ORIENTAL - AUTOMATIC GEO-REFERENCING AND ORTHO-RECTIFICATION OF ARCHAEOLOGICAL AERIAL PHOTOGRAPHS

W. Karel$^{\phi,\psi*}$, M. Doneus$^{\phi,\chi}$, G. Verhoeven$^{\chi,\phi}$, C. Briese$^{\chi,\psi}$, C. Ressl$^{\psi}$, N. Pfeifer$^{\psi}$

$^{\phi}$ Vienna Institute for Archaeological Science, University of Vienna
Franz-Klein-Gasse 1, 1190 Vienna, Austria
$^{\chi}$ LBI for Archaeological Prospection & Virtual Archaeology
Hohe Warte 38, 1190 Vienna, Austria
$^{\psi}$ Department of Geodesy and Geoinformation, Vienna University of Technology
Gusshausstrasse 27-29, 1040 Vienna, Austria

KEY WORDS: aerial photograph, orientation, georeference, archaeology, automation, software

ABSTRACT:

This paper presents the newly developed software OrientAL, which aims at providing a fully automated processing chain from aerial photographs to orthophoto maps. It considers the special requirements of archaeological aerial images, including oblique imagery, single images, poor approximate georeferencing, and historic photographs. As a first step the automatic relative orientation of images from an archaeological image archive is presented.

1 INTRODUCTION

Although aerial photography is one of the oldest archaeological prospection methods, its potential is often not completely understood. While most archaeologists regard this method as highly efficient in detecting sites, its archaeological value goes far beyond: it provides detailed information on the layout and individual features of archaeological and palaeoenvironmental sites and structures. This information, however, can only be used, when the photographs are georeferenced, (ortho) rectified, interpreted and mapped. Georeferencing assigns spatial information to the imagery and explicitly defines their location and rotation with respect to a certain Earth-fixed coordinate frame and is a requirement to transcribe the relevant archaeological information content of an aerial photograph to a map.

Irrespective of the method applied, the georeferencing of the images is commonly determined with ground control points (GCPs), whose measurement and identification is a time-consuming operation that is only partly shortened in photogrammetric packages by automatic tie point detection and bundle block adjustments of multiple images. Especially in large-scale archaeological projects with thousands of images, this is currently considered to be the main bottleneck in terms of personnel and temporal resources. Additionally, specialised software, photogrammetric skills, and experience are required. Therefore, aerial archaeology does often not go beyond the data acquisition stage. As a result, millions of aerial photographs (APs) are currently stored in archives that often only allow a very limited geometric access of the imagery. These archives contain an enormous amount of valuable optical information for detecting and documenting archaeological landscapes which cannot be exploited efficiently.

Hence automated techniques for the georeferencing of archaeological APs are highly desirable. A number of approaches for the automatic, relative orientation of image blocks have been suggested (Pollefeys et al., 2004, Mayer, 2005) and a number of software packages for this task are available - e.g. Bundler (Snavely et al., 2006), Photoscan Professional (Agisoft, 2013), Airphoto SE (Irwin Scollar, 2013), and Apero (Pierrot Deseilligny and Clery, 2011). While the methods are partly overlapping, our contribution is especially

1. to design a system in which additional observations, beyond image coordinates and control points, can be integrated,
2. to provide an interface to Python which allows the assembly of custom workflows in scripts,
3. to deliver the estimated accuracy of the unknown parameters (exterior orientation of images, tie point coordinates, etc.),
4. to link it to an archaeological image archive and thus provide a tailored processing for archaeologists, and
5. to automate the process of absolute orientation (datum definition) based on orthophotos and terrain models.

While 1. and 2. are demonstrated in this article, 3., 4. and 5. are only shown in the planned version.

1.1 The Project

These tasks are researched within a project, which is supported by the Austrian Science Fund (FWF) and which started in March 2012. Three international partner organisations (Vienna Institute of Archaeological Science of the University of Vienna, LBI for Archaeological Prospection and Virtual Archaeology based on an international partnership, and the Research Groups of Photogrammetry and Remote Sensing of TU Vienna) are co-operating within the project. The project is designed specifically for APs made during archaeological reconnaissance flights. The list given above thus has to be specialized to consider the following:

- ✓ both vertical and oblique images are acquired,
- ✓ different non-metric (uncalibrated) cameras are used,
- ✓ oblique APs often lack a high number and/or good distribution of possible ground control points,
- ✓ oblique APs are often shot from an extremely oblique viewpoint, and
- ✓ often even a rough geocoding is missing (e.g. position information by a hand-held GNSS (global navigation satellite system) receiver).

*Corresponding author. wilfried.karel@geo.tuwien.ac.at

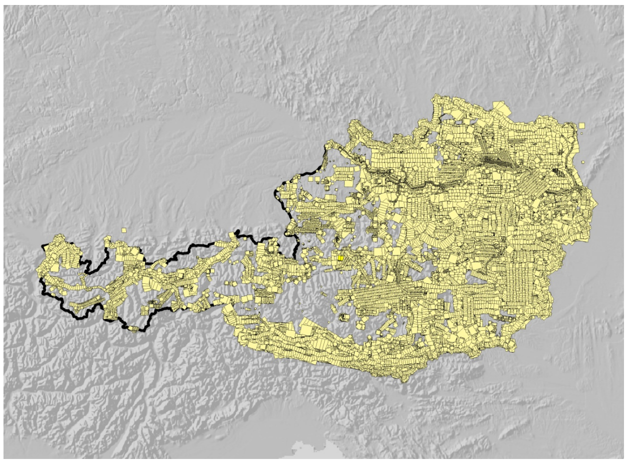

Figure 1: Map of the approximately determined footprints of more than 110,000 APs stored at the aerial archive of the Department for Prehistoric and Medieval Archaeology in Vienna.

The aerial archaeology unit at the Department for Prehistoric and Medieval Archaeology of the University of Vienna (UFG) stores more than 110,000 vertical and oblique APs in its aerial archive (Doneus et al., 2001) , with so far roughly 60% of them digitised with high resolution. These will be used as photographic resource. The metadata of all photographs including their footprints are stored in a GIS-based database (see fig. 1).

2 METHOD

2.1 Automatic relative orientation of images

Given a set of APs, our approach for automatic orthophoto generation first co-registers the APs to each other. We detect scale invariant keypoints (Lindeberg, 1993) in every image and generate feature vectors that describe their local neighbourhood (Lowe, 2004). These descriptors are inserted into appropriate search trees for efficient nearest-neighbour lookup (Muja and Lowe, 2009).

Subsequently, for every possibly image pair in the set, we iterate over the keypoints of one image, and select as the corresponding keypoint in the other image the one with the most similar descriptor, where similarity is defined as the reciprocal of the 2-norm of descriptor difference vectors. Still in descriptor space, these correspondences are validated: by reversal of the search direction, they are tested on symmetry, and they are dropped otherwise. Also, the ratio of the similarities of corresponding descriptors and respective second most similar descriptors must satisfy a certain threshold. Finally, keypoint scales must not differ by a too high value.

In the following, two graph structures are used to tackle combinatorial problems:

- *Keypoint tracking*: introduce the keypoints of all images as vertices into a graph, and insert an edge for each correspondence - thereby creating connected components of transitively corresponding keypoints that are projections of the same object point (Moulon and Monasse, 2012).

- *Image connectivity*: introduce all images as vertices into another graph, and link them with a weighted edge, where edge weights represent the expected quality of an image pair's reconstructed relative orientation, defined as the product of the number of corresponding keypoints in the resp. image pair and a measure of perspective distortion (Pollefeys et al., 2004). Initially, in this graph all images are marked as *unoriented*, and all edges are marked as *untried*.

Now, we select as the initial image pair the heaviest, untried edge that connects two unoriented images. Based on the correspondences of the image pair, we compute its relative orientation via the essential matrix, by a RANSAC version (Fischler and Bolles, 1981) of the 5-point-algorithm (Nistér, 2003) and subsequent robust bundle block adjustment. In this step, camera calibrations are used that are found either in a resp. data base, or are derived from Exif (exchangeable image file format) image meta data. If relative orientation fails, then we mark the edge correspondingly, and repeat this step from the start, until an image pair has been successfully oriented.

Subsequently, we select as the current image the unoriented image with the most observations of already reconstructed object points. We compute its orientation by a RANSAC version of spatial resection. Next, we insert observations for the correspondences of the current image into the bundle block of oriented images, together with orientation parameters for the newly oriented image and eventual new object points. Also at this point, we introduce observations and parameters for eventual untried or failed edges between other already oriented images and the newly oriented one. Moreover, we now search for additional correspondences, this time limiting the search space by maximum image residuals. Finally, we mark the newly processed edges as used, and do a robust adjustment of the bundle block of oriented images, after which the residuals are inspected, and seemingly erroneous observations are deactivated. We repeat this step until all images have been oriented.

Finally, a non-robust adjustment is conducted, which estimates interior orientations and distortion parameters for every combination of camera/lens make and nominal focal length present in the block.

2.2 Orthophoto generation

In addition to the archaeological APs to be georeferenced and ortho-rectified, our approach for automatic orthophoto production relies on two additional data sets: pre-existing orthophoto maps to extract GCPs from, and pre-existing DSMs / DTMs (digital surface / terrain models) to compute terrain heights for these GCPs.

In the search for correspondences between the block of oriented images and the pre-existing orthophoto, we limit the search space as far as possible, using available meta information of the images. This meta information may e.g. have been observed directly, or it may be known from the flight plan. For example, an approximative georeferencing can be derived from given, coarse footprints of the images of an extended block, together with a given (constant) flight height. Using a DSM / DTM, terrain relief is considered, and in order for correspondences to be accepted, they need to be detected in multiple images, with reasonable residuals.

If the resolution of the pre-existing DSM / DTM is not sufficient for the generation of the new orthophoto, then at this point, we generate a dense DSM based on the aerial images, using semi-global matching (Hirschmüller, 2008).

Finally, we ortho-rectify all APs and produce an orthophoto map by mosaicking the outcome. Our method for automatic orthophoto generation is depicted in fig. 2.

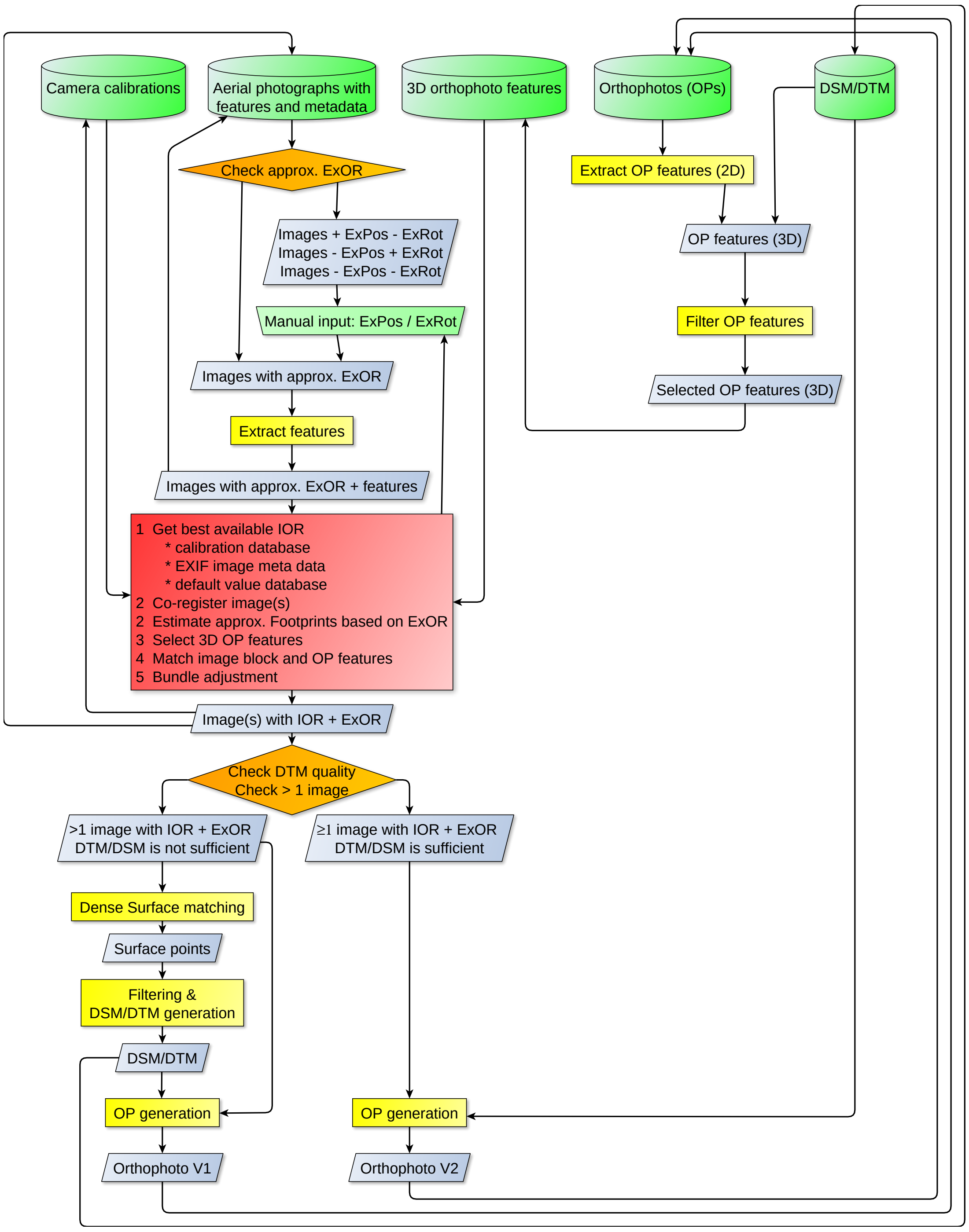

Figure 2: Flowchart of all processing steps needed for automatic orthophoto production.

2.3 Discussion

As explained above, we georeference in two steps a set of APs that belong to the same flight, where first, the APs are co-registered to each other, and only afterwards, the resulting block is georeferenced as a whole, using a pre-existing orthophoto and a DSM / DTM. The reason why we do so is that the aerial images have been captured with the same equipment within a short period of time and similar flight height, while the pre-existing orthophoto is probably based on imagery of different characteristics, making the search for correspondences more error prone. However, coarse image positions known beforehand can serve to estimate orientations and limit the search space if the image block has been co-registered already. Also, correspondences can immediately be checked not only in descriptor space, but also concerning geometry.

In the validation of correspondences in descriptor space, we introduce a threshold on the ratio of keypoint scales. We adapt this threshold to accommodate to different flight heights, focal lengths and maximum nadir angles.

The approach described above assumes that meaningful initial values for camera interior orientations are available or can be derived, with the minimum requirement being that the approximate focal length is known beforehand. Otherwise, we estimate the fundamental matrix instead of the essential matrix, and in the transition from projective to metric space, we assume that all images share the same interior orientation. If the terrain features enough relief, then separate interior orientations can still be estimated for each image in the bundle block adjustment.

Also, the approach is tailored to digital images. For historic imagery captured with photographic film, an additional step can be introduced, which detects fiducial marks and transforms the image coordinates into the camera frame.

So far, we use image pairs for the computation of initial relative orientations, which limit geometrically valid correspondences to a narrow band around the resp. epipolar line. Relatively oriented image triples define two epipolar lines for each image correspondence, which generally intersect in an image point. Thus, usage of image triples allows for a more robust outlier detection and hence, a more robust initial relative orientation. However, it also introduces a higher level of complexity concerning computation times and implementation.

In our approach, we propose the usage of point-like features. However, for the search of correspondences between the aerial image block and the pre-existing orthophoto, usage of feature regions seems to be promising.

3 STRUCTURE

We develop OrientAL in two computer languages: C/C++ and Python, and we choose the language of implementation according to the task needed to solve: C/C++ for low-level, performance-critical functionality, and Python for high-level functions. Implementations in Python feature the benefits of being easily readable, debuggable and adaptable by end-users (for study, experimentation, or due to special needs) on the one hand, and of the large amount of high-level functionality (e.g. for visualization) available in Python on the other hand. Binding C/C++ libraries to Python introduces additional development costs, but Python bindings provide an easily accessible way for end-users to integrate OrientAL into custom scripts.

Fig. 3 depicts the basic software structure of OrientAL in its current state. As can be seen, OrientAL contains the central adjustment module *oriental::adjust* and its Python-binding *oriental.adjust*, which is based on the non-linear least-squares adjustment library Ceres (Agarwal and Mierle, 2013) that provides low-level administration of observations and parameters, automatic and numeric differentiation, and a common interface to various numerical solvers. In addition to the definition and solution of non-linear least-squares problems, *oriental.adjust* provides parameter qualities and the option to define callback functions that are invoked on each iteration, for the evaluation of custom cost- and loss functions, or for logging. We use NumPy (Jones et al., 2001–) arrays as the Python type of vectors and matrices, which *oriental.adjust* by help of Boost.NumPy (Daftery, n.d.) translates to C/C++ references that are passed to *oriental::adjust*.

Listing 1 shows a basic example of how to use *oriental.adjust* to estimate the exterior orientation of a single image, based on image observations of control points. After the necessary imports and the initialisation of parameters and observations, it creates an instance of an adjustment problem and adds to it a residual block for each image point observation, passing a cost function holding the observations and the definition of residual computation, an instance of the squared loss function, and the concerned parameters. As only the exterior orientation shall be estimated, the other parameters need to be set constant before solving the problem.

Most of the other functionality currently is bundled in high-level Python scripts that directly call the Python bindings of third-party libraries. OpenCV (Bradski, 2000) is used for feature point detection, description and matching, and for two-view algorithms (relative orientation of image pairs, forward intersection of object points). Additionally, we use GDAL (Warmerdam, 2008) for accessing image meta data, and SpatiaLite (Furieri, 2013) as spatial data base.

We plan to transcribe OrientAL Python code to C/C++ in cases where functionality turns out to be performance critical. As can be seen in fig. 3, all but the Python visualization modules are (also) accessible from C/C++. Thus, transcription of OrientAL code from Python to C/C++ is possible without changing the usage of third-party libraries.

```python
# imports
from numpy import array
from oriental import adjust as adj
from oriental.adjust import parameters as pars
# parameters: rotation angles
angles = pars.EulerAngles(
             parametrization=adj.EulerAngles.omfika,
             array=array([.1, -.02,.3]) )
# parameters: projection centre
prjCtr = array([ 0., 0., 10. ])
# parameters: interior orientation
ior    = array([ 150., -150., 100. ])
# parameters: distortion
adp    = pars.ADP( normalizationRadius=75. )
# parameters: object points
objPoints = array([ [  10.,  10., 0. ],
                    [ -10.,  10., 0. ],
                    [ -10., -10., 0. ],
                    [  10., -10., 0. ] ] )
# noisy observations: image points
obs = array([ [ ior[0]+100.+.432, ior[1]+100.+.472 ],
              [ ior[0]-100.-.654, ior[1]+100.+.581 ],
              [ ior[0]-100.+.589, ior[1]-100.+.240 ],
              [ ior[0]+100.-.487, ior[1]-100.+.541 ] ])

# squared loss
loss = adj.loss.Trivial()
# setup the adjustment problem
problem = adj.Problem()
for i in range( objPoints.shape[0] ):
    problem.AddResidualBlock(
      adj.cost.Photo( imgObs[i][0], imgObs[i][1] ),
      loss,
      prjCtr,
      angles,
      ior,
      adp,
      objPoints[i] )
    problem.SetParameterBlockConstant( objPoints[i] )

problem.SetParameterBlockConstant( ior )
problem.SetParameterBlockConstant( adp )
options = adj.Solver.Options()
summary = adj.Solver.Summary()
# solve!
adj.Solve( options, problem, summary )
```

Listing 1: A simple least-squares adjustment in Python using *oriental.adjust*.

4 APPLICATION

Fig. 4 shows a bundle of relatively oriented archaeological APs depicted as tetrahedrons with coordinate axes, and the reconstructed, sparse cloud of object points. The final bundle block consists of 35 APs, 70566 image observations, and 30137 object points. Processing has been fully automatic. The APs image the civil amphitheatre of Roman *Carnuntum*, Austria, and were taken through the side windows of an aircraft, using a digital amateur camera. As mentioned in the introduction, OrientAL is not yet feature complete, and thus, no georeference is shown.

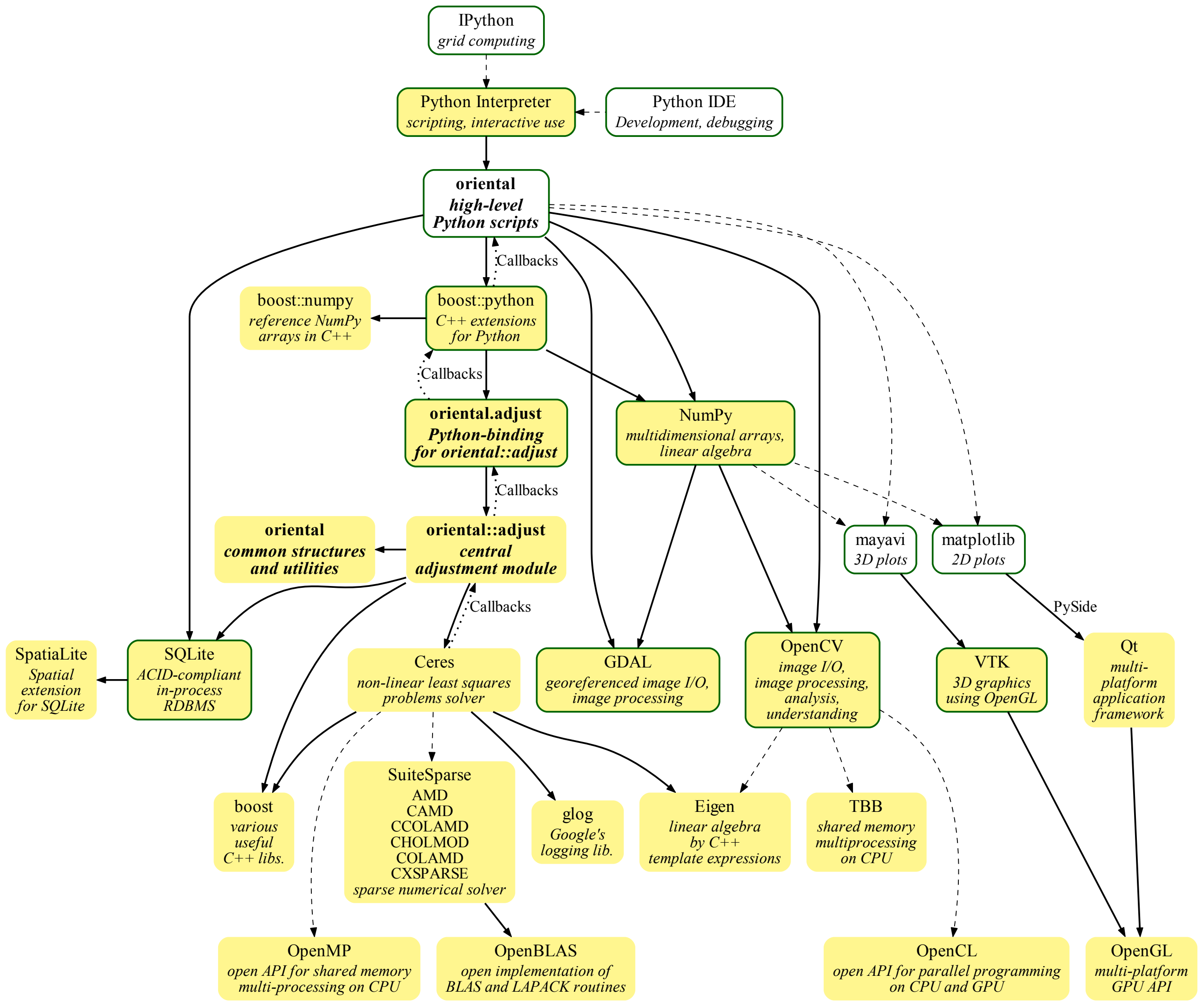

Figure 3: Call graph of modules involved with OrientAL; green frame : modules accessible from Python; yellow fill : modules accessible from C/C++; dashed edges: optional modules;

5 CONLUSIONS AND FUTURE WORK

An **archaeological case study** will serve to test and evaluate all proposed solutions for automated orthophoto generation. The proposed 250 km^2 large area in the eastern part of Lower Austria seems to be very promising both archaeologically and in terms of software testing. It is largely covered by vertical and oblique APs from various years and seasons, with scales ranging from 1:25,000 to 1:5,000. The evaluation should identify problems and improve our approach, both using conventional aeroplanes as well as UAVs.

We firmly believe that the expected results will be of high importance for the archaeological discipline and beyond. By providing archaeologists with easily implementable methods for aerial photo rectification, the enormous amount of valuable information on archaeological landscapes will be accessible to mapping. This could have a dramatic effect on the future archaeological process, where accurate, detailed and coherent GIS-based maps of whole regions can add important information for directing archaeological research and cultural heritage management.

ACKNOWLEDGEMENTS

This research is being carried out with the financial support of the Austrian Science Fund (FWF): P24116-N23.

The Ludwig Boltzmann Institute for Archaeological Prospection and Virtual Archaeology (`http://www.archpro.lbg.ac.at/`) is based on an international cooperation of the Ludwig Boltzmann Gesellschaft (A), the University of Vienna (A), the Vienna University of Technology (A), the Austrian Central Institute for Meteorology and Geodynamics (A), the office of the provincial government of Lower Austria (A), Airborne Technologies GmbH (A), RGZM-Roman-Germanic Central Museum Mainz (D), RA-Swedish National Heritage Board (S), IBM VISTA-University of Birmingham (GB) and NIKU-Norwegian Institute for Cultural Heritage Research (N).

REFERENCES

Agarwal, S. and Mierle, K., 2013. Ceres Solver: Tutorial & Reference. Google Inc.

Agisoft, 2013. Agisoft Photoscan Professional Edition. `http://www.agisoft.ru/products/photoscan/professional/` Accessed on 09 April 2013.

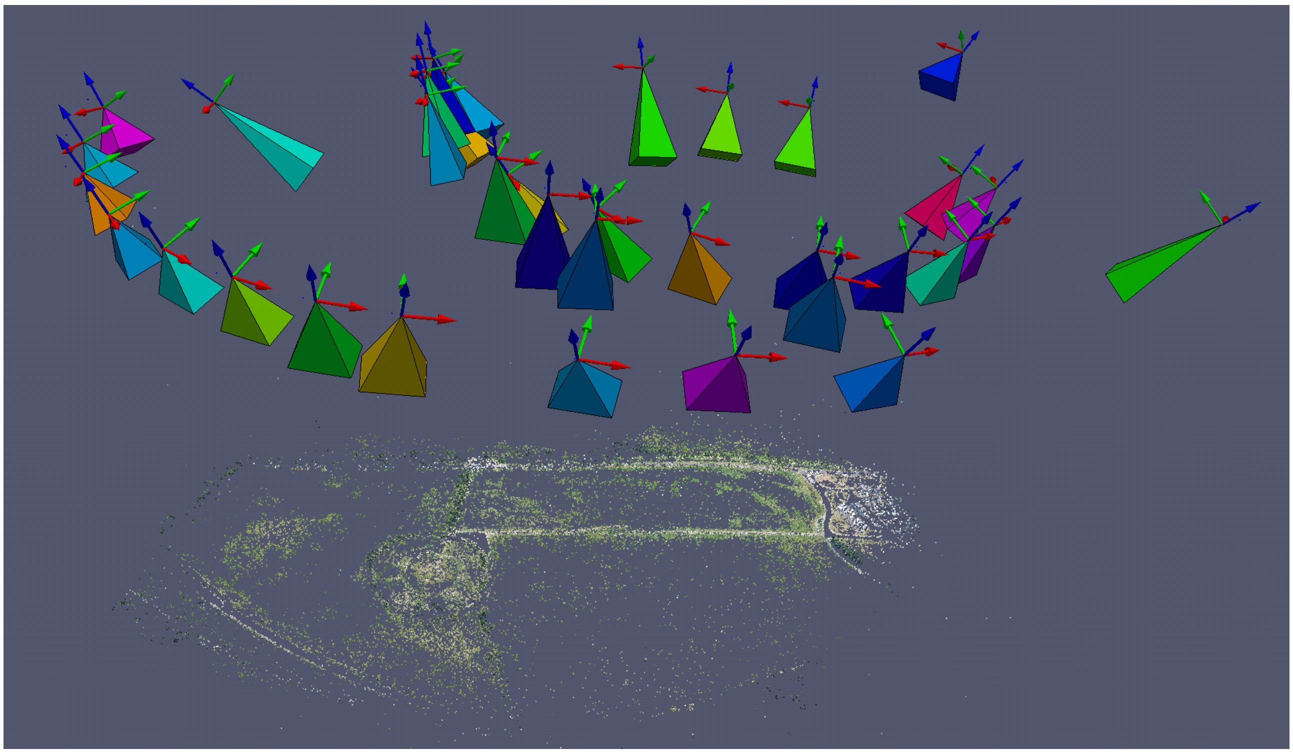

Figure 4: Automatically oriented archaeological APs of Roman *Carnuntum*, Austria, together with the reconstructed, sparse point cloud.

Bradski, G., 2000. The OpenCV Library. Dr. Dobb's Journal of Software Tools.

Daftery, A., n.d. Boost.NumPy. `https://github.com/ndarray/Boost.NumPy` Accessed on 09 April 2013.

Doneus, M., Eder-Hinterleitner, A. and Neubauer, W., 2001. Archaeological prospection in austria. In: M. Doneus, A. Eder-Hinterleitner and W. Neubauer (eds), 4th International Conference on Archaeological Prospection, Austrian Academy of Sciences Press, Vienna, Austria, pp. 11–34.

Fischler, M. A. and Bolles, R. C., 1981. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. Communications of the ACM 24(6), pp. 381–395.

Furieri, A., 2013. SpatiaLite. `https://www.gaia-gis.it/fossil/libspatialite/index` Accessed on 09 April 2013.

Hirschmüller, H., 2008. Stereo processing by semiglobal matching and mutual information. Pattern Analysis and Machine Intelligence, IEEE Transactions on 30(2), pp. 328–341.

Irwin Scollar, 2013. AirPhoto SE. `http://www.uni-koeln.de/~al001/airphotose.html` Accessed on 09 April 2013.

Jones, E., Oliphant, T., Peterson, P. et al., 2001–. SciPy: Open source scientific tools for Python. `http://www.scipy.org/` Accessed on 09 April 2013.

Lindeberg, T., 1993. Scale-space theory in computer vision. Kluwer Academic Print on Demand.

Lowe, D. G., 2004. Distinctive image features from scale-invariant keypoints. International journal of computer vision 60(2), pp. 91–110.

Mayer, H., 2005. Robust least-squares adjustment based orientation and auto-calibration of wide-baseline image sequences. In: International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences XXXVI (Part 3/W36), Beijing, China.

Moulon, P. and Monasse, P., 2012. Unordered feature tracking made fast and easy. In: proc. 9th European Conference on Visual Media Production.

Muja, M. and Lowe, D. G., 2009. Fast approximate nearest neighbors with automatic algorithm configuration. In: International Conference on Computer Vision Theory and Applications (VISSAPP'09), pp. 331–340.

Nistér, D., 2003. An efficient solution to the five-point relative pose problem. In: Computer Vision and Pattern Recognition, 2003. Proceedings. 2003 IEEE Computer Society Conference on, Vol. 2, IEEE, pp. II–195.

Pierrot Deseilligny, M. and Clery, I., 2011. Apero, an open source bundle adjusment software for automatic calibration and orientation of set of images. In: International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences XXXVIII (Part 5/W16), Trento, Italy.

Pollefeys, M., Van Gool, L., Vergauwen, M., Verbiest, F., Cornelis, K., Tops, J. and Koch, R., 2004. Visual modeling with a hand-held camera. Int. J. Comput. Vision 59(3), pp. 207–232.

Snavely, N., Seitz, S. M. and Szeliski, R., 2006. Photo tourism: exploring photo collections in 3d. ACM Trans. Graph. 25(3), pp. 835–846.

Warmerdam, F., 2008. The Geospatial Data Abstraction Library. In: G. B. Hall and M. G. Leahy (eds), Open Source Approaches in Spatial Data Handling, Advances in Geographic Information Science, Vol. 2, Springer Berlin Heidelberg, pp. 87–104.