ORIGINAL ARTICLE

PCOT: an open-source toolkit for multispectral image processing

James Finnis[1] | Helen C. Miles[2] | Ariel Ladegaard[3] | Matt Gunn[4]

[1]Aberystwyth University
[2]Aberystwyth University
[3]Aberystwyth University
[4]Aberystwyth University

Correspondence
James Finnis, Department of Computer Science, Llandinam Building, Aberystwyth University, Penglais, Aberystwyth, SY23 3DB, United Kingdom
Email: jcf12@aber.ac.uk

Funding information
UK Space Agency, grants ST/V002066/1 and ST/Y005996/1

PCOT is a Python program and library which allows users to manipulate multispectral images and associated data. It is in active development in support of the ExoMars mission and intended to be used on data from the Rosalind Franklin rover, but it has much greater potential for use beyond this specific context. PCOT operates on a graph model - the data is processed through a set of nodes which manipulate it in various ways (e.g. add regions of interest, perform maths, splice images together, merge image channels, plot spectra). A PCOT document describes this graph, and we intend that documents are distributed along with the data they generate to help reproducibility. PCOT is open-source, and contributions can be made to the core software, as plugins, or by using PCOT as a library in your own code.

KEYWORDS
multispectral, spectroscopy, remote sensing, open-source, software

1 | INTRODUCTION

The *Rosalind Franklin* rover (see Fig. 1), part of the European Space Agency (ESA) ExoMars programme, is due to launch in 2028 and search for signs of life preserved in the sub-surface at Oxia Planum. Rosalind Franklin will carry a suite of scientific instruments for this purpose, and possesses the ability to drill two metres into the surface to extract samples for molecular analysis (Vago et al., 2017). The rover will be guided to potential sites of interest using images from the Panoramic Camera system *PanCam* (Coates et al., 2017), which teams stereo multispectral Wide-Angle Cameras (WACs) with the variable focus High Resolution Camera (HRC) which produces colour images using a Bayer pattern filter.

This paper introduces the PanCam Operations Toolkit (PCOT), a Python program and library being developed primarily to process and analyse images returned from the PanCam instrument, but lending itself to any kind of multispectral image processing, including uncertainty and data quality.

In preparation for the mission, processes were developed to analyse the image data, e.g. (Allender et al., 2020). The initial development in this area was *ExoSpec*, which allowed PanCam multispectral datasets to be processed to provide information about the distribution of spectral properties in the scene (Allender et al., 2018). ExoSpec was developed using the widely used ENVI framework and proprietary IDL language (NV5 Geospatial Solutions, 2024). However, the PanCam science and engineering team involves a very broad spectrum of academic and industry contributors world-wide, with diverse specialisms and career stages, many of whom do not have access to IDL or ENVI licences; therefore it was decided that an open-source alternative was required.

PCOT was created as a replacement for ExoSpec, providing a modern, versatile open-source analysis package for use during the mission. It has a strong focus on provenance, clarity of process, and repeatability. Given that most observational data contains uncertainty and error information, all PCOT values contain uncertainty and data quality information and propagate them through all calculations. PCOT is being developed in Python because of its wide use in the scientific community and the large number of available packages that it may utilise.

We are opening PCOT's source to the wider community and building extensibility into it from the ground up, in the hope that the community will build on it.

2 | RELATED WORK

The PCOT application and upper layers of the PCOT library constitute a visual dataflow language in which data (multispectral images, scalar values etc.) flow through a directed graph of operations (Johnston et al., 2004; Schwarzkopf, 2020). The systems which have most directly inspired PCOT are Miller Puckette's Max and Pure Data (Puckette, 1996) visual languages for audio processing , although there are some influences from the popular LabVIEW system design platform (Kodosky, 2020) and from the diagrams used by the MASCOT software engineering methodology (Simpson, 1986). One final system which should be mentioned is Harpia - a visual dataflow language which can process RGB images using the OpenCV library (S2i, 2009).

However, PCOT differs from these systems in the following respects:

- it is written in Python for extensibility and interoperation with other systems;
- it serves as both an application and a library, where the lower layers are available to end users;
- it is primarily designed for multispectral image processing but can also process other kinds of composite data (including user-defined data types);
- it can process data with uncertainty and quality information (see Sec. 3.4);
- all data carries source information specifying from which inputs it is ultimately derived, which can constitute an "audit trail."

In the context of other space exploration missions, NASA have significant heritage with their Multimission Image Processing Laboratory (MIPL), Video Image Communication And Retrieval *VICAR* (NASA-AMMOS, 1966), and Integrated Software for Imagers and Spectrometers v3 (ISIS3) (Rodriguez, 2024) software, which has been used over decades of missions (Maki et al., 2012). In the Earth Observation domain, GDAL is a comprehensive open-source software library for geospatial data (Rouault et al., 2024), and Orfeo Toolbox provides image processing aimed at remote sensing applications (CNES and CS, 2024).

The decision to develop PCOT rather than using an existing package stemmed from several requirements, the primary one being the intention to prioritise provenance and accuracy of the data through error propagation, data quality tracking, and traceability, features not available in the previously mentioned software. Secondary requirements were to be easy-to-use through a graphical user interface – while the packages noted above are largely command-line based – and to support users in writing Python plugins.

3 | SOFTWARE DESCRIPTION AND DEVELOPMENT

PCOT is designed to help scientists and engineers analyse PanCam data and produce useful secondary data products, although it lends itself to any task involving processing multispectral image data. For example, with PCOT you can:

- load ENVI multispectral images - currently only band-sequential (BSQ) interleaved 4-byte float although more subtypes can easily be added;
- load PDS4 (Planetary Data System 4 - the current iteration of NASA's data standard) multispectral images - again, currently limited to the "spec-rad" (spectral radiance) end products of the calibration pipeline although more can be added;
- load multiple images in other formats (e.g. PNG and raw data) and combine them into multispectral images;
- perform calibration tasks on these images;
- define and annotate regions of interest in the data;
- perform mathematical operations across entire images;
- view spectra, histograms, spectral parameter maps with false colour palettes as a result of these mathematical operations

and many other things besides. PCOT is highly extensible and open-source, so any missing functionality is easily added.

Following ExoSpec, PCOT is designed to aid the ExoMars science teams in meeting the science objective *to search for signs of past and present life on Mars (ExoMars Project Team, 2010)* by supporting the use of PanCam and Enfys (previously ISEM) in characterising the geological context and guiding target selection (Vago et al., 2017). It acts downstream from the Rover Operations Command Centre (ROCC) on PanCam images which have already been radiometrically and geometrically corrected (Fig. 2), and will form part of a suite of tools designed to exploit the stereo and multi-focus capabilities of PanCam (Paar et al., 2016; Traxler et al., 2022). As PCOT has been developed as a successor to ExoSpec, its primary purpose is to generate relative reflectance images and spectral parameter maps - images which are derived by mathematically combining and processing the values of the bands in the source image to highlight particular features in the spectrum. It is not intended to work directly with stereo or 3D datasets, as these will be handled by PRo3D and associated tools (Paar et al., 2016; Traxler et al., 2022) however there is scope in the future to develop interoperation between these tools, for example, allowing users to overlay spectral parameter maps onto the 3D data.

Spectral parameter maps are particularly useful for planning science activities, and much work has already been done on the discovery of useful spectral parameters for Mars exploration (Viviano et al., 2014). It will also be able to produce spectra from regions of interest. Beyond the already planned tasks, the flexible and adaptable nature of PCOT will allow it to perform a wide range of unforeseen calculations. It is anticipated that PCOT would be used at all stages of the PanCam teams' surface operations (Fig. 3), facilitating quick, data-driven decisions on a day-to-day tactical timescale, broader strategic work in the medium term, and longer-term analysis of collected data.

Although PCOT is designed to fit within the PanCam ecosystem of processing and tools, it functions independently, allowing it to be used for a far wider variety of purposes beyond ExoMars.

3.1 | A PCOT Document

The verifiability and reproducibility of data generated by PCOT is of paramount importance. To this end, a PCOT document is a data product which can be shared between users, which fully describes how the data was generated from the initial import to the final output of processed secondary products. Users are encouraged to exchange PCOT documents in addition to, or instead of, the generated images or data.

A PCOT document consists of:

- The **inputs** - data loaded from sources external to PCOT. These are kept separate from the graph (see below) to allow a different graph to be used on the same inputs, or the same graph to be used on different inputs.
- The **graph** - a set of nodes and connections between them which define operations to be performed on inputs.
- The **settings** - these are global to the entire application and hold information such as caption styles and font preferences.

3.1.1 | Inputs and Parameters

PCOT can handle many kinds of data. It is particularly suited to processing multispectral images with uncertainty and error data and can currently read ENVI and PDS4 formats, along with more common RGB formats (such as PNG) which can be collated into multispectral images. It is also possible to read raw binary data, provided some format information is specified by the user.

Parameters are currently stored in nodes and are initialised/defaulted when a node is created. Nodes which are copied create independent duplicates of the parameters. Parameters are also stored in PCOT document files (as part of the graph). A system for modifying these parameters from text files as part of a "batch processing system" is in progress. and graphs can be edited and run from Python programs without starting PCOT itself, using PCOT as a library.

PCOT parameters can be supplemented by data included in configuration files: for example, camera filter data such as bandwidth and bandpass are stored in text files, allowing easy modifications and additions by users.

3.1.2 | The Graph

A PCOT document manipulates data in a graph - a network of *nodes*, each of which takes some data, performs some operation on it, and may display or output derived data as the user requires. Crucially, this is a *directed acyclic graph:* each connection has a direction going from an output of one node to an input of another, so there cannot be any loops.

Fig. 5 shows the main PCOT interface with no nodes opened - this is PCOT as it would appear when first started. The two most important areas are the graph view on the right and the node view on the left. Fig. 6 shows PCOT with some nodes added to the graph and one of them (a *sink* node) open for editing.

3.2 | Nodes

Nodes are the basis of operations within PCOT; Fig. 7 shows two example nodes in the graph with key features annotated. Each node has a display text - this is usually the node's type, but in the case of the *expr* node(as shown in one of the examples in Fig. 7) it is the expression being calculated. Each node is shown as a box with input connections on the top and output connections at the bottom, and each connection may have a display label. The inputs and outputs are coloured by type: blue is perhaps the most common and indicates an image. Each node has a pop-up help box that gives information on its usage.

Nodes already available include:

- image recolouring methods such as contrast stretch, gradient and decorrelation stretch;
- region of interest selection, including basic shape selection, point selection, and painted regions;
- band depth calculation;
- a number of powerful region-of-interest editing nodes;
- radiometric calibration target detection assistance;
- image registration, with both auto and manual methods; and
- a powerful expression evaluator which allows users to manipulate data with arithmetic operators and functions - for example, "a$671 / a$438" divides the values in the 671nm band by the values in the 438nm band for the image on input *a,* providing a spectral parameter map for ferric minerals and dust (Allender et al., 2018). This is the calculation performed by the graph in Fig. 6.

Comment and constant nodes are also available but are the exception to the structure given above in that both take direct input from the user, allowing values to be typed into the node box in the graph. Examples are shown in Fig. 8. To support reproducibility and verifiability, the current versions of any nodes contained within the graph are recorded when saving a PCOT document, both by recording the node's version number and by creating an MD5 hash for the node's file. This ensures that the contents of the node file are checked regardless of whether the version number has been incremented. On loading a PCOT document, the versions of the nodes in the document are compared with those available in the local directory; if there is a variance then the user is warned of potential incompatibility as this may produce different results to what was expected.

3.2.1 | The Node Tab

Each node in the graph view can be "opened" by double-clicking, which will open a view of the node and its editable parameters in the node area. Multiple nodes can be open at a time, with each node getting a tab as shown in Fig. 6. Nodes can be "undocked" into separate windows by double-clicking the tab.

3.2.2 | The Canvas

All nodes which are linked to images make use of a standard "canvas" user interface component, as shown in the *sink* node open in Fig. 6. This has several features uniquely suiting it to multispectral work:

- the image can easily be panned and zoomed.
- The mapping of the image bands to the RGB channels of the viewed image can easily be modified (and if the "Guess RGB" button is clicked the closest bands to those visible colours will be selected).
- A "quick spectrum" view can be opened to show a spectrum of the pixel currently under the cursor which changes as the user navigates through the image.
- The image can be normalised for viewing using all bands, just the visible bands, or just the visible bands with each band being separately normalised. Additionally, the image can be normalised to the entire image or just the visible region.
- Uncertainty and Data Quality bits (see Sec. 3.4) can easily be viewed as overlays.

3.3 | Internal Architecture

The architecture of PCOT is shown in Fig. 4. The application is built upon a Python library consisting of:

- ImageCube and Value classes, encapsulating multispectral, scalar and other array data with uncertainty and data quality and with operations which propagate those quantities;
- The Datum class, which wraps all quantities in PCOT allowing them to interoperate, so that it is easy (for example) to define the result of multiplying an image by a scalar.
- Datum functions which can be called from both the application's expression parser and from Python code, and can be extended by the user. These operate on and return Datum objects.
- Datum type classes which describe the behaviours of different kinds of Datum, and allow new types to be easily registered by the user.

Within the application itself, new types of nodes can be created by writing new classes to define their behaviour.

3.4 | Uncertainty, data quality and internal image format

Most data types (such as images and scalars) can be derived from observational data, which often has associated uncertainty and data quality information. PCOT stores data of these types as triplets of nominal (mean) value, uncertainty (as standard deviation), and a set of data quality (DQ) bits. Nominal, uncertainty and DQ in images are stored on a per-pixel, per band basis. Both the nominal and uncertainty values are stored as 32-bit floating point values, while the DQ bits are 16-bit integers. This means each pixel is stored as 10 8-bit bytes per band, and a 1024x1024 image requires 10Mib of memory for each band.

Uncertainty calculations are propagated and compounded where appropriate, following the principles defined in the ISO standard Guide to Uncertainty Measurement (GUM) (JCGM, 2008). The calculations within PCOT were verified using the Python uncertainties package (Lebigot, 2009), which was not used directly to keep PCOT as lightweight as possible.

DQ bits are used to indicate whether the data has no uncertainty, whether it stems from an error in the observation, an invalid calculation (e.g. division by zero), or whether there is simply no data. These bits are also propagated through calculations - any value calculated from a value with a DQ bit set will also receive that DQ bit.

This data is propagated through all calculations where possible, and if this is not possible, the special data quality bit NOUNC (no uncertainty) is added. This generally happens where the calculation is intractable or uncertainty is irrelevant, such as a decorrelation stretch.

We are forced to assume that all quantities are independent, so care must be taken by users to avoid grossly incorrect results. For example, consider the graph in Fig. 9, consisting of a scalar constant node with the value 0 ± 1, feeding into an expression node as the variable a, where the expression calculated is $a - a$. This should produce 0 ± 0, but the actual result is $0 \pm \sqrt{2}$.

3.5 | Source tracking

Each datum handled by PCOT has information describing where that datum ultimately comes from:

- Non-image Datum objects have a source set, which is a collection of Source objects. Only data which are not derived from observational inputs are permitted to have a "null source set" consisting only of a single "null source."
- Images have a "multiband source", consisting of a source set for each band. This is still considered as a single source set - the union of the sets for each band - for some operations.
- Each individual source in a set can have information describing the origin of the data (for example a PDS4 Logical and Version Identifier or just a filename for other image sources), an integer giving the input number into which the source was originally loaded, and information describing the camera filter used to capture the data. Some of these may be absent: an image loaded from an RGB file will not contain specific bandwidth and bandpass data in its sources, for example.

When an operation is performed that combines two or more source sets, the resulting data's source set is the union of those sets. For example, adding two images together band-wise will result in an image in which the source set for each band is the union of the corresponding source sets of the addends.

3.6 | Extensibility

As noted above PCOT is highly extensible, with the user being able to manage multiple directories of Python "plugins" which can:

- Create new menu items,
- Create new node types,
- Create new functions for use in both the expression parser and Python code, and
- Create new PCOT data types for use in any of the above.

It is envisaged that a typical user might have a private plugins directory in their home directory and access to an organisation-wide plugins directory (or more than one).

3.7 | PCOT as a library

As well as being run as an application, PCOT allows users to write Python scripts which use PCOT components. This script can use core components like Datum, ImageCube and Value objects, but also parts of the application code such as documents and graphs. A typical example might be a script to read a PCOT document and run some data through that document's graph. We show an example of this in full, to show how simple this is:

```
# This example opens a graph, processes some ENVI files
# through that graph, and saves them as ENVI files.

import pcot
from pcot.document import Document
from pcot.datum import Datum
from pcot.dataformats import envi

# initialise PCOT
pcot.setup()
# load the document - this contains the graph we want to run
doc = Document("1.pcot")

# run the graph for some ENVI files. We'll just do one here.
for file in ("1",):  # extension omitted
    # load the given ENVI file into input 0
    rv = doc.setInputENVI(0, f"{file}.hdr")  # extension added
    if rv is not None:
        raise RuntimeError(f"{rv}")
    # run the graph by telling the document it has changed
    doc.changed()
    # get the "sink" node
    outNode = doc.getNodeByName("sink")
    # get its output Datum and, assuming it's an image,
    # get the image from it.
    img = outNode.out.get(Datum.IMG)
    print(f"Image size: {img.w} x {img.h} x {img.channels}")
    # write to new ENVI, e.g. 1b.hdr
    envi.write(f"{file}b",img)
```

It is also possible to load and process data without using the document/graph mechanism at all, using just the core operations on Datum objects:

```
from pathlib import Path
import pcot
from pcot.dataformats import load
import pcot.datumfuncs as df

pcot.setup()

# get a list of the files in the test data directory which are LWAC images
# (images from the left Wide Angle Camera)

testdatadir = "PCOTdata/rcp_output"
filenames = [str(x) for x in Path(testdatadir).glob("*l0*.xml")]

# and now load those files assuming they are PDS4 labels with data in files with corresponding
# names, merging them into a single image Datum. Each band in this
image will have a source set
# consisting of a single source, which contains information about the filter that was used
d = load.pds4(filenames)

# run the expression "norm((d%670)/(d%540))". The "%" operator has been overridden to allow
# individual bands to be extracted from images, so this will divide the 670nm band by the
# 540nm band, and then normalise the result to [0,1]. Then apply gamma correction by raising
# the result to the power 0.3 and extract the image from the resulting Datum.

res = df.norm( ((d%670) / (d%540))**0.3 ).get(Datum.IMG)

# write the resulting single-band image as a monochrome RGB.

res.rgbWrite("testout.png")
```

4 | SOFTWARE ACCESS AND USE

While PCOT is being developed primarily for PanCam science and engineering operations, it is relevant to any application where images are manipulated for scientific analysis, where the propagation of uncertainty data is important, and where the process used to manipulate them should be recorded. As it is primarily designed to work with multispectral images, PCOT is particularly suited to geoscience and biological science applications provided those images fit in memory (see Sec. 3.4 for a discussion of data size).

PCOT is actively in development, but is maintained using tested releases to ensure users can be involved in the development and work with functional releases. It is version-controlled, fully open-source and available under an MIT License on GitHub: `https://github.com/AU-ExoMars/PCOT`. An accompanying repository has recently been established to collect plugins for sharing amongst users: `https://github.com/AU-ExoMars/PCOT-Plugins`, and a 'cookbook' to record commonly used PCOT recipes which will be linked from the GitHub repository.

4.1 | Citing the software

If you use PCOT in your research we would be grateful if you could cite it as follows:

Finnis, J., Miles, H., Gunn, M., & Ladegaard, A. (2024). PanCam Operations Toolkit: PCOT. Zenodo. `https://doi.org/10.5281/zenodo.12819549`

5 | FUTURE WORK

PCOT is under active development in collaboration with the ExoMars science and engineering teams. The immediate priorities are:

- Allowing conversion of raw digital numbers (DNs) into radiance values by flat-fielding and exposure compensation - this duplicates work done in the ROCC pipeline, but will be useful for testing and for users of other camera systems.
- Allowing conversion of radiance images into reflectance using mission images of the PanCam Calibration Target and pre-flight recorded reflectance values.
- Processing data from the Enfys IR reflectance spectrometer instrument and combining it with PanCam data.
- Further developing support for source tracking and provenance; in the case of ExoMars we will explore the potential for PCOT graphs and parameters to be stored in the data archive.
- A Parameter Management and Batch System that will facilitate writing batch files by describing inputs and parameters and passing these to a runner application.
- Improving the user documentation and holding workshops, building a user community.

We will continue to build PCOT toward the launch in 2028 and beyond, for both the ExoMars mission and for others, with the direction and help of a user community of which we fervently hope readers of this short paper will be a part.

6 | CONCLUSIONS

PCOT is a Python application and library primarily intended for processing geological data from the PanCam instrument on the Rosalind Franklin Mars rover, although it has broad potential to be used for any task involving processing multispectral image data. PCOT can be used through a GUI or as a library, and uses a graph model to process data in steps that can be easily described and understood. Work done in PCOT is saved as a document which describes the graph, with the intention that this will facilitate the sharing of repeatable, shareable and explainable data analysis.

PCOT also stores uncertainty and data quality information with all its data, and propagates these through all calculations.

PCOT is open source and being co-designed with collaborators within the PanCam science team already, but there is enormous potential for PCOT to be used for a wide variety of geoscience applications. We encourage interested readers to get involved at every level, from using PCOT and sending feedback, to developing your own plugins, or contributing to the core development of the software. The source code is available online, see Sec. 4.

7 | ACKNOWLEDGEMENTS

The research presented in this paper was funded by the UK Space Agency grant ST/V002066/1, and ongoing development is funded by the UK Space Agency grant ST/Y005996/1. The authors are grateful to early users of PCOT who have already shaped it a great deal.

8 | CONFLICTS OF INTEREST

None to report.

9 | SUPPORTING INFORMATION

The following supporting information is available as part of the online article:

Source code S1. PCOT is open source software released under the MIT license. It is available online at `https://github.com/AU-ExoMars/PCOT` and at the DOI `https://doi.org/10.5281/zenodo.12819549`.

Video S2. A brief video demonstration of PCOT is available at `https://youtube.com/watch?v=FWEU3nOlqWg` demonstrating construction of a spectral parameter map of part of an image, and plotting spectra of a few points.

Table S3. The table at `https://au-exomars.github.io/PCOT/gettingstarted/connectors/` shows the patterns and colours currently used to indicate data type in node connectors.

10 | FIGURES

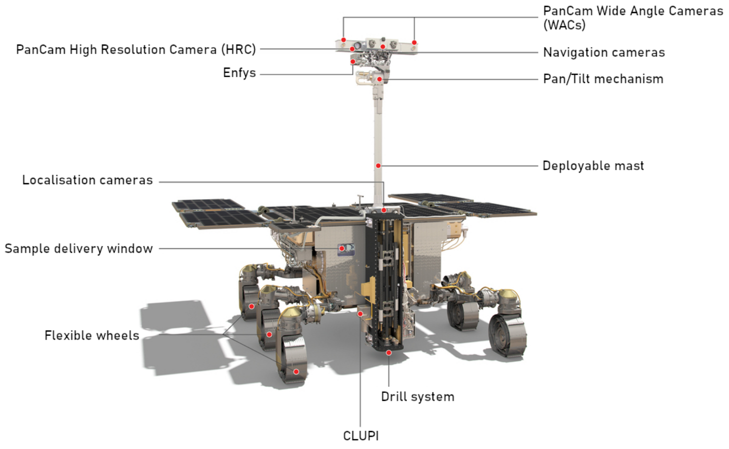


FIGURE 1 The Rosalind Franklin rover with key instruments labelled. Image credit: ESA/ATG medialab.

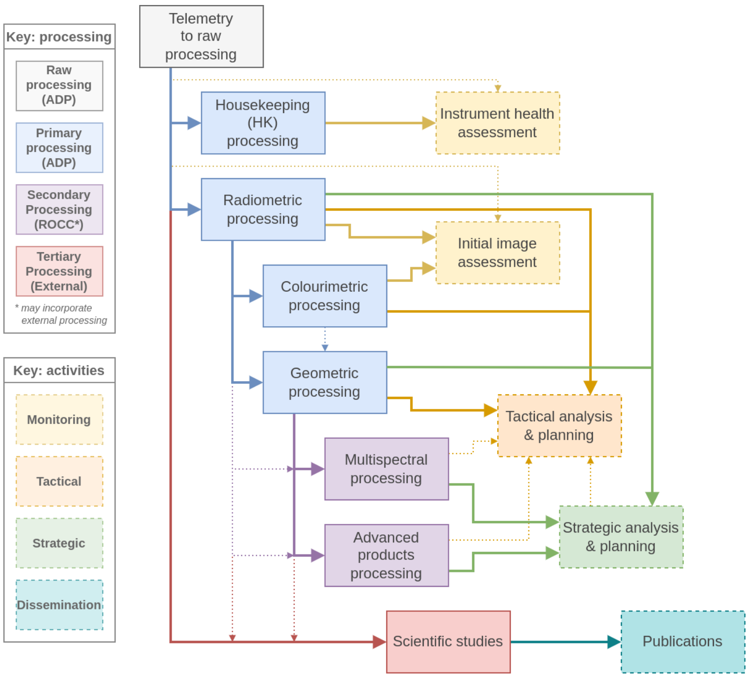

Key: processing
Raw processing (ADP)
Primary processing (ADP)
Secondary Processing (ROCC*)
Tertiary Processing (External)
* may incorporate external processing
Key: activities
Monitoring
Tactical
Strategic
Dissemination
Telemetry to raw processing
Housekeeping (HK) processing
Instrument health assessment
Radiometric processing
Initial image assessment
Colourimetric processing
Geometric processing
Tactical analysis & planning
Multispectral processing
Strategic analysis & planning
Advanced products processing
Scientific studies
Publications

FIGURE 2 An overview of the PanCam ground processing pipeline components and the rover operations activities they feed into; image reproduced from (Ladegaard et al., 2023) with permission. permission.

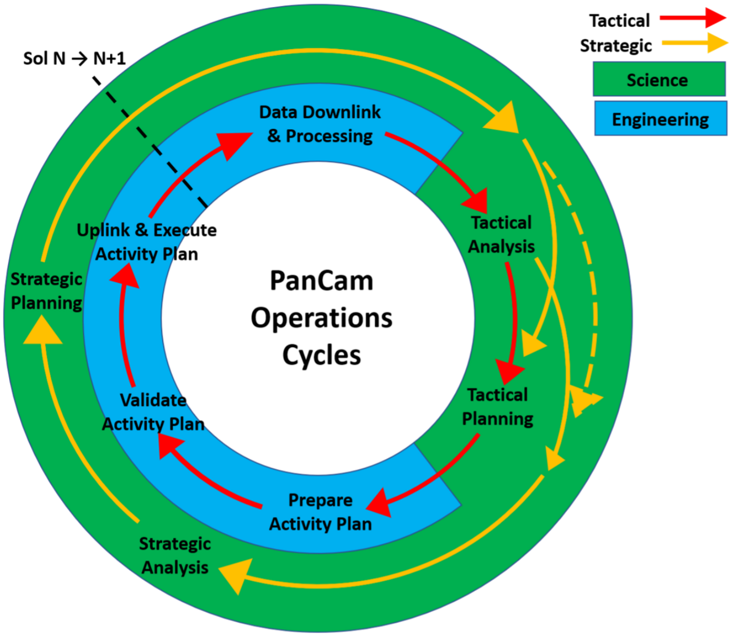


FIGURE 3 The planned cycle structure for PanCam operations, highlighting the very short-term tactical timeline, and how it is interlinked with the medium-term strategic timeline.

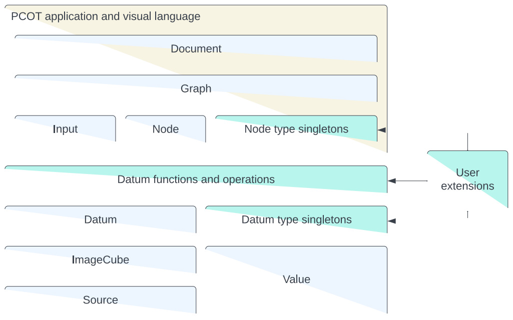


FIGURE 4 Primary PCOT software components

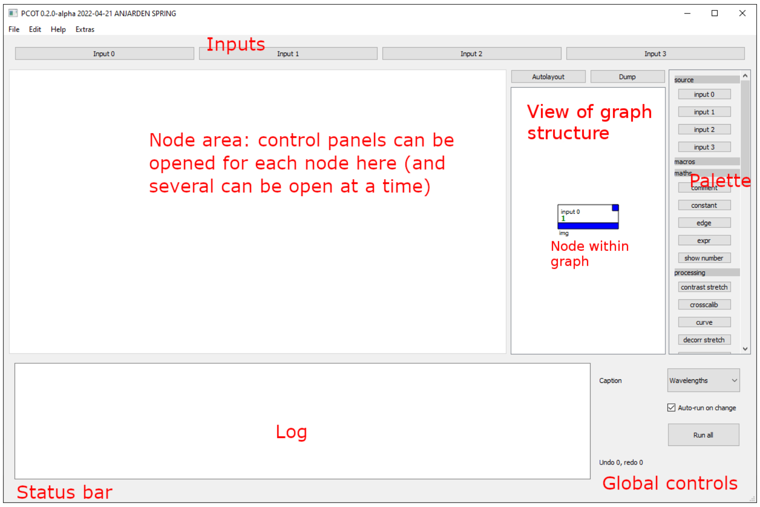
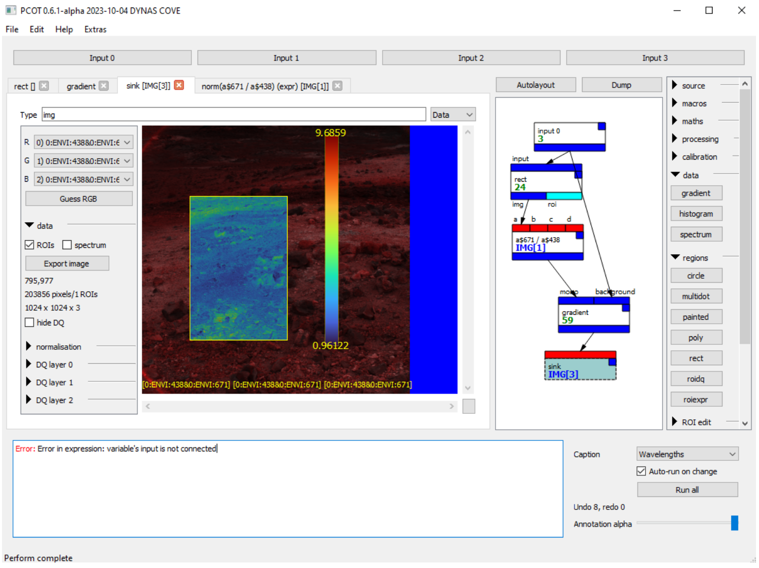


FIGURE 5 The PCOT graphical user interface, with red text noting the main areas of the interface.



```
R 0) 0:ENVI:438&0:ENVI:6
G 1) 0:ENVI:438&0:ENVI:6
B 2) 0:ENVI:438&0:ENVI:6
795,977
203856 pixels/1 ROIs
1024 x 1024 x 3
9.6859
0.96122
[0:ENVI:438&0:ENVI:671] [0:ENVI:438&0:ENVI:671] [0:ENVI:438&0:ENVI:671]
input 0
3
input
rect
24
img roi
a b c d
a$671 / a$438
IMG[1]
mono background
gradient
59
sink
IMG[3]
Error: Error in expression: variable's input is not connected
Undo 8, redo 0
Perform complete
```

FIGURE 6 An example of PCOT being used to extract a region of interest from an image, manipulate the region of interest, then reinsert it into the original image

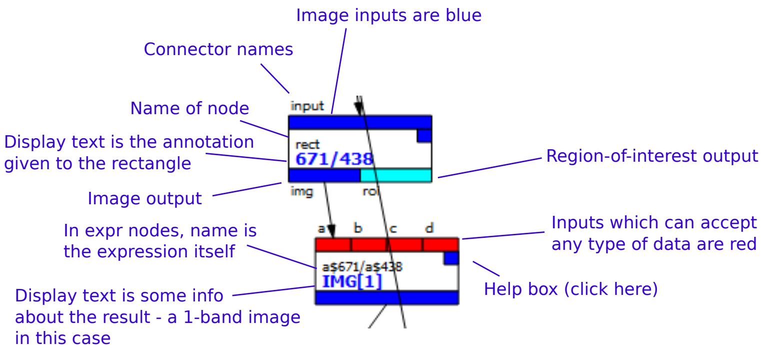
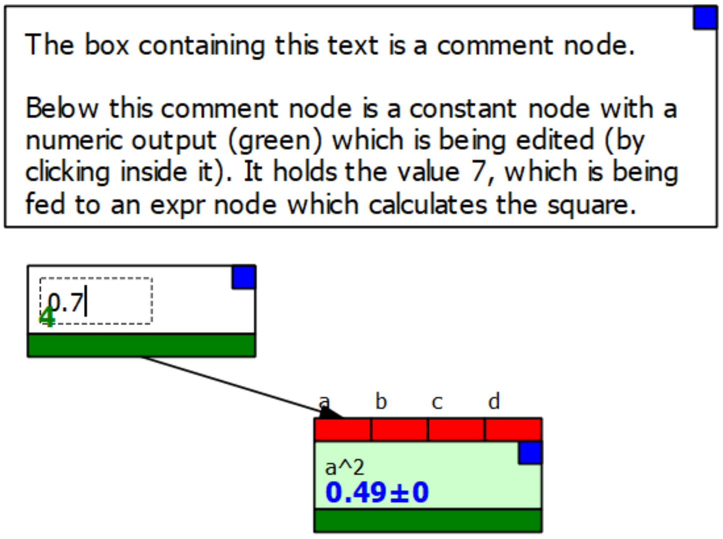


Image inputs are blue
Connector names
Name of node
Display text is the annotation given to the rectangle
Region-of-interest output
Image output
In expr nodes, name is the expression itself
Inputs which can accept any type of data are red
Help box (click here)
Display text is some info about the result - a 1-band image in this case
input
rect
671/438
img
roi
a
b
c
d
a$671/a$438
IMG[1]

FIGURE 7 Two example nodes of a PCOT graph, annotated to show the inputs, outputs, and key features.


The box containing this text is a comment node.

Below this comment node is a constant node with a numeric output (green) which is being edited (by clicking inside it). It holds the value 7, which is being fed to an expr node which calculates the square.

FIGURE 8 Examples of comment and constant nodes; these are distinct from the other types of node available in PCOT.

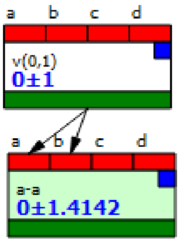


FIGURE 9 A graph producing an incorrect uncertainty due to an assumption of independence

REFERENCES

E. J. Allender, C. R. Cousins, M. D. Gunn, and C. M. Caudill. Multiscale and Multispectral Characterization of Mineralogy with the ExoMars 2022 Rover Remote Sensing Payload. *Earth and Space Science*, 7(4), apr 2020. doi: 10.1029/2019ea000692. URL https://doi.org/10.1029%2F2019ea000692.

Elyse J. Allender, Roger B. Stabbins, Matthew D. Gunn, Claire R. Cousins, and Andrew J. Coates. The ExoMars Spectral Tool (ExoSpec): an image analysis tool for ExoMars 2020 PanCam imagery. In Lorenzo Bruzzone, Francesca Bovolo, and Jon Atli Benediktsson, editors, *Image and Signal Processing for Remote Sensing XXIV*. SPIE, oct 2018. doi: 10.1117/12.2325659. URL https://doi.org/10.1117%2F12.2325659.

CNES and CS. ORFEO ToolBox. 2024. URL https://www.orfeo-toolbox.org. Accessed on Fri, August 02, 2024.

A.J. Coates, R. Jaumann, A.D. Griffiths, C.E. Leff, N. Schmitz, J.-L. Josset, G. Paar, M. Gunn, E. Hauber, C.R. Cousins, R.E. Cross, P. Grindrod, J.C. Bridges, M. Balme, S. Gupta, I.A. Crawford, P. Irwin, R. Stabbins, D. Tirsch, J.L. Vago, T. Theodorou, M. Caballo-Perucha, G.R. Osinski, and the PanCam Team. The PanCam Instrument for the ExoMars Rover. *Astrobiology*, 17(6-7):511–541, jul 2017. doi: 10.1089/ast.2016.1548. URL https://doi.org/10.1089%2Fast.2016.1548.

ExoMars Project Team. ExoMars Programme Objectives Document. *EXM-MS-RS-ESA-00007*, 2.0, 2010.

JCGM. Evaluation of measurement data — Guide to the expression of uncertainty in measurement. *100:2008(E)*, 2008. doi: 10.59161/jcgm100-2008e. URL http://dx.doi.org/10.59161/JCGM100-2008E.

Wesley M Johnston, JR Paul Hanna, and Richard J Millar. Advances in dataflow programming languages. *ACM computing surveys (CSUR)*, 36(1):1–34, 2004.

Jeffrey Kodosky. LabVIEW. *Proceedings of the ACM on Programming Languages*, 4(HOPL):1–54, 2020.

Ariel Ladegaard, Matt Gunn, Helen C. Miles, and Laurence Tyler. AUPE: An Emulator for the ExoMars PanCam Instrument. In Peter Vangorp and David Hunter, editors, *Computer Graphics and Visual Computing (CGVC)*. The Eurographics Association, 2023. ISBN 978-3-03868-231-8. doi: 10.2312/cgvc.20231199.

Eric O. Lebigot. Uncertainties: a Python package for calculations with uncertainties. 2009. URL https://uncertainties.readthedocs.io/en/latest/index.html. Accessed on Fri, August 02, 2024.

J. Maki, D. Thiessen, A. Pourangi, P. Kobzeff, T. Litwin, L. Scherr, S. Elliott, A. Dingizian, and M. Maimone. The Mars Science Laboratory Engineering Cameras. *Space Science Reviews*, 170(1–4), may, pages=77–93 2012. ISSN 1572-9672. doi: 10.1007/s11214-012-9882-4. URL http://dx.doi.org/10.1007/s11214-012-9882-4.

NASA-AMMOS. VICAR: Video Image Communication And Retrieval. *GitHub*, 1966. URL https://github.com/NASA-AMMOS/VICAR. Accessed on Fri, August 02, 2024.

NV5 Geospatial Solutions. ENVI image processing and analysis software. 2024. URL https://www.nv5geospatialsoftware.com/Products/ENVI. Accessed: 2024-07-23.

Gerhard Paar, Robert G. Deen, Jan-Peter Muller, Nuno Silva, Peter Iles, Affan Shaukat, and Yang Gao. Vision and Image Processing, aug, pages=105–179 2016. URL http://dx.doi.org/10.1002/9783527684977.ch3.

Miller Puckette. Pure Data: another integrated computer music environment. *Proceedings of the second intercollege computer music concerts*, pages 37–41, 1996.

Kelvin Rodriguez. Integrated Software for Imagers and Spectrometers (ISIS) 8.0.3, 2024. URL https://code.usgs.gov/astrogeology/isis/-/tags/8.0.3.

Even Rouault, Frank Warmerdam, Kurt Schwehr, Andrey Kiselev, Howard Butler, Mateusz Łoskot, Tamas Szekeres, Etienne Tourigny, Martin Landa, Idan Miara, Ben Elliston, Kumar Chaitanya, Lucian Plesea, Daniel Morissette, Ari Jolma, Nyall Dawson, Daniel Baston, Craig de Stigter, and Hiroshi Miura. GDAL, 2024. URL https://zenodo.org/doi/10.5281/zenodo.12545688.

S2i. Harpia - development of a graphic interface for learning, implementation and management of vision systems, 2009. URL `https://s2i.das.ufsc.br/harpia/en/home.html`.

Malte Schwarzkopf. The Remarkable Utility of Dataflow Computing. https://www.sigops.org/2020/the-remarkable-utility-of-dataflow-computing/, March 2020. URL `https://www.sigops.org/2020/the-remarkable-utility-of-dataflow-computing/`. Accessed on Thu, March 14, 2024.

Hugo Simpson. The MASCOT method. *Software Engineering Journal*, 1(3):103–120, 1986.

Christoph Traxler, Thomas Ortner, Gerd Hesina, Robert Barnes, Sanjeev Gupta, Gerhard Paar, Jan-Peter Muller, Yu Tao, and Konrad Willner. The PRoViDE Framework: Accurate 3D Geological Models for Virtual Exploration of the Martian Surface from Rover and Orbital Imagery, apr, pages=33–55 2022. URL `http://dx.doi.org/10.1002/9781119313922.ch3`.

Jorge L. Vago, Frances Westall, Pasteur Instrument Teams Landing S, Andrew J. Coates, Ralf Jaumann, Oleg Korablev, Valérie Ciarletti, Igor Mitrofanov, Jean-Luc Josset, Maria Cristina De Sanctis, Jean-Pierre Bibring, Fernando Rull, Fred Goesmann, Harald Steininger, Walter Goetz, William Brinckerhoff, Cyril Szopa, François Raulin, Frances Westall, Howell G. M. Edwards, Lyle G. Whyte, Alberto G. Fairén, Jean-Pierre Bibring, John Bridges, Ernst Hauber, Gian Gabriele Ori, Stephanie Werner, Damien Loizeau, Ruslan O. Kuzmin, Rebecca M. E. Williams, Jessica Flahaut, François Forget, Jorge L. Vago, Daniel Rodionov, Oleg Korablev, Håkan Svedhem, Elliot Sefton-Nash, Gerhard Kminek, Leila Lorenzoni, Luc Joudrier, Viktor Mikhailov, Alexander Zashchirinskiy, Sergei Alexashkin, Fabio Calantropio, Andrea Merlo, Pantelis Poulakis, Olivier Witasse, Olivier Bayle, Silvia Bayón, Uwe Meierhenrich, John Carter, Juan Manuel García-Ruiz, Pietro Baglioni, Albert Haldemann, Andrew J. Ball, André Debus, Robert Lindner, Frédéric Haessig, David Monteiro, Roland Trautner, Christoph Voland, Pierre Rebeyre, Duncan Goulty, Frédéric Didot, Stephen Durrant, Eric Zekri, Detlef Koschny, Andrea Toni, Gianfranco Visentin, Martin Zwick, Michel van Winnendael, Martín Azkarate, Christophe Carreau, and the ExoMars Project Team. Habitability on Early Mars and the Search for Biosignatures with the ExoMars Rover. *Astrobiology*, 17(6-7):471–510, jul 2017. doi: 10.1089/ast.2016.1533. URL `https://doi.org/10.1089%2Fast.2016.1533`.

Christina E. Viviano, Frank P. Seelos, Scott L. Murchie, Eliezer G. Kahn, Kimberley D. Seelos, Howard W. Taylor, Kelly Taylor, Bethany L. Ehlmann, Sandra M. Wiseman, John F. Mustard, and M. Frank Morgan. Revised CRISM spectral parameters and summary products based on the currently detected mineral diversity on Mars. *Journal of Geophysical Research: Planets*, 119(6), jun, pages=1403–1431 2014. ISSN 2169-9100. doi: 10.1002/2014je004627. URL `http://dx.doi.org/10.1002/2014JE004627`.