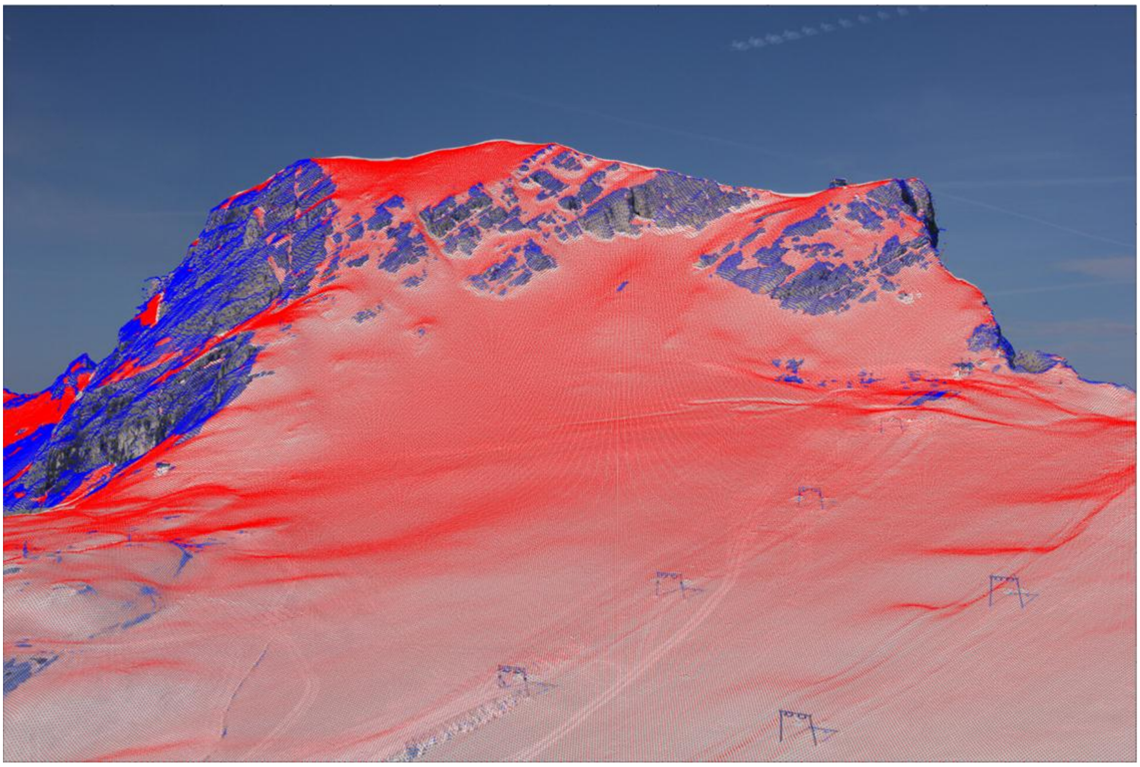
# PRACTISE
# –
# Photo Rectification And ClassificaTIon SoftwarE



**– PRACTISE V.1.0 User's Manual –**

**Stefan Härer,**
**Matthias Bernhardt,**
**Javier G. Corripio**
**&**
**Karsten Schulz**

Stefan Härer
Department of Geography
LMU Munich
Luisenstraße 37
80333 Munich
Germany
e-mail: s.haerer@iggf.geo.uni-muenchen.de



19 October 2012

# Contents

# 1. Introduction

Digital cameras project the real world (3D) into an image plane (2D) and simultaneously, store the spectral information of landscapes in the 3 visible bands: red, green and blue. As commercial camera systems are cost-effective and easy-to-use, this ability makes camera images to a valuable source of information in research (e.g. in phenology, land cover studies, volcanology, glaciology and snow hydrology). The re-projection of the 2D photograph to the 3D landscape allows furthermore a quantitative spatial analysis and thus the derivation of the status of spatially distributed land surface parameters. The Matlab routine PRACTISE (Photo Rectification And ClassificaTIon SoftwarE) was developed to comfortably execute this task. The freely available code (V.1.0) is focused on snow cover patterns in mountainous terrain whereas it can be easily adapted to any other land surface variable. The software includes several modules where the user can (de)activitate optional features which are given in brackets:

- (Accuracy assessment of the camera location and the field of view)
- (Global optimisation of the camera location and the field of view)
- (Viewshed generation)
- Projection
- Automatic or manual snow classification

A bulk mode facilitates an analysis of a complete time series of photographs.

We herein focus on the application of PRACTISE to derive a snow cover map while we refer the reader to the publication "Härer, S., M. Bernhardt, J. G. Corripio and K. Schulz (2012): PRACTISE – Photo Rectification And ClassificaTIon SoftwarE, GMD" if a detailed description of the applied algorithms is needed.

# 2. PRACTISE example

The necessary inputs of PRACTISE are:

- a photograph,
- a digital elevation model (DEM),
- the camera location $\boldsymbol{C_o}$,
- the camera target point $\boldsymbol{T}$ (centre of the photograph),
- and the inner camera orientation (focal length $f$, sensor (CCD or CMOS) width $w$ and height $h$).

Optional is the declaration of ground control points (GCPs) that are only needed for the accuracy assessment and/or the global optimisation routine. A viewshed can also be generated externally in e.g. any geoinformation system and given as an input.

We will present the functionality using a test case including a DEM (Fig. 1a) and a photograph of Schneefernerkopf (11 May 2011, 8.15 a.m), Zugspitze, Germany (Fig. 1b). The necessary input data, as well as the example output is supplied in the distribution of PRACTISE for demonstration issues. The spatial resolution of the DEM is 30 m and the photograph has about 4.5 Megapixel (Mpx). From a user's point of view, the input file ("Input_PRACTISE.m") is the only file where changes have to be specified. Beforehand, we want to emphasize that no changes in the input file in the condition terms of the "if-else-statements" are necessary. The processing will be illustrated step by step.

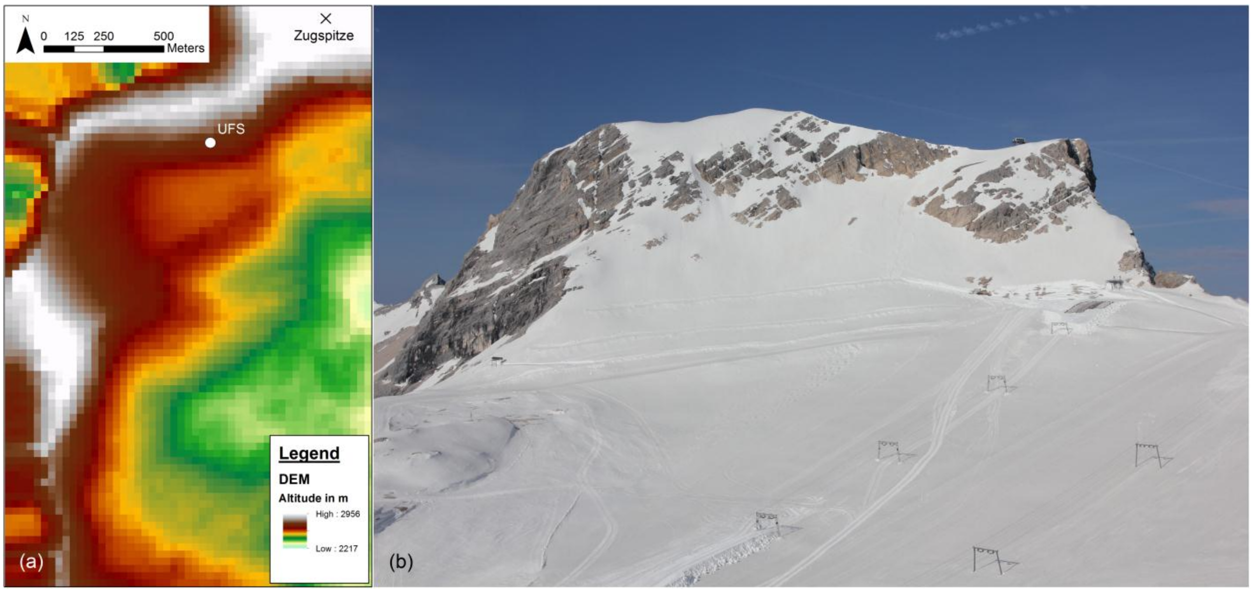



**Figure 1: (a) The used DEM of the area in the southwest of Zugspitze mountain in the Bavarian Alps (Germany) has a spatial resolution of 30 m. (b) The photograph (4.5 Mpx) shows the northeast facing slope Schneefernerkopf taken from the Environmental Research Station Schneefernerhaus (UFS) on 11 May 2011 at 8.15 a.m..**

## 2.1. Input

In our example, we adapt the module switches in the input file to:

- `vs=true` (viewshed generation),
- `gs=true` (accuracy assessment of $C_o$ and the field of view using GCPs),
- `os=true` (optimisation of $C_o$ and the field of view using GCPs),
- `cs=true` (automatic classification),
- `is=false` (classification of a single TIFF photograph).

Hence, the viewshed generation, the accuracy assessment and the optimisation using GCPs are switched on, as well as the automatic classification module for a single image is chosen.

The next step is the declaration of the input and output folders and files, these are:

- `fin_image='photos/ufs20110511_0815_4dot5Mpx.tif'` (location and name of the photograph),
- `fin_demW='input/dem_30m.txt'` (location and name of the DEM file),

- `N_gcpW=6` and `fin_gcpW='input/GCPortho6_4dot5Mpx.txt'` (number of GCPs, location and name of the GCP file),
- `fout_folder='output/'` (location of the output files),
- `fout_viewW_asc='ufs20110511_0815_4dot5Mpx_4dot5Mpx_view.asc'` (name of the generated viewshed file),
- `fout_classW_asc='ufs20110511_0815_4dot5Mpx_class.asc'` (name of the derived snow cover map file),
- `fout_mat='ufs20110511_0815_4dot5Mpx_proj.mat'` (name of the saved Matlab workspace including all important variables of the processing).

After the declaration of the files, the camera location and its field of view have to be specified:

- `cam(:,1)=[649299.97, 5253358.26]` (longitude and latitude of $\boldsymbol{C_o}$ in m, the altitude will be calculated using the DEM and "cam_off")
- `cam_off=1.5` (installation height of the camera above the surface in m)
- `cam(:,2)=[648740.85, 5252771.33]` (longitude and latitude of $\boldsymbol{T}$ in m, the altitude will be calculated using the DEM)
- `cam_rol=0.0` (roll angle of the camera in degree, where horizontal is equal to 0 and clockwise spans from 0 to +90, while anti-clockwise ranges from -90 to 0)
- `cam_foc=0.031` (focal length of the camera lens in m)
- `cam_hei=0.0149` and `cam_wid=0.0223` (height and width of the CCD of the camera in m)
- `pix_r=1728` and `pix_c=2592` (number of pixel rows and columns in the photograph)

Then, two parameters are needed for the automatic classification procedure:

- `thres_b_orig=127` (first guess of the minimum snow threshold in the blue band)
- `movavgwindow=5` (size of the smoothing window for the histogram of the blue band)

At last, the DDS parameters for the optimisation module have to be defined by the user:

- `UBD=[150, 150, 5, 2, 100, 100, 0.005]` and `LBD=[-150, -150, -1.0, -2, -100, -100, -0.005]` (upper and lower boundary deviations from the seed parameters of the optimisation: "cam(:,1)", "cam_off", "cam_rol", "cam(:,2)", "cam_foc")
- `DDS_R=0.2` (Neighbourhood perturbation size, i.e. the maximum change of a parameter in one iteration step in relation to the corresponding parameter range)
- `DDS_MaxEval=3000` (maximum number of optimisation iterations)

We want to note here, that normally neither the automatic classification parameters "thres_b_orig" and "movavgwindow" nor the DDS parameter "DDS_R" have to be changed.

## 2.2. Results

The results are computed by evaluating the file PRACTISE.m and are subsequently saved in the above specified output location. The generated text files for the viewshed (Fig. 2a) and the snow cover map (Fig. 2b) can be loaded directly in any geoinformation system. The saved Matlab workspace includes all important input and output variables of the processing. The corresponding variable descriptions can be found in Appendix 1. Additionally, up to three Matlab figures are stored by default in the output folder (if the corresponding module is activated):

- "DDS_opti.fig" (if "os=true": iterative optimisation path, Fig. 3a)
- "DDS_freq_hist.fig" (if "cs=true": histogram of the extracted DN in the blue band of the photograph including the snow threshold, Fig. 3b)
- "OverlayClass.fig" (in any case: overlay of the classified DEM pixels and the photograph, Fig. 3c)

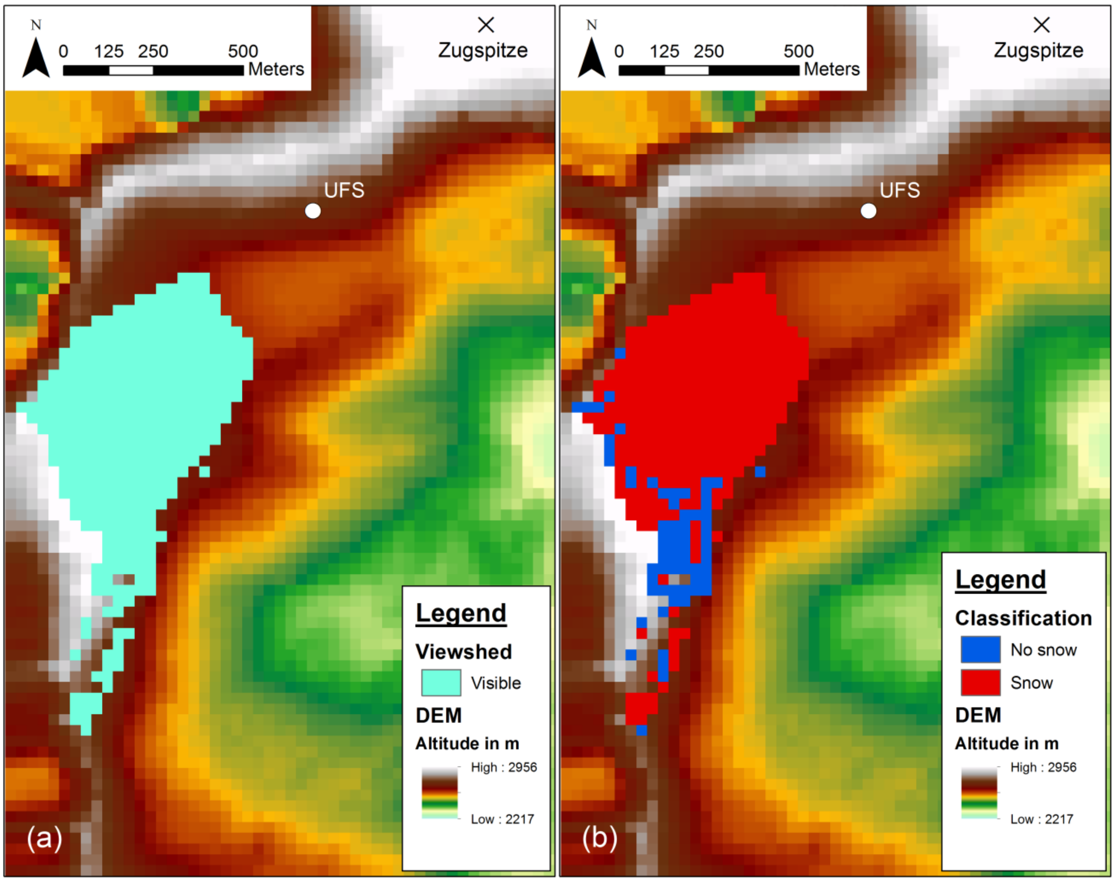



**Figure 2: (a) The optimised viewshed and (b) the snow cover map using the automatic classification are illustrated for the photograph (4.5 Mpx) of 11 May 2011 at 8.15 a.m..**

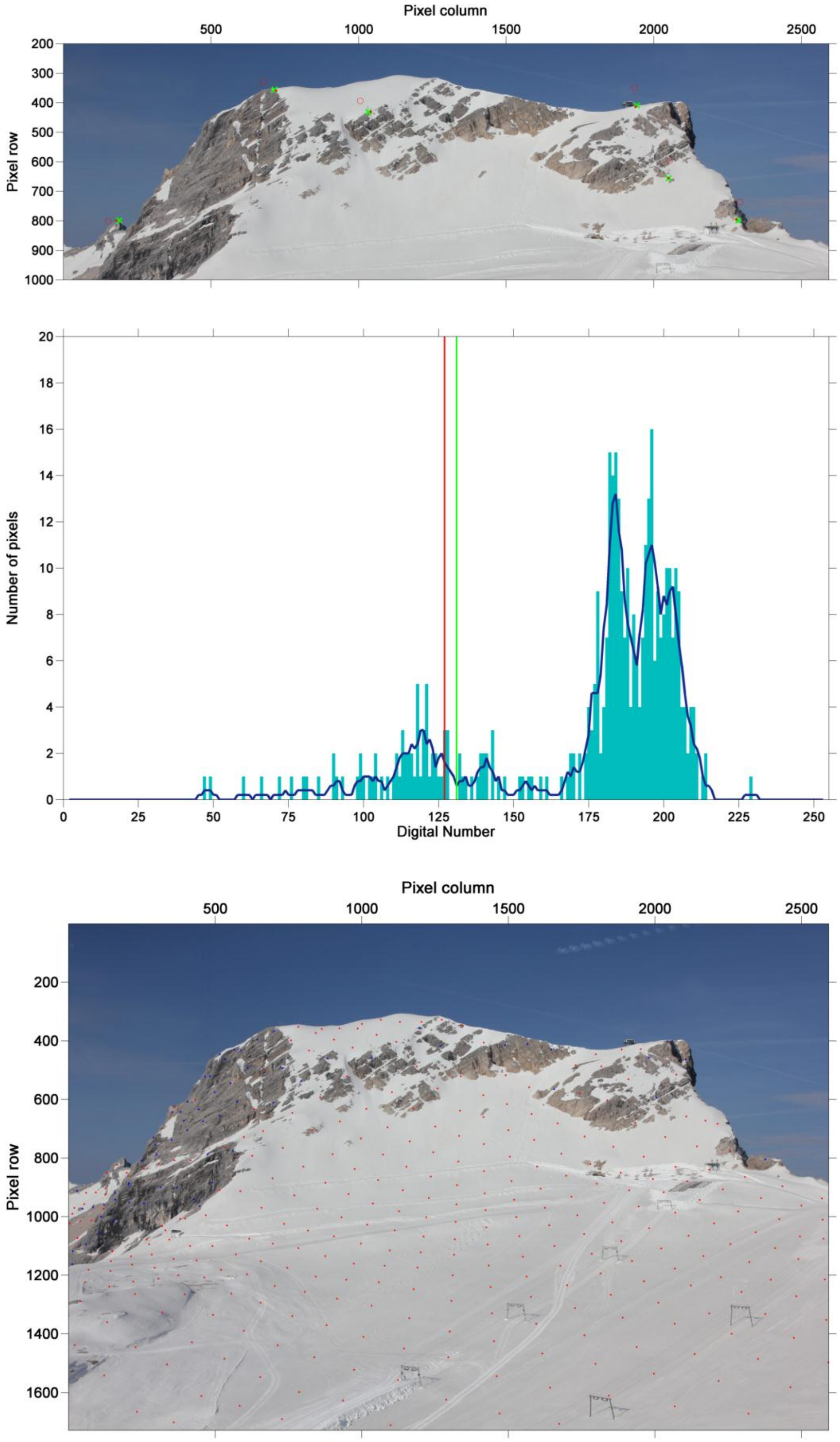



**Figure 3: (a) The extracted area of the photograph (including the GCPs) shows the optimisation of the camera location and its field of view and thus displays the convergence of the projected GCPs, before (red circles) and after (red dots) the optimisation, to the real GCPs (green crosses). (b) The automatic classification procedure is based on a frequency histogram of the DN values in the blue band (cyan bars) that are extracted by the projected DEM pixels. These are further on smoothed (blue line) and the first local minimum greater or equal than a DN threshold of 127 (red vertical line) is depicted as the snow threshold (131, green vertical line). (c) The overlay of the projected DEM pixels and the photograph illustrates the derived snow classification with red dots for snow and blue dots for no snow.**

We want to state here that we normally would recommend using the manual classification procedure in this test case as the automatic classification using a histogram (Fig. 3b) is statistically not very reliable using a coarse spatial resolution of the DEM (30 m). This is due to the small number of 383 DN values that can be extracted of the photograph. Considering that, the visual investigation reveals a quite good quality in the classification result (Fig. 3c).

# Appendix

**Appendix 1: List and description of the variables in the saved Matlab workspace for the test case. Variables that might occur with a different choice of switches are declared in grey.**

| Variable name | Description |
|---|---|
| Azi | Array(2,2): Boundaries of the azimuth viewing angle at $C_o$; clockwise and with north being 0 in radiant (1st column), sector number of the boundaries, clockwise and with the NE sector being 1 (2nd column); vs=true |
| DDS_Max_Eval | Maximum number of optimisation iterations; os=true |
| DDS_R | Neighbourhood perturbation size, i.e. the maximum change of a parameter in one iteration step of the optimisation (in relation to the corresponding parameter range); os=true |
| LB | Vector(1,6): Lower boundary of the seed parameters (X0); os=true |
| N_gcpW | Number of GCPs; gs=true |
| UB | Vector(1,6): Upper boundary of the seed parameters (X0); os=true |
| Vert | Vector(2,1): Boundaries of the vertical viewing angle at $C_o$; from maximum to minimum; horizontal is 0, upwards is positive, downwards is negative in radiant; vs=true |
| X0 | Vector(1,6): Seed parameters before the optimisation consisting of longitude and latitude of $C_o$, cam_off, cam_rol and the longitude and latitude of $T$; os=true |
| Xopt | Vector(1,6): Parameters after the optimisation consisting of longitude and latitude of $C_o$, cam_off, cam_rol and the longitude and latitude of $T$; os=true |
| cam | Array(3,2): longitude, latitude and altitude at $C_o$ in metres (1st column), same at $T$ (2nd column); if os=true, then optimised values, else original values |
| cam_foc | Focal length of the camera lens in metres |
| cam_hei | Height of the CCD of the camera in metres |
| cam_off | Installation height of the camera above the surface in metres |
| cam_orig | Array(3,2): longitude, latitude and altitude at $C_o$ in metres (1st column), same at $T$ (2nd column); os=true |
| cam_rc | Array(3,2): column position, row position of $C_o$ in the DEM grid and altitude of $C_o$ in metres divided by DEM pixel length (1st column), same at $T$ (2nd column); if os=true, then optimised values, else original values |
| cam_rc_orig | Array(3,2): column position, row position of $C_o$ in the DEM grid and altitude of $C_o$ in metres divided by DEM pixel length (1st column), same at $T$ (2nd column); os=true |
| cam_rol | Roll angle of the camera in degree; horizontal is 0, clockwise spans from 0 to +90, anti-clockwise from -90 to 0 |
| cam_wid | Height of the CCD of the camera in metres |
| cs | Switch for the classification method |
| delta_rgb | Maximum of the RGB difference (delta of max(RGB) to min(RGB)) in a "snow" pixel; cs=false |
| demP | Array(2,Number of projected DEM pixels): column and row position of a DEM pixel in the image plane (1st column), same for the next DEM pixel (2nd column), … |
| demW | Array(3,Number of projected DEM pixels): longitude, latitude and altitude of a DEM pixel (1st column), same for the next DEM pixel (2nd column), … |
| fin_demW | Location and name of the DEM file |
| fin_gcpW | Location and name of the GCP file; gs=true |
| fin_image | Location and name of the photograph |
| fin_viewW | Location and name of the viewshed file; vs=false |

| | |
|---|---|
| fout_classW_asc | Name of the derived snow cover map file |
| fout_folder | Location of the output files |
| fout_viewW_asc | Name of the generated viewshed file; vs=true |
| gcpP | Array(2,N_gcpW): column and row position of a GCP in the image plane (1st column), same for the next GCP (2nd column), …; gs=true |
| gcpRMSE_opt | Root mean square error (RMSE) of the projected to the real GCP position after the optimisation; os=true |
| gcpRMSE_orig | Root mean square error (RMSE) of the projected to the real GCP position without optimisation; gs=true |
| gcpW | Array(5,N_gcpW): longitude, latitude, altitude, column and row position of a GCP (1st column), same for the next GCP (2nd column), …; gs=true |
| gs | Switch for the accuracy assessment of $\boldsymbol{C_o}$ and the field of view using GCPs |
| header_W | Vector(6,1): Complete headerlines (names and values) of the DEM file |
| headerv_W | Vector(6,1): Values of header_W |
| is | Switch for the classification of a single or multiple TIFF photographs |
| movavgwindow | Size of the window to calculate a moving average of rgbhist; cs=true |
| name_gcpW | Vector(1,6): Names of the GCPs; gs=true |
| os | Switch for the optimisation of $\boldsymbol{C_o}$ and the field of view using GCPs |
| pix_c | Number of pixel columns in the photograph |
| pix_r | Number of pixel rows in the photograph |
| rgbhist | Histogram of extracted DEM pixels in the blue band; cs=true |
| rgbhistmean | Calculated moving average of rgbhist; cs=true |
| rgbimage | Array(8,Number of projected DEM pixels): row and column position in the DEM grid, column and row position in the image plane, red, green and blue band value in the corresponding photograph pixel and the snow classification of a DEM pixel (1st column), same for the next DEM pixel (2nd column), … |
| thres_b_image | Calculated minimum snow threshold in the blue band searching the first local minimum in rgbhistmean greater than or equal to thres_b_orig, cs=true |
| thres_b_orig | First guess of the minimum snow threshold in the blue band; cs=true |
| thres_rgb | Vector(3,1): Minimum snow threshold in the red, green and blue band; cs=false |
| vs | Switch for the viewshed generation |