Oblique aerial photography viewer and monoplotting tool for building façade inspection and disaster assessment

Arnadi Murtiyoso
Department of Topography
INSA Strasbourg, France
arnadi.murtiyoso@insa-strasbourg.fr

Abstract

Aerial photography has a long history of being employed for mapping purposes due to some of its main advantages, including large scope of area and minimization of field work. Moreover, nowadays UAVs may be deployed to address problems which require a quick response, including in the case of damage assessment in the aftermath of a disaster. Multi-camera oblique imagery is a system of aerial photography data acquisition where it captures not only the conventional nadir images, but tilted images as well in the same time. In this paper, a particular use of such imagery in the field of building façade inspection as well as disaster management will be addressed. The main idea is to be able to inspect a building from four cardinal directions by simply clicking on it on a nadir image containing the same object. The application uses Matlab language and its GUIDE module to implement the user interface. It uses the collinearity condition to establish the relation between the camera and the image and object spaces. In addition, efficient monoplotting tools have also been developed and attached to the main application for the purpose of building height determination, distance measurement, and digitization from the oblique images. The resulting Graphical User Interface (GUI) is capable of identifying a building from several oblique points of view, as well as calculating the approximate height of buildings, ground distances, and perform basic vectorization procedures. The geometrical accuracy remains a function of the image resolution and user expertise while several useful tools and features should still be added to improve the overall performance of this application. This program is a work in progress conducted during an internship on the research topic "Oblique Aerial Cameras" at the 3D Optical Metrology research unit of FBK Trento, Italy.

Keywords: oblique imagery, façade inspection, disaster management, monoplotting

Introduction

Photogrammetry has been a standard tool often used for large scale mappings. It provides the user with planimetric as well as altimetric data from the photographed area, and eventually its orthophotos. It has also seen increasing use for smaller scale mapping with the rise of UAVs (Unmanned Aerial Vehicles) and low cost cameras. This UAV photogrammetry has proved to be a versatile and useful tool in fast response jobs, such as that during a disaster with the main objective of creating an up-to-date map in a short period of time.

However, damages on buildings which occur in certain highly urbanized areas are harder to determine due to the existence of edifices which may be uniform in some cities. Damages in these areas tend to manifest in the building façades. Since conventional photogrammetry relies, or focuses rather, on nadir images, these damages are harder to determine. In this regard, oblique images can provide measurements and mapping with better speed of compilation and lack of the fieldwork needed as important advantages (Höhle, 2008).

State of the art

The idea of a multi-camera oblique imagery is to set a system with multiple cameras. There exist several commercial systems which provide this kind of imagery, such as Midas 5 and Vision Map A3 (Rupnik et al., 2014). The first camera is always directed towards the nadir, as in conventional photogrammetry. Meanwhile, the other cameras are tilted to a certain degree as to create oblique imagery (Gerke et al., 2014).

Following Rupnik et al. (2014), there are three types of multi-camera oblique imagery system available:

1. Fan configuration, with twin cameras which extend the cross-track ground coverage. (e.g. Trimble AIC x2, VisionMap, Dual DigiCAM);
2. Maltese cross configuration, with one single nadir camera and four cameras tilted towards cardinal directions (e.g. IGI, Midas TRACK' AIR, Leica, Vexcel, etc.);
3. Block configuration (e.g. Trimble AIC x4, IGI DigiCam Quattro, etc.).

A fully automatic image orientation and dense matching for oblique images is currently an interesting topic of research, as it will speed up certain procedures such as city modeling considerably. However, this paper will address another application of the oblique imagery, which is to determine damages on building façades in the aftermath of a disaster as well as building inspection. The main aim of this paper is to create a viewer which can be used to quickly assess the condition of objects on ground taken by oblique cameras, and perform a few basic measurements with the help of monoplotting concepts.

Monoplotting refers to a photogrammetric system where single oblique and unrectified photographs or aerial nadir images are related to the Digital Elevation Models (DEM) of the corresponding real world (Bozzini et al., 2012). In this regard, the intersection between the land surface (modeled by the DEM) and the ray from the camera will be the object's position in 3D object space. This is unlike the more common stereo-photogrammetry where we intersect the rays emitted from two or more cameras to determine this position (Fig. 1). This method is useful, for example when we need to quickly measure points from a single available photo.

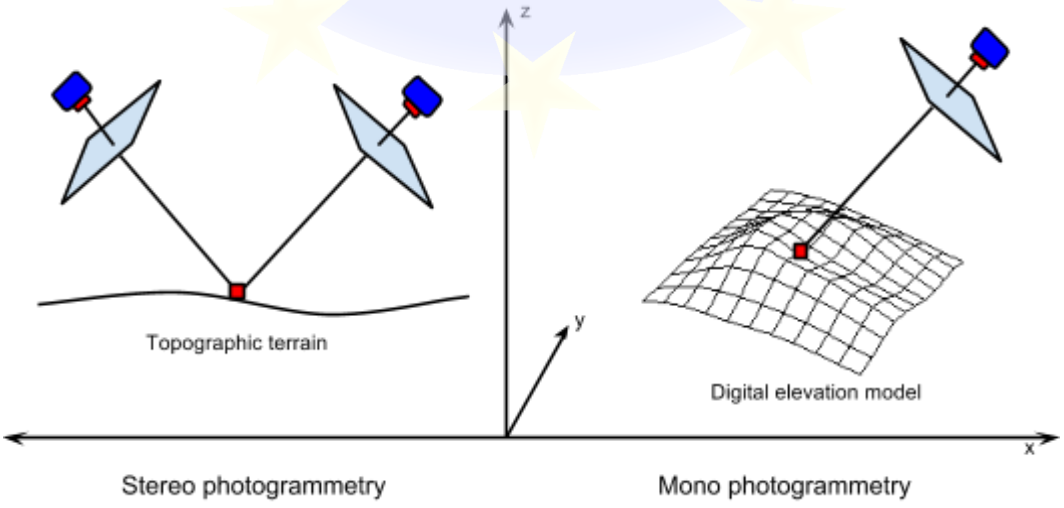

Figure 1. Principles of stereo and mono photogrammetry.

Address Offices in Brussels : Rue du Nord 76, BE – 1000 Bruxelles. Tel +32/2/217.39.72 Fax +32/2/219.31.47
E-mail: jean-yves.pirlot@clge.eu - www.clge.eu EC-Register of interest representatives 29077535421-69

In this paper several monoplotting concepts are implemented. Some simple algorithms to determine the point's object space coordinates from a single image are further developed into three tools which may help the user in analyzing the images as well as performing simple measurements.

Description of application

The data used for experimental purposes in this application is oblique and nadir imageries obtained from a modified Midas 5 system called BlomOblique with Canon EOS 5D Mark II cameras. Two data sets were used, one over the city of Milan for demo and another one over the town of San Felice sul Panaro (Italy) which was hit by an earthquake in May 2012.

The implemented viewer is meant to be a simple interface where the user can click on the nadir image (Cam 1) and to be immediately directed towards the oblique images (Cams 2 through 5) of the respective point. It is developed using the Matlab language through its Graphical User Interface (GUI) module, GUIDE. The principal inputs for this program are:

1. Aerial photos, both nadir and oblique in .jpeg format
2. Initial/approximate exterior orientation (EO) parameters obtained from the platform through GNSS and IMU (Remondino et al., 2014) for all photos in .txt format
3. Internal orientation parameters (IO) obtained from camera calibration (Remondino et al., 2014) performed through the software APERO in .xml format
4. Average ground height (in meter) which we will call Z
5. Pixel size of the camera's CCD (in millimeter) and its sensor size in pixels

The user is capable of browsing these inputs according to the respective placements in the GUI. All IO and EO files must be loaded in their respective places. At this stage, the "Select Point" button is still greyed. The average ground height in meters (Z) as well as the CCD's pixel size in millimeters and the sensor size must also be inputted before continuing.

It should be noted that for the input of images, the user should first browse the folder containing the images ("Image Directory" button). Afterwards, the application will filter the folder to search for photos containing the name of the camera (Cam1, Cam2, etc) in their filenames e.g. *010284Cam1.jpg* or *022612_Cam2.jpg*. Afterwards, files with "Cam1" on their names will be listed in the list-box below the "Cam 1 (Nadir View)" figure. The user can select which image to show in the figure by clicking the filename on this list-box.

After inputting all the necessary data, a "Set Cams" button must be pushed in order for the program to create a temporary cell array containing a list of all the image names inside the file directory. With this list, it will also look for the images' respective EO and IO parameters from within the inputs and store them within the cell array. This step is necessary to narrow down the search for EO and IO parameters in the next more critical computational step to only a set of filenames available in the directory, rather than performing a search in the input file. As the number of images recorded in the inputted EO text file may be hundreds if not thousands, this mechanism provides a faster computing time.

Upon completion of the listing algorithm, the "Select Point" button will then be available for use. The user may then click on this button, whereas a crosshair would appear to allow them to click on any point on the nadir image. Several default Matlab figures tools such as zoom and pan as well as data cursor tool which enables the user to inquire the image coordinates (with pixel coordinate system) of a particular object are also available on the upper menu of the interface. Upon clicking a point, the

Address Offices in Brussels : Rue du Nord 76, BE – 1000 Bruxelles. Tel +32/2/217.39.72 Fax +32/2/219.31.47
E-mail: jean-yves.pirlot@clge.eu - www.clge.eu EC-Register of interest representatives 29077535421-69

algorithm will look for the appropriate EO and IO parameters from the cell array established in the previous step, and then compute the object space coordinates of the point by collinearity. It will then back-project this 3D coordinates back to the pixel coordinates of the other oblique images.

The list-boxes below the respective figures will then show all the images in the directory which contains the point selected within its borders. The program will also only show images from a particular camera in their respective list-box. The user can then choose from which listed image the point will be shown, and afterwards press the "Zoom to point" button to zoom into this point.

The monoplotting tools consist of three simple tools allowing the user to perform analyses and computations on the images:

1. Building height calculator, as the name suggests, is a tool to calculate the height of a building.
2. 3D ground distance calculator, calculates the distance between 2 points. However, since the program relies on DEM (in this case a flat surface with altitude Z) and not DSM, this distance assumes that the points are on the ground. Nevertheless it may be useful to measure, for example, the width of a road.
3. Vectorizer, is a simple digitizing tool. The user must first select two points on the base of the building which will serve as a reference line for the calculation of the coordinates. Then the user may start drawing polygons or lines by clicking on the "Draw" button. When all the drawings are finished and saved, the user may click on the "Export" button to have a text file (named "output.txt" and located in the workspace folder) and a .dxf file (named "outputdxf.dxf") created which list all the polygons.

Algorithm

In this particular program, we may divide the algorithm into two, namely the main collinearity and back-projection algorithm, used in the point selection and façade identification function and the auxiliary monoplotting algorithm which permits the user to calculate building heights amongst others. Both use the collinearity condition as their main mathematical relation.

Collinearity and back-projection

The backbone of this algorithm is the collinearity condition which describes the relation between a camera, the image of an object, and the object itself in 3D space. We have the collinearity equations which formalizes this condition in two well-known equations (adapted from Sheng, 2005):

$$x + \Delta x = x_0 - f.\frac{r_{11}(X - X_s) + r_{12}(Y - Y_s) + r_{13}(Z - Z_s)}{r_{31}(X - X_s) + r_{32}(Y - Y_s) + r_{33}(Z - Z_s)}$$
$$y + \Delta y = y_0 - f.\frac{r_{21}(X - X_s) + r_{22}(Y - Y_s) + r_{23}(Z - Z_s)}{r_{31}(X - X_s) + r_{32}(Y - Y_s) + r_{33}(Z - Z_s)} \quad \text{Eq. 1}$$

As well as their inverse forms:

$$X = X_s + (Z - Z_s).\frac{r_{11}(x - x_0) + r_{21}(y - y_0) + r_{31}(-f)}{r_{13}(x - x_0) + r_{23}(y - y_0) + r_{33}(-f)}$$
$$Y = Y_s + (Z - Z_s).\frac{r_{12}(x - x_0) + r_{22}(y - y_0) + r_{32}(-f)}{r_{13}(x - x_0) + r_{23}(y - y_0) + r_{33}(-f)} \quad \text{Eq. 2}$$

The X, Y and Z denotes the point's coordinates in the object space while x and y those at the image space. Notice that in these equations, we require the interior orientation (IO) and exterior orientation

Address Offices in Brussels : Rue du Nord 76, BE – 1000 Bruxelles. Tel +32/2/217.39.72 Fax +32/2/219.31.47
E-mail: jean-yves.pirlot@clge.eu - www.clge.eu EC-Register of interest representatives 29077535421-69

(EO) parameters. The IO parameters set the camera coordinate system and include the image coordinates of the camera's principal point (x_0 and y_0) as well as its focal length (f). For this project, the additional parameters (Δx and Δy) are considered to be null for now (although fully integrable). The EO parameters include the rotational and translational part and they describe the way coordinates in object frame can be transferred to the camera frame, or in other words they position the camera coordinate system with respect to the reference frame. The rotations are expressed in terms of angles ω, φ and κ around X, Y and Z axes respectively. They parameterize the 3x3 rotational matrix such that its columns perform sequential rotations along the three principal axes. The translational part is a 3D vector X_s, Y_s, and Z_s that represents the position of the camera perspective center in the global coordinate system.

Notice also that in order to solve the planimetric image space coordinates, we would require the value for Z (altimetric coordinate of the point). In monoplotting, this Z value is usually acquired through a DEM (Höhle, 2008). However, for this application the value of the average ground height is used instead which means that we assume that the ground is a flat surface. Although this may be a generalization of the terrain, for the purposes of this viewer it gives a good enough approximation of the DEM provided that the actual ground tends to be flat.

By employing the inverse collinearity equations (Eq. 2), we may compute a particular point's object space coordinates. We can then use the same coordinates to back-project them to the image space coordinates on the other oblique images using the collinearity equations (Eq. 1). Upon obtaining the image space coordinates for each of the oblique image inside the image directory, the program will launch a filtering algorithm to limit the images shown on the list-box to be only those in which the point is visible. Afterwards by clicking on the zoom button, the figure will auto-zoom to the respective image coordinate and therefore show the same point from several oblique perspectives.

Monoplotting tools

The first algorithm for 3D ground distance calculator is a simple calculation of the object space coordinates of the two points selected by Eq. 2. This is then followed by computing the 3D distance by using the Pythagorean formula. The only thing to note is that the distance will be on the ground, and may not reflect the correct results for distances on buildings (e.g. windows).

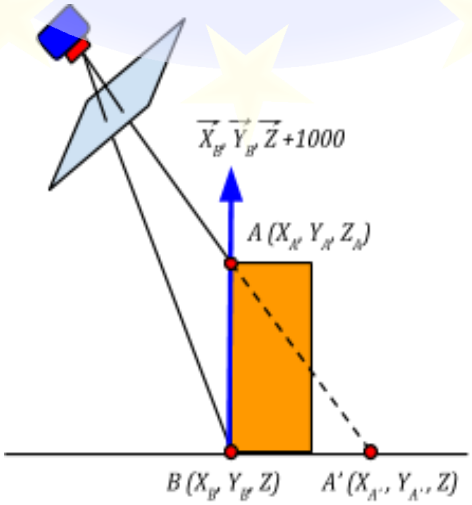

Figure 2. Determination of building height.

The algorithm for the building height calculator begins with the determination of the building base's image space coordinates. For this, again we use Eq.2 with the average ground height as the Z input.

Address Offices in Brussels : Rue du Nord 76, BE – 1000 Bruxelles. Tel +32/2/217.39.72 Fax +32/2/219.31.47
E-mail: jean-yves.pirlot@clge.eu - www.clge.eu EC-Register of interest representatives 29077535421-69

Afterwards, using the same planimetric coordinates for the building base (X_B, Y_B), we draw a straight line directed towards the Z axis (See Fig. 2), with the Z value equals to Z_B added by 1000 meters (under the assumption that no building is higher than 1 kilometer). Note that Z_B is equal to the average ground height, here noted simply as Z. The idea is to use this intermediary vector to find an intersection with the ray passing through the roof of the building.

Therefore, the next step would be to calculate the object space coordinates of the building roof using Eq.2. The notation A' denotes the intersection between the ray which passes through the building roof on the image space with the arbitrary plane of average ground height with the altitude Z, hence giving the object space coordinates of A' ($X_{A'}$, $Y_{A'}$, Z). By intersecting this ray with the previous intermediary vector which we had established, we will find the intersection coordinates which would be homologous to the building roof's object space coordinates (X_A, Y_A, Z_A). Afterwards by finding the difference between Z_A and Z, we will thus find the height of the building.

A similar procedure albeit a little more complicated is followed for the vectorizer tool (See Fig.3). In this case, the points of the digitized polygon may be multiple therefore the building height algorithm becomes impractical to determine the digitized point's altitude. Rather, the building base is defined by two points (B_1 and B_2). As before, the space coordinates of these two points are calculated using Eq.2. Another arbitrary point (noted B'_2) is then taken with the altitude added by 1000 meters. From these 3 points, a plane is defined.

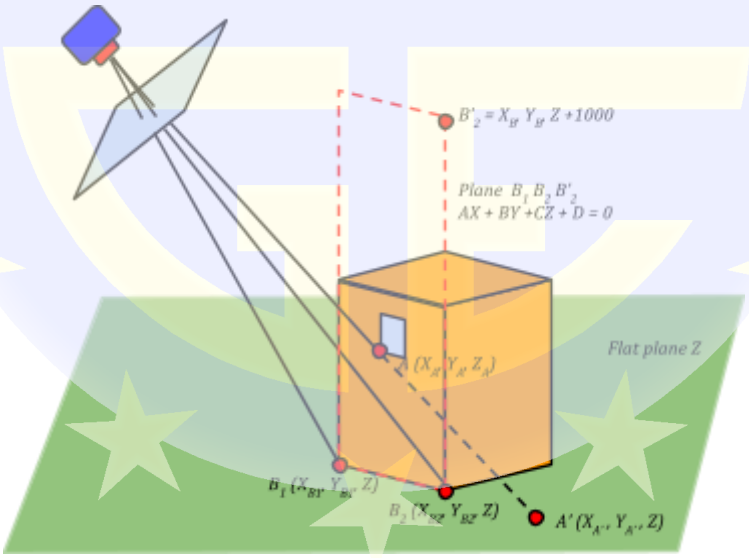

Figure 3. Determination of façade height for the vectorizer.

With this vertical plane defined, any ray above the B_1-B_2 line will intersect with it. We can therefore calculate the object space coordinates of the digitized point (A') using Eq.2, and then intersect it with the plane to determine its altitude (Z_A).

It should be noted though that for both the building height calculator and the vectorizer, the calculation of Z relies on the assumption that the building's walls are straight (hence the added Z value for the intermediary points in both cases). Therefore it may yield false results when used to digitize tilted surfaces such as roofs.

Address Offices in Brussels : Rue du Nord 76, BE – 1000 Bruxelles. Tel +32/2/217.39.72 Fax +32/2/219.31.47
E-mail: jean-yves.pirlot@clge.eu - www.clge.eu EC-Register of interest representatives 29077535421-69

Results and analysis

Figure 4. Use of the collinearity and back-projection functions to inspect a building's façades.

Address Offices in Brussels : Rue du Nord 76, BE – 1000 Bruxelles. Tel +32/2/217.39.72 Fax +32/2/219.31.47
E-mail: jean-yves.pirlot@clge.eu - www.clge.eu EC-Register of interest representatives 29077535421-69

The main function of the viewer application is to be able to inspect the sides of a building from different oblique points of view. This is accomplished by the calculation of the object's coordinates in the object space and re-projecting them to the other images using Equations 1 and 2 as previously described. The resulting visuals can be seen in Fig. 4.

The building height calculator and 3D ground distance calculator yield the results as shown in Fig. 5. As there are no ground truth data yet, a comparison has been taken with the average height of buildings. According to the Council on Tall Buildings and Urban Habitat (www.ctbuh.org), a residential building with 6 stories (as shown in Fig. 6a) should have an average height of 26.6 meters. The building height calculator gives a value of 24.4 meters for a residential building in the Milan data set.

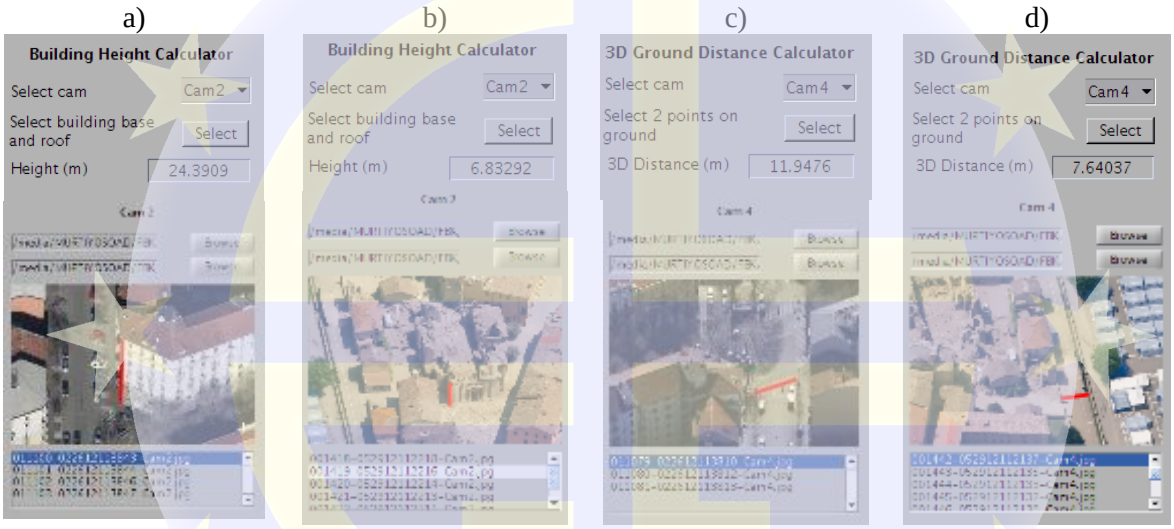

Figure 5. Use of the monoplotting principles to derive building heights (a, b) and measure ground distances such as a streets' widths (c, d) in urban and natural hazard scenarios.

Likewise, an example of the use of the vectorizer tool is shown in Fig. 6, while the resulting exported data is seen in Fig.7. The user may digitize multiple images to get the coordinates of a building from various oblique points of view. By clicking on the "Save" button, each of the digitized polygons will be saved.

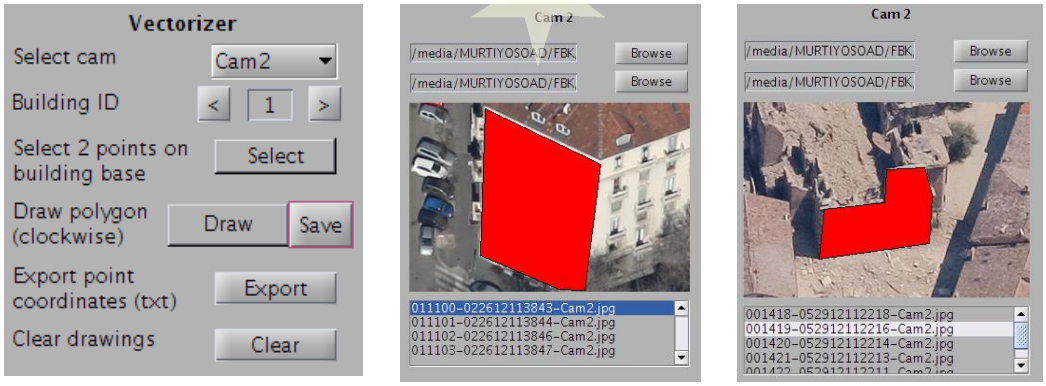

Figure 6. Digitization of a building façade or a damaged element using the vectorizer tool.

Address Offices in Brussels : Rue du Nord 76, BE – 1000 Bruxelles. Tel +32/2/217.39.72 Fax +32/2/219.31.47
E-mail: jean-yves.pirlot@clge.eu - www.clge.eu EC-Register of interest representatives 29077535421-69

```
101 011100-022612113843-Cam2.jpg 3362.067833 1558.369825 517717.167028 5036452.568628 160.869839
102 011100-022612113843-Cam2.jpg 3378.067833 1614.369825 517718.470754 5036451.602480 170.733751
103 011100-022612113843-Cam2.jpg 3431.067833 1600.369825 517723.779662 5036447.668225 173.178845
104 011100-022612113843-Cam2.jpg 3439.067833 1555.369825 517724.817336 5036446.899238 167.095900
```

Figure 7. Exported coordinates of the digitized polygon for further 3D modeling procedures.

In the output file, the first column denotes the ID of the points digitized in order of the sequence of the digitization. They are in hundreds as the first digit refers to the building's ID e.g. 101 for Building 1 first click and 304 for Building 3 fourth click. Each polygon digitized is separated by an empty line. The second column shows the filename of the image on which it is digitized, while the third and fourth columns show the image space coordinates of each point in pixel coordinate system and the last three columns denote the object space coordinates.

The user may then plot this data in another software to create a rough 3D model of the digitized object (Fig. 8). However, the accuracy as well as the precision of the model will not be too high due to various assumptions taken by the calculating algorithm. Nevertheless, this particular capability may be useful in some building inspection and disaster management applications.

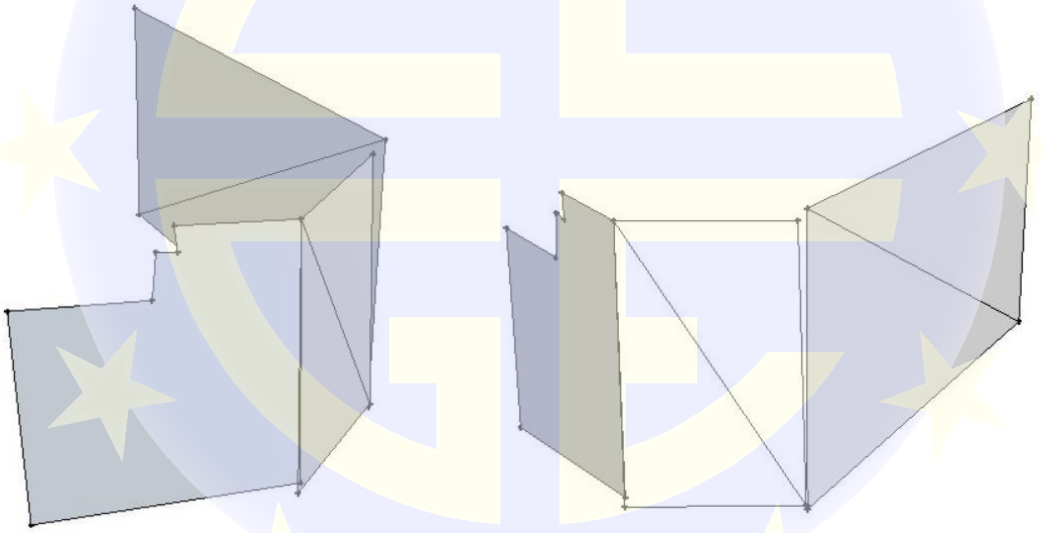

Figure 8. Example of some exported coordinates of a digitized building in Milan plotted in Sketchup.

Conclusions and further works

Seeing as this is still a work in progress, several aspects can definitely be improved and more tools and capabilities added. Accuracy of the results may be improved, amongst others, by the inclusion of the image distortion parameters and the use of a proper DEM file for the Z input. It should also be noted that for this application, an initial EO and IO data from GNSS and IMU sensors aboard the platform were used. Further accuracy may be attained by the use of EO and IO parameters obtained from image orientation (for example bundle adjustment method) and/or calibration respectively. See Grussenmeyer and Al Khalil (2002) for different methods of image orientation.

Some remaining issues such as the limitations of monoplotting can also be seen in this viewer. The use of a vertical line or plane emanating from building bases means that the algorithm assumes that the building façades are all vertical in nature. This would mean that the digitization of tilted roofs or walls will give generally worse results. The inclusion of stereo-photogrammetry tools may address this problem. Also, proper ground truthing should be performed to truly assess the real quality of the

Address Offices in Brussels : Rue du Nord 76, BE – 1000 Bruxelles. Tel +32/2/217.39.72 Fax +32/2/219.31.47
E-mail: jean-yves.pirlot@clge.eu - www.clge.eu EC-Register of interest representatives 29077535421-69

program's calculations. A further case study with the data set from the disaster-struck San Felice region is expected.

However, even in this early stage, this viewer has shown some important capabilities which can be implemented in building façade inspection as well as disaster management. In particular, the vectorizer tool may be developed further to generate a CAD file for further processing in CAD based programs. This is very useful in the aftermath of a disaster, to classify the damages quickly and to provide information for the purposes of aid and relief planning.

Acknowledgements

This work is conducted during an internship on the research topic "Oblique Aerial Cameras" at the 3D Optical Metrology (3DOM) research unit of the Fondazione Bruno Kessler (FBK), Trento, Italy. The author wishes to acknowledge CGR Spa/Blom Italy for the data provided. Furthermore, the author wishes to express thanks to Fabio Remondino, Ewelina Rupnik, and Francesco Nex (FBK Trento) for their discussions as well as supervision.

References and further reading

Bozzini, C., Conedera, M. and Krebs, P. (2012). A new monoplotting tool to extract georeferenced vector data and orthorectified raster data from oblique non-metric photographs. *International Journal of Heritage in the Digital Era*, 1(3), pp.499--518.

Ctbuh.org, (2014). *CTBUH Tall Building Height Calculators*. [online] Available at: http://www.ctbuh.org/TallBuildings/HeightStatistics/HeightCalculator/OnLineCalculator/tabid/1068/language/en-GB/Default.aspx [Accessed 19 Jul. 2014].

Gerke, M., Slagboom, Y. and Vosselman, G. (2014). Oblique Airborne Photogrammetry. *GIM International*, pp.18-21.

Grussenmeyer, P. and Al Khalil, O. (2002). Solutions for exterior orientation in photogrammetry: a review. *The photogrammetric record*, 17(100), pp.615--634.

Höhle, J. (2008). Photogrammetric measurements in oblique aerial images. *Photogrammetrie, Fernerkundung, Geoinformation*, (1), pp.7--14.

Remondino, F., Rupnik, E. and Nex, F. (2014). Automated Processing of Oblique Imagery. *GIM International*, pp.16-19.

Rupnik, E., Nex, F. and Remondino, F. (2014). Oblique Multi-Camera Systems-Orientation and Dense Matching Issues. *ISPRS-International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, 1(1), pp.107--114.

Sheng, Y. (2005). Theoretical analysis of the iterative photogrammetric method to determining ground coordinates from photo coordinates and a DEM. *Photogrammetric Engineering \& Remote Sensing*, 71(7), pp.863--871.

Address Offices in Brussels : Rue du Nord 76, BE – 1000 Bruxelles. Tel +32/2/217.39.72 Fax +32/2/219.31.47
E-mail: jean-yves.pirlot@clge.eu - www.clge.eu EC-Register of interest representatives 29077535421-69