PyTrx: a Python-based monoscopic terrestrial photogrammetry toolset for glaciology

Penelope How[1], Nicholas R.J. Hulton[2], Lynne Buie[2], Douglas I. Benn[3]

how@asiaq.gl
@DrPennyHow

1 ASIAQ GREENLAND SURVEY
2 THE UNIVERSITY OF EDINBURGH
3 University of St Andrews

1. Introducing PyTrx

PyTrx (short for 'Python Tracking') is a **freely-available**, **object-oriented toolset** (Fig. 1) programmed in Python, created for the purpose of calculating real-world measurements from oblique images and time-lapse image sequences.

Its functionality includes **deriving velocities through feature-tracking**, **automated and manual area/line detection**, **image registration**, **camera calibration and optimisation**, and **georectification** [1].

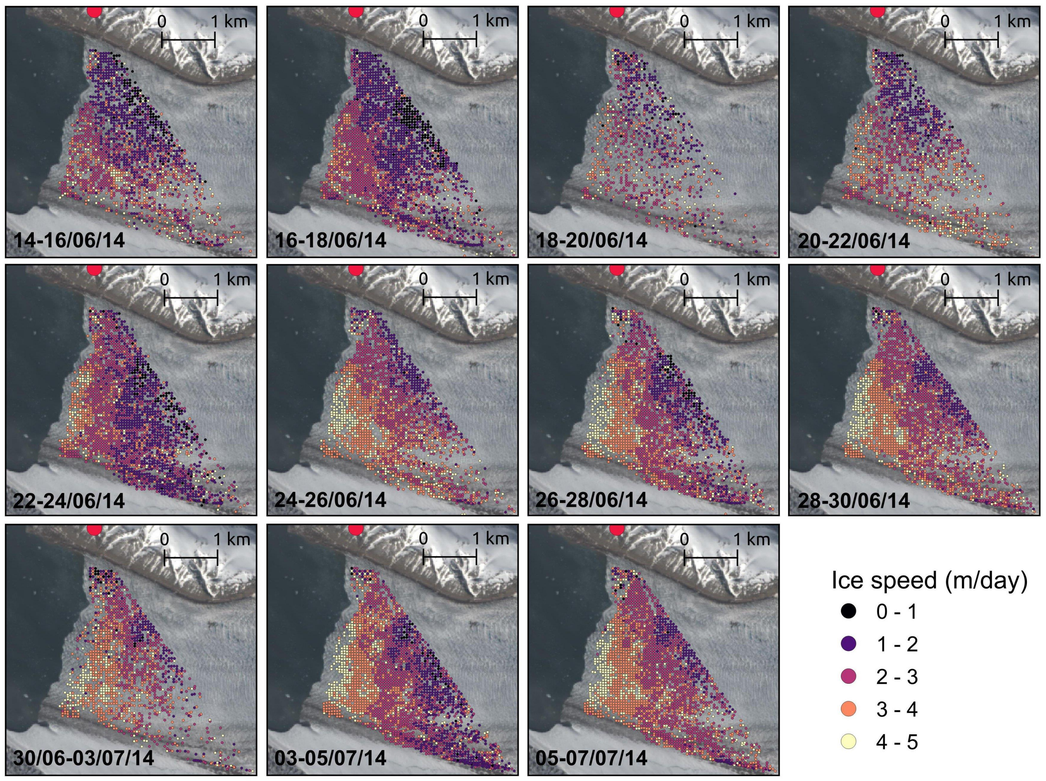

Fig 2. *An example of PyTrx's template matching capabilities on time-lapse images from Kronebreen, Svalbard (camera location denoted by red point). The sequence of velocity maps show an early season speed-up at the terminus, where velocities increase from an average of 2.5 m/day (14--16 June, first panel) to 4.7 m/day (5--7 July, last panel). Ice velocities were computed as 10×10 pixel templates (represented as points) over a 50×50 m grid, matched using normalised cross-correlation, and georectified using a camera model composed of its intrinsic characteristics (i.e. focal length, principal point, skew and lens distortion coefficients) and extrinsic information (i.e. camera location and pose, DEM, and ground control points).*

2. Deriving velocities via template matching or Optical Flow

PyTrx offers **two separate methods for deriving velocities** from image pairs:

1. A **traditional template matching method**, modelled from [2], [3] and [4], which calculates displacements through a grid of templates. This produces regular velocity measurements as shown in Fig. 2.

2. A **sparse Optical Flow matching method**, which calculates displacements as a set of small templates determined by corner coherency. This produces a collection of densely populated point velocities.

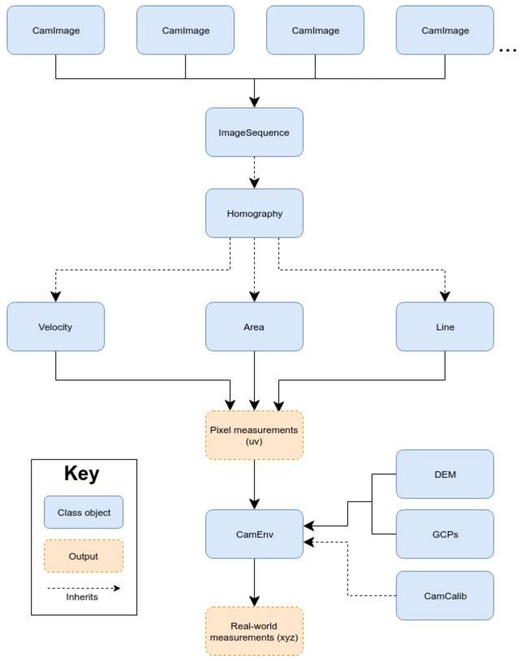

Fig 1. *PyTrx is designed with object-oriented flow, meaning that each major processing element is formed as an object with variables and functions. PyTrx's main functionality is available as stand-alone functions that handle operations between an image pair, or as callable commands within the toolset's objects that handle operations over an entire image sequence.*

3. Determining area/line features through automated and manual detection

Area and line features, such as supraglacial lake outlines, meltwater plume footprints (Fig. 3) and terminus positions, can be delineated through either **automated or manual detection methods** in PyTrx.

Pixel enhancement and **image segmentation** is used to automatically identify and delineate areal features, whilst the manual method offers **interactive plotting** for the user to define features on each inputted image.

Get PyTrx at:
pip install pytrx

github.com/PennyHow/PyTrx – for example scripts and applications
pytrx.readthedocs.io – for toolset guide and documentation

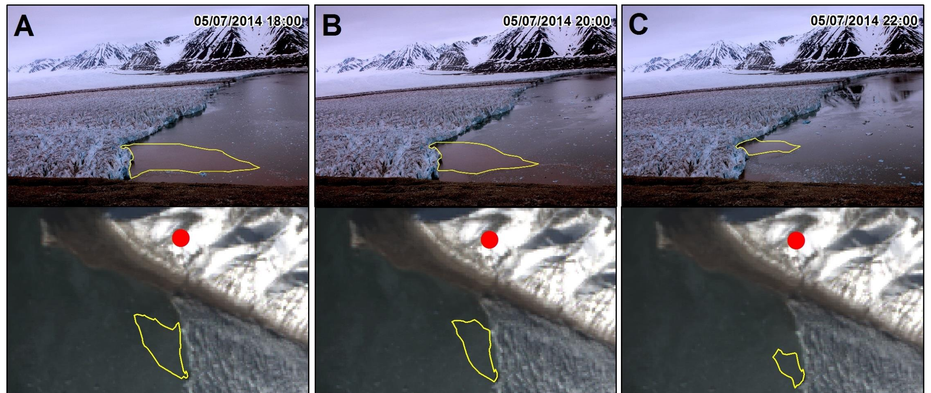

Fig 3. *Changes in meltwater plume footprints delineated manually from a sequence of time-lapse images from Kronebreen, Svalbard (red point denoting camera location) [5]. The surface expression of the meltwater plume has been distinguished through images captured on 5 July 2014 at 18:00 (A), 20:00 (B), and 22:00 (C), and demonstrates part of its diurnal recession. Each plot shows the plume definition in the image plane (top), and its georectification to real-world coordinates (bottom).*

4. Camera optimisation and error estimation with GCPs

Errors in the camera model can be constrained through the **optimisation routine** available in PyTrx, which refines projection of the 3D scene to the image plane based on the positions of a set of ground control points (GCPs).

The remaining difference between the 3D GCP positions anf their corresponding reprojected positions from the image are used as **a direct measure of error from the georectification process** (Fig. 4).

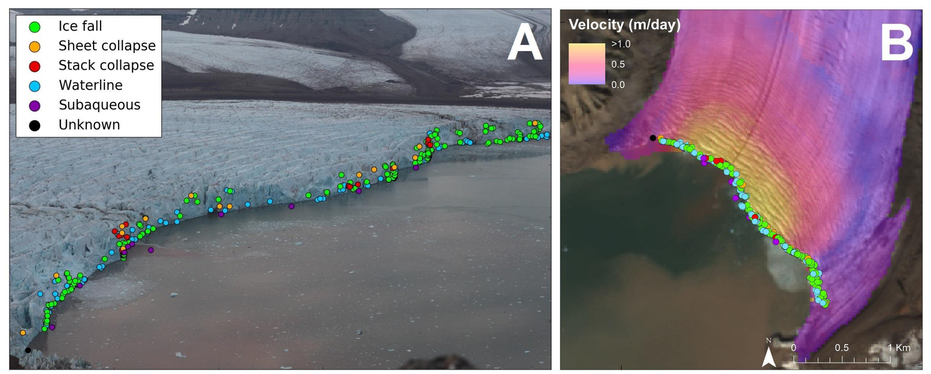

Fig 4. *Calving events observed across a 30-hour period (7–8 August 2015) at Tunabreen, Svalbard defined in the image plane (A) and georectified using PyTrx (B) [6]. Events were manually detected from image captured every three seconds, from which the style of calving was interpreted. The camera model was refined using the Trust Region Reflective algorithm implemented in PyTrx's optimization routine, producing a general error estimate of 3 m. This is a fairly conservative estimate, and reflects the maximum error in the far region in the image where the GCPs were defined.*

References: 1] How et al. (2020) *Frontiers Earth Sci.* [2] Messerli & Grinsted (2015) *Geosci. Instrum. Method Data Sys.* 4, 23-34. [3] James et al. (2016) *J. Glaciol.* 62(231), 159-169. [4] Schwalbe & Maas (2017) *J. Earth Surf. Dynam.* 5, 861-879. [5] How et al. (2017) *Cryosphere* 11, 2691-2710. [6] How et al. (2019) *Ann. Glaciol.* 60(78), 20-31