1.3	DS	ALL	05/08/2016	Updated for new software
1.4-1.6	DS	ALL	19/01/2017	Various corrections
1.7	DS	ALL	08/02/2017	Shifted various info to the ICD
1.8	DS	ALL	11/03/2019	Removed 5V references
1.9	HW	3.4	26/07/2019	Improved camera axis image
1.10	DS, HW	ALL	05/09/2019	Changed document for V3 CubeSense
1.11	DS	ALL	06/01/2020	Updated TLMs and TCMDs