DOE/SC-ARM-TR-252

Point Cloud of Cloud Points (PCCP) Value-Added Product Report

DM Romps R Oktem

August 2020

DISCLAIMER

DOE/SC-ARM-TR-252

Point Cloud of Cloud Points (PCCP) Value-Added Product Report

DM Romps
R Oktem
Both at Lawrence Berkeley National Laboratory

August 2020

Work supported by the U.S. Department of Energy,
Office of Science, Office of Biological and Environmental Research

Acknowledgments

Support for this research was provided by Argonne National Laboratory.

Acronyms and Abbreviations

2D	two-dimensional
3D	three-dimensional
ARM	Atmospheric Radiation Measurement
CF	Central Facility
DLT	direct linear transformation
FOV	field of view
GPS	Global Positioning System
PCCP	Point Cloud of Cloud Points
SGP	Southern Great Plains
UTC	Coordinated Universal Time
VAP	value-added product

Contents

Figures

Tables

1.0 Introduction

In this report, we refer to a pair of cameras that capture synchronized pictures with overlapping fields of view (FOV) as a stereo pair. Each stereo pair independently performs a stereo reconstruction of cloud points. Currently, there are three U.S. Department of Energy Atmospheric Radiation Measurement (ARM) user facility stereo pairs (six cameras in total) positioned around the Southern Great Plains (SGP) observatory's Central Facility (CF; Romps and Oktem 2017). Time-synchronized pictures from the two cameras in a stereo pair can be paired together to obtain a three-dimensional (3D) reconstruction of feature points by triangulation. This document explains how we use ARM stereo cameras and stereophotogrammetric principles to generate the Point Cloud of Cloud Points (PCCP) Value-Added Product (VAP). The PCCP VAP is essentially a set of 3D positions representing the locations of cloud features in the sky. Thousands of cloud features can be reconstructed instantly in each stereo pair's FOV, which covers an area of tens of square kilometers. Cloud base and cloud top heights can be extracted from the PCCP product.

2.0 Procedure

Stereo reconstruction is an implementation of projective geometry to estimate the position of an object in 3D space using a pinhole camera model. Figure 1 (Oktem et. al. 2014, reused with permission from ©American Meteorological Society) illustrates this in a simplified two-dimensional (2D) example. The green and the blue squares are the projections of the black square onto two separate image lines. Each image line represents the projection screen of a pinhole camera. The position of the black square can be determined as the intersection point of the green and blue lines that pass through the pinholes. In a more realistic case, the black square has a position in 3D space, the projection screens are planes rather than lines, and the projection screens associated with the separate cameras do not necessarily lie on the same plane. However, similar geometrical principles still hold, and the 3D position of an object can be estimated from its two projections. The general problem of 3D stereo reconstruction is tackled in three basic steps: camera calibration, feature matching, and back-projection (triangulation). Each step is briefly described in the following subsections.

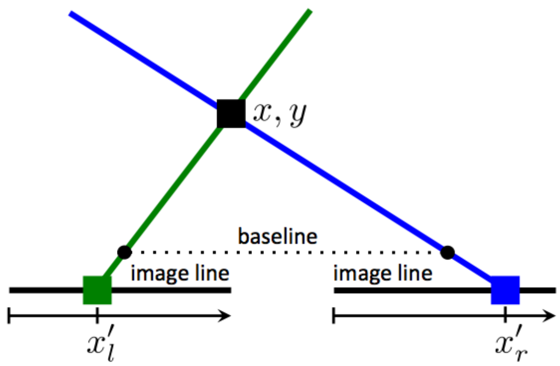

Figure 1. An example of stereo reconstruction in 2D. Each camera is represented by a pinhole (black dot) and a projection screen (black line). The object of interest (black square) projects onto the reference (right) and pairing (left) cameras at the positions of the blue square and green square, respectively. The position of the black square is calculated as the intersection of these two lines (this figure is reused from Oktem et. al. 2014 with permission from © American Meteorological Society).

2.1 Camera Calibration

The mapping from a 3D object to its projection on a camera screen is defined by the camera's intrinsic and extrinsic parameters. Camera calibration is the process of estimating these parameters. The intrinsic parameters are focal length, sensor size, and relative pinhole position with respect to the camera screen, and they are fixed for a specific camera. Intrinsic camera calibration is a well-formulated problem (Hartley et al. 2003) that can be solved by calibration packages such as OpenCV libraries (Bradski et al. 2008). These calibration packages take up-close snapshots of a checkerboard captured at multiple angles as inputs. They estimate the intrinsic parameters by using the relative positions of the true and projected checkerboard corners. Radial distortion parameters related to wide-angle lens optics are also estimated by these packages. Although the theory of stereophotogrammetry is formulated for pinhole cameras, real camera lenses do not behave as pinholes. Wide-angle lenses, in particular, suffer from a significant amount of deviation of the projection from that generated by a pinhole. Each of the ARM stereo cameras is equipped with a wide-angle lens that causes noticeable radial distortion. This distortion can be corrected by inverse mapping once the distortion parameters that are also fixed for a specific camera lens are estimated. Figure 2 displays a snapshot from one of the ARM stereo cameras before (left panel) and after (right panel) distortion correction.

Figure 2. A sample image from the ARM stereo camera at site E45a. The image on the right is corrected for radial distortion.

The extrinsic camera parameters are the three Euler angles (pitch, yaw, roll) and the three translation parameters defining the orientation and location of a camera with respect to a reference Cartesian frame. Therefore, they need to be estimated after a camera is mounted in place. The translation parameters of a camera can be determined by a Global Positioning System (GPS) device. Estimation of Euler angles often relies on at least three external landmarks of known 3D world coordinates. We use the angular position of the brightest stars/planets as the external landmarks. We identify the projection of the star/planet on the image plane (camera screen) on a clear night sky. Sirius, Venus, and Jupiter are the brightest celestial objects that can be identified in the ARM stereo camera images when they are in the FOV. We use a nonlinear optimization method that aims to minimize the sum of squared distances between the observed (on the projection screen) and the calculated (by stereophotogrammetry) projections of star/planet positions to predict the three Euler angles.

External calibration with celestial objects is good at estimating Euler angles within around 0.1 degree accuracy. However, stereo calibration requires that the two cameras within a pair are calibrated with at least 0.01 degree accuracy with respect to each other. Such accuracy is achieved by adding the epipolar constraint to the celestial calibration. Consider the green line in Figure 1. Any object placed on the green/blue line is projected onto the same green/blue square on the left/right screen, but the projections of these objects form a line on the right/left screen. This line is called the “epipolar line” associated with the green/blue square in the left/right image. Each projection in an image has an associated epipolar line in the other pair. Therefore, the projection of an object in the pairing image must lie on the epipolar line associated with the object’s projection in the reference image, and that is the epipolar constraint. More details on how celestial projection and epipolar constraint are formulated for extrinsic camera calibration can be found in (Oktem et al. 2014, 2015). The relations between a 3D point and its projections and the epipolar constraint are also listed in Appendix A.

2.2 Feature Extraction and Matching

Features are distinct visual points in an image. They are often detected using gradient-based approaches. We extract cloud features based on the Canny edge detector that identifies positions of high gradient pixels (Figure 3). This process is performed over the reference camera picture only. Next, we need to find the corresponding matches in the pairing picture. We refer to the camera that is to the right of the baseline as the *reference camera*, and the one that is to the left of the baseline as the *pairing camera*. The matching pixel of each extracted feature is sought on its epipolar line in the pairing camera's picture. We developed a hierarchical block search method to match the features with their pairs in a time-efficient manner. The matching is performed over low-resolution and reduced-size (by a factor of eight in both dimensions) image pairs at first. This significantly reduces the search area and the number of false matches. Then, these initial matches are updated recursively by moving up to a higher-resolution level, until the full resolution images are processed. At each hierarchical level, the matches from the previous level are considered as coarse estimates. They are updated by searching for better matches in a restricted neighborhood of the coarse estimates (instead of full-range search). The measure for a match is the cross-correlation coefficient between the reference and the pairing blocks. A block is composed of a picture's intensity values inside a rectangular region that has the feature point of interest at its center. The best match is the one that outputs the highest correlation coefficient higher than a certain threshold. The feature is discarded (i.e., no match is found) if the highest cross-correlation coefficient stays below the threshold that is set separately for each hierarchical level. In Figure 3, some of the extracted features are marked in blue in the right panel. The matches found by the summarized method are marked in green in the left panel of the same figure.

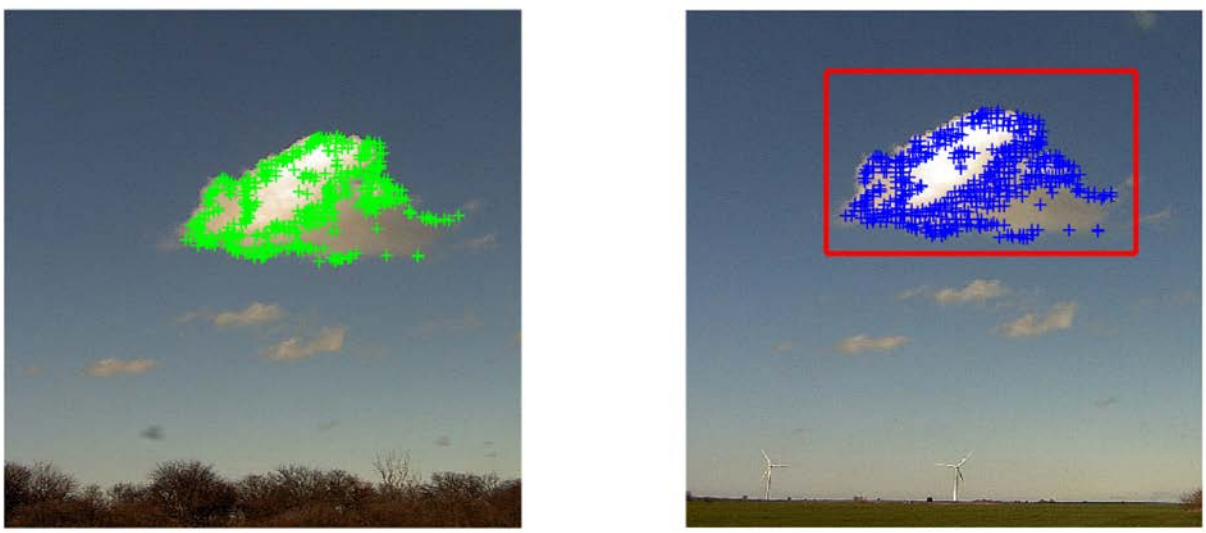

Figure 3. The pictures in the right and the left panels are the cropped samples from the reference and the pairing cameras, respectively, at site E45. The blue marks in the right panel correspond to the detected cloud features within the red box in the reference picture. The green marks in the left panel are the matched pairs of these features.

2.3 Stereo Reconstruction by Back-Projection

In the final step, we use direct linear transformation (DLT) to estimate the 3D positions corresponding to the matched features. DLT is a triangulation method that combines the linear mapping between the 3D coordinates and the 2D matches in the reference and pairing images into the form of a homogeneous linear equation, and solves for the 3D coordinates using singular value decomposition (see Hartley et. al. 2003 for details on derivation and Oktem et. al. 2014 for details on implementation). The reconstructed coordinates corresponding to the features of the cloud in Figure 3 are plotted on a 3D graph in Figure 4.

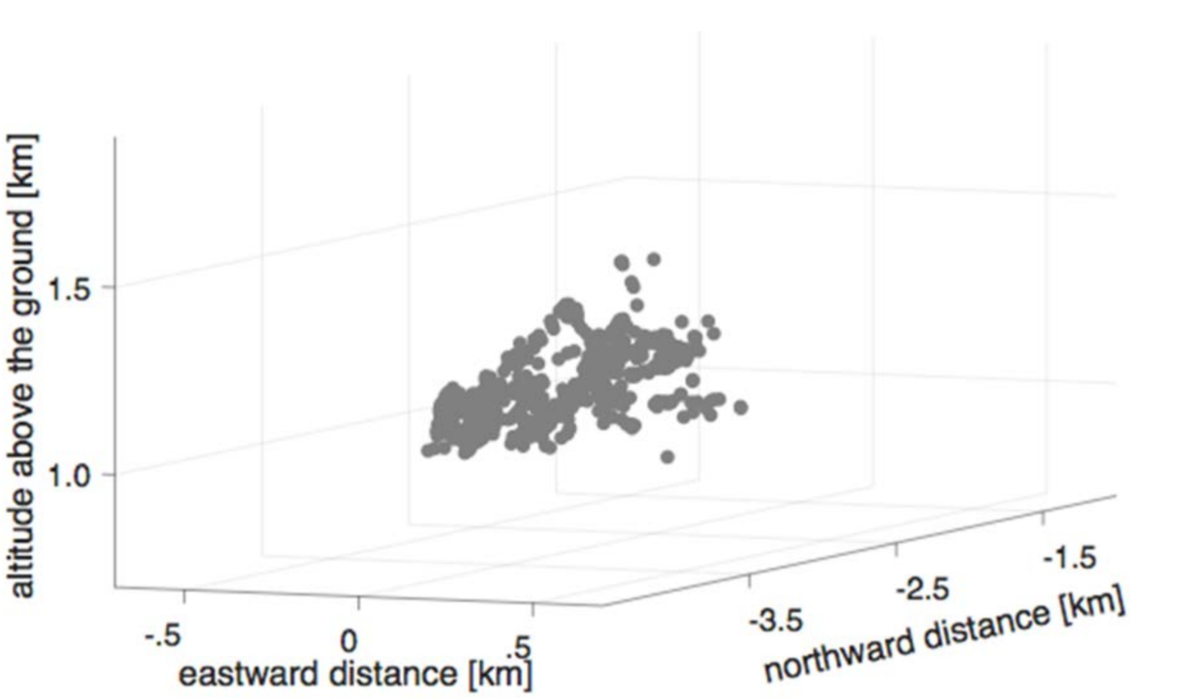

Figure 4. 3D coordinates corresponding to the features shown in Figure 3.

3.0 Software Package

We developed a software package in C++ to implement the stereophotogrammetric cloud point reconstruction described in Section 2. The package uses OpenCV routines for basic processes such as matrix operations and camera distortion correction and it has a modular structure to enable execution and portability of different functions independently. For example, camera calibration is carried out by a standalone module and it generates camera-specific configuration files that are updated as needed. Note that intrinsic camera parameters are not expected to change unless the cameras are replaced, but a change in camera orientation as small as 0.01 degree requires that the extrinsic calibration is updated. Currently, only the PCCP VAP generation module is ported to the ARM system.

3.1 PCCP VAP ALGORITHM

The PCCP VAP generation module executes through the following four main stages for each time interval starting from sunrise until sunset for a day:

- Pre-processing:
  - Distortion correction and image enhancement — Images are read in pairs, corrected for radial distortion, and enhanced by unsharp masking to emphasize the high-frequency features.
  - Cloud mask generation and sun detection — Regions of clouds are extracted from the background and the region of sun (if in FOV) is discarded.
- Feature extraction and matching:
  - Distinctive cloud features are identified inside the cloud mask in the reference image.

  - A match for each feature is sought in the pairing image by using the hierarchical block search method described in Section 2.2.
- Post-processing:
  - A 3D filtering is applied to remove false positives.
- Back-projection — DLT is used to estimate the 3D coordinates corresponding to the filtered matches.

Note that although post-processing substantially reduces the number of false positives, it does not completely eliminate them at all times. The algorithm may report false detections under clear sky but hazy conditions when the sun is close to the horizon and is in the FOV of the camera. However, the number of false detections for such cases is in the order of tens whereas the number of correct positives are in the order of thousands in most cloud conditions. Another point worth mentioning is that the algorithm may not be able to detect any cloud points at all when the clouds do not exhibit any distinct features such as in the case of heavy rain, fog, snow, etc.

3.2 Input and Output Data for the PCCP VAP

The PCCP VAP algorithm reads synchronized images from a stereo pair as input. The input images are tagged as

sgpstereocamaxx.a1.YYYYMMDD.HHMMSS.jpg,

sgpstereocambxx.a1.YYYYMMDD.HHMMSS.jpg,

the letters a and b referring to the reference and the pairing cameras, respectively.

The dimensions and the variables in PCCP VAP datastream are:

Dimensions:

time (UNLIMITED)

camera_a_col = 2592, number of columns of the reference image

camera_a_row = 1944, number of rows of the reference image

Variables:

base_time – Base time in Epoch in seconds since 1970-1-1 0:00:00 0:00

time_offset – Time offset from base_time in seconds since YYYY-MM-DD 00:00:00 0:00

time – Time from midnight since YYYY-MM-DD 00:00:00 0:00

x_relative – Distance to the East in meters relative to the base, function of (time, camera_a_col, camera_a_row)

y_relative – Distance to the North in meters relative to the base, function of (time, camera_a_col, camera_a_row)

z_relative – Height in meters relative to the base, function of (time, camera_a_col, camera_a_row)

camera_b_col – References camera_b x axis to camera_a pixel axes, function of (time, camera_a_col, camera_a_row)

camera_b_row – References camera_b y axis to camera_a pixel axes, function of (time, camera_a_col, camera_a_row)

base_lat – North latitude of the base coordinate in degrees

base_lon – East longitude of the base coordinate in degrees

base_alt – Altitude of the base coordinate above mean sea level in meters

input_images – Names of the processed input stereocamera images

lat – North latitude

lon – East longitude

alt – Altitude above mean sea level in meters.

The **base_lat**, **base_lon**, and **base_alt** variables refer to the position of the base. For ARM cameras situated at the SGP CF, base is selected to be the position of the Doppler lidar as its data is frequently used for comparison. The **lat**, **lon**, and **alt** variables refer to the position of the reference camera, but not to the PCCP data. The GPS positions of the ARM baseline stereo cameras are listed in Table 1.

x_relative, **y_relative**, and **z_relative** are the fundamental variables identifying the positions of the detected cloud points. They are stored in the datastream as a function of time, columns (**camera_a_col**), and rows (**camera_a_row**) of the reference camera. Note that the number of pixels in a camera screen (camera_a_row times camera_a_column) is the maximum number of features that can be detected at any time instance. However, most of the pixels of an image at a time may not correspond to a distinctive feature or may not correspond to a cloud point at all, therefore most of the **x_relative**, **y_relative**, and **z_relative** entries have a value of "-99999", denoting that no cloud point is detected at that pixel. The valid entries of these variables are exhibited in Figure 5 for March 24, 2020. The upper panel presents a heatmap showing the distribution of reconstructed cloud points by time and height using the variable **z_relative**. The lower panels show the distortion-corrected image captured at 20:43:20 UTC on the left and **x_relative** versus **y_relative** (black dots) and the two camera positions (blue and green squares) on the right panel.

Table 1. GPS locations of SGP stereo setups.

Camera site	Latitude (N°)	Longitude (W°)	Altitude (m)
SGP/E43a	36.63704	97.53817	311
SGP/E43b	36.64056	97.53435	310
SGP/E44a	36.63705	97.43015	319
SGP/E44b	36.63360	97.42650	321
SGP/E45a	36.54990	97.47970	317
SGP/E45b	36.54950	97.48550	316

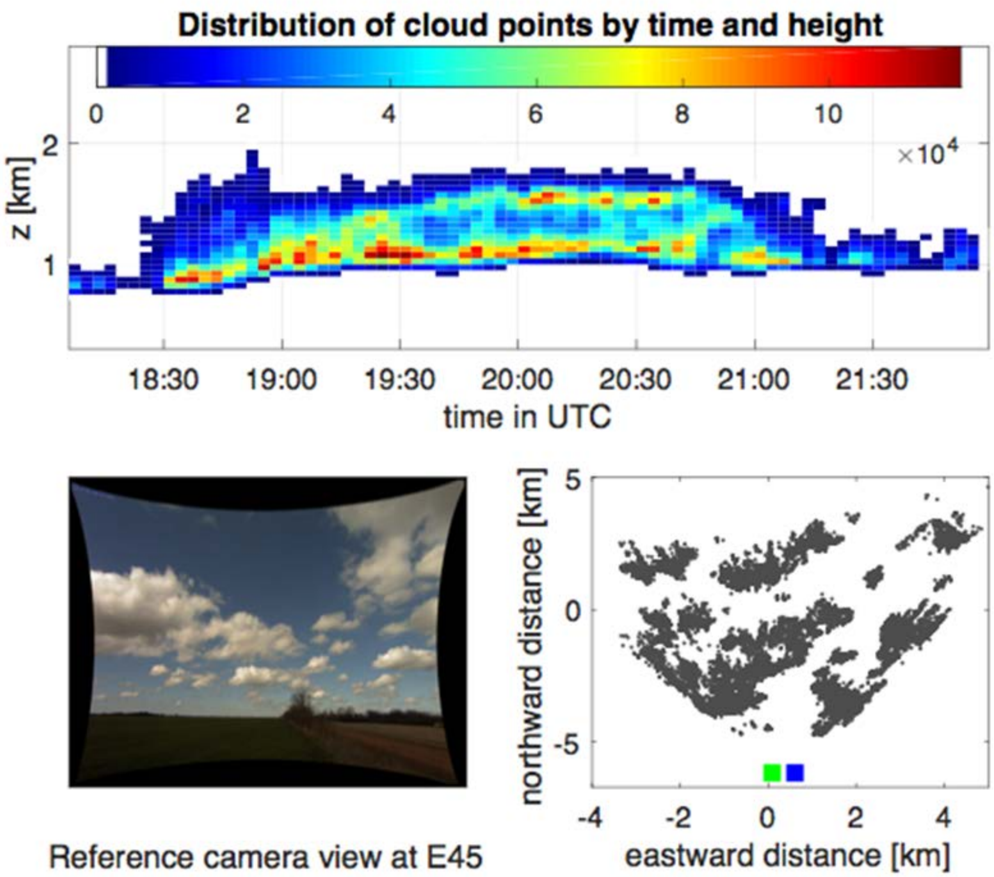

Figure 5. PCCP VAP variables for E45 site on March 24, 2020. The upper panel displays the distribution of the reconstructed cloud points by time and height. The colors represent the count of the detected points in the corresponding bin. The bottom left panel shows the distortion corrected reference camera image at time 20:43:20 UTC. The bottom right panel displays **x_relative** versus **y_relative** for the reconstructed cloud points at 20:43:20 UTC. The blue and the green squares in the bottom right panel represent the reference and the pairing cameras at the site, respectively.

4.0 References

Bradski, G. and A Kaehler. 2008. *Learning OpenCV, Computer Vision with the OpenCV Library*. O'Reilly Media.

Hartley, R, and A Zisserman. 2003. *Multiple View Geometry in Computer Vision*. Cambridge University Press, Cambridge, United Kingdom.

Oktem, R, Prabhat, J Lee, A Thomas, P Zuidema, and DM Romps. 2014. "Stereophotogrammetry of oceanic clouds." *Journal of Atmospheric and Oceanic Technology* 31(7):1482–1501, https://doi.org/10.1175/JTECH-D-13-00224.1

Oktem, R, and DM Romps. 2015. "Observing atmospheric clouds through stereo reconstruction." *Proceedings of SPIE* 9393, https://doi.org/10.1117/12.2083395

Romps, DM, and R Oktem. 2017. Stereo Cameras for Clouds (STEREOCAM) Instrument Handbook. Department of Energy. DOE/SC-ARM-TR-204.

Appendix A

Projection Relations

Let $X = (x, y, z, 1)$ denote the homogenous coordinates of a point in 3D space. Similarly, let $X' = (x', y', 1)$ be the homogeneous coordinates of its projection in the camera screen. For a pinhole camera, the relation between X and X' is expressed as

$$X' = P \cdot X.$$

P is the projection matrix that can be decomposed into camera, rotation, and translation matrices as

$$P = C \cdot R \cdot T,$$

$$C = \begin{bmatrix} fk_x & 0 & c_x \\ 0 & fk_y & c_y \\ 0 & 0 & 1 \end{bmatrix},$$

$$R = \begin{bmatrix} 1 & 0 & 0 \\ 0 & cos\theta_1 & sin\theta_1 \\ 0 & -sin\theta_1 & cos\theta_1 \end{bmatrix} \begin{bmatrix} cos\theta_2 & 0 & -sin\theta_2 \\ 0 & 1 & 0 \\ sin\theta_2 & 0 & cos\theta_2 \end{bmatrix} \begin{bmatrix} cos\theta_3 & -sin\theta_3 & 0 \\ sin\theta_3 & cos\theta_3 & 0 \\ 0 & 0 & 1 \end{bmatrix},$$

$$T = \begin{bmatrix} 1 & 0 & 0 & -x_0 \\ 0 & 1 & 0 & -y_0 \\ 0 & 0 & 1 & -z_0 \end{bmatrix},$$

where $f, k_x(k_y), c_x(c_y)$ are the focal length, number of pixels per distance on the camera screen's $x'(y')$direction, and the principal point coordinate of the camera screen (camera intrinsic parameters), $\theta_1, \theta_2, \theta_3, x_0, y_0, z_0$ are the Euler angles (pitch, yaw, roll) and camera world coordinates (camera extrinsic parameters), respectively.

Let X_r' and X_l' denote the two projections of a 3D point onto the reference and pairing stereo cameras, respectively. The epipolar line equation associated with X_r' is given by the equation

$$e_l = F \cdot X_r',$$

where F is the fundamental matrix (see Hartley et. al. 2003 for derivation of the fundamental matrix). The epipolar constraint dictates that X_l' lies on the epipolar line e_l, therefore

$$X_l'^T F X_r' = 0.$$

Appendix B

Sample Code to Extract Instantaneous PCCP Data

```
#######################################################################################
#Contact roktem@lbl.gov for questions
#Date : June 2020
#This script reads PCCP data for a time instance and displays relevant plots
#Usage: python pccp_snapshot.py <input_nc_filename> <output_plot_filename> <time_instance>
#       e.g. python pccp_snapshot.py sgppccpE45.c1.20190512.130000.nc sample_plot 201100
# reads data from sgppccpE45.c1.20190512.130000.nc and saves the output in sample_plot.png for time
20:11:00

import matplotlib as mpl
import numpy as np
import numpy.ma as ma
import sys
from mpl_toolkits.mplot3d import Axes3D
import matplotlib.pyplot as plt
from netCDF4 import Dataset
import time

plt.rc('xtick', labelsize=14)
plt.rc('ytick', labelsize=14)
plt.rc('axes', labelsize=16)
plt.rc('figure', titlesize=16)

###2D plot of x,y variables in subplot(2,2,si)
def plot2D(x,y,fig,si,xlabel,ylabel):
    ax = fig.add_subplot(2,2,si)
    ax.scatter(x, y, s=1, marker='x', c= 'gray')
    ax.set_xlabel(xlabel)
    ax.set_ylabel(ylabel)
    ax.xaxis.labelpad = 5
    ax.yaxis.labelpad = 5

###3D plot of x,y,z variables in subplot(2,2,si)
def plot3D(x,y,z,fig,si):
    #check if data point count is sufficient for display
    if (len(x)>10):
        ax = fig.add_subplot(2,2,si, projection='3d')
        x = [x[0:len(x)]]
```

```
    y = [y[0:len(y)]]
    z = [z[0:len(z)]]
    x1 = int(min(min(x)))
    x2 = int(max(max(x)))
    y1 = int(min(min(y)))
    y2 = int(max(max(y)))

    ax.scatter(x, y, z, c='gray', marker='o')
    ax.xaxis.set_ticks(np.arange(x1,x2,int((x2-x1+2)/4)+0.5))
    ax.yaxis.set_ticks(np.arange(y1,y2,int((y2-y1+2)/4)+0.5))
    ax.view_init(elev=15, azim=-70)
    ax.set_xlabel('X [km] ')
    ax.set_ylabel('Y [km] ')
    ax.xaxis.labelpad = 15
    ax.yaxis.labelpad = 15
    ax.zaxis.set_ticks(np.arange(0,9,2))
    ax.set_zlabel('Z [km] ')

#filter out values that exceed 50km
def FilterRange(data_in, indnz):
  if (indnz==-1):
    indnz = (abs(data_in) < 50000).nonzero()
  data_out = data_in[indnz]/1000 # convert to km
  return data_out,indnz

##########MAIN#######################################################
#####################################################################`
def main():
  #check input arguments
  if (len(sys.argv)<4):
    print('Usage: python pccp_snapshot.py in_filename out_plot_name time(HHMMSS)')
    print('e.g. python pccp_snapshot.py sgppccpE44.c1.20180420.130000.nc sample 170000')
    sys.exit();

  inFileName = sys.argv[1] #input nc filename
  outFileName = sys.argv[2] #output png filename
  curTimeStr = sys.argv[3] #time to display
  curTime = int(curTimeStr[0:2])*3600+int(curTimeStr[2:4])*60+int(curTimeStr[4:6]) #convert to
integer

  #read input
  try:
    dataset = Dataset(inFileName, 'r', format='NETCDF4')
  except:
    print('Could not open file:', inFileName)

  x_relative_var = dataset.variables['x_relative']
  y_relative_var = dataset.variables['y_relative']
  z_relative_var = dataset.variables['z_relative']
  timeoff = dataset.variables['time'][:]
```

```
    tind = np.array(range(0,len(timeoff)))
    ti = tind[timeoff == curTime ]
    if (len(ti)==0): #if no data is available at the time entered
        dt = abs(timeoff-curTime)
        tn = tind[dt==min(dt)]
        print('No data available at the entered time, the nearest available time is at
',time.strftime("%H:%M:%S", time.gmtime(timeoff[tn])))
    else:
        x_slice,ind_nonzero = FilterRange(x_relative_var[ti][:][:],-1)
        y_slice,ind_nonzero = FilterRange(y_relative_var[ti][:][:],ind_nonzero)
        z_slice,ind_nonzero = FilterRange(z_relative_var[ti][:][:],ind_nonzero)
        print(len(x_slice),' cloud points are extracted')

        fig = plt.figure(figsize=(20,10))
        fig.suptitle('PCCP data from file '+inFileName +' at '+curTimeStr + ' UTC')
        plot2D(x_slice,y_slice,fig,1,'direction eastward [km]','direction northward [km]')
        plot2D(x_slice,z_slice,fig,2,'direction eastward [km]','altitude above the ground [km]')
        plot2D(y_slice,z_slice,fig,3,'direction northward [km]','altitude above the ground [km]')
        plot3D(x_slice,y_slice,z_slice,fig,4)

        fig.savefig('{}.png'.format(outFileName))
    dataset.close()


##########################################################################`
main()
```

ARM

www.arm.gov

U.S. DEPARTMENT OF
ENERGY

Office of Science