International Journal of Geo-Information

MDPI

Article

Dense Local Azimuth–Elevation Map for the Integration of GIS Data and Camera Images

Gilbert Maître

Independent Researcher, Route de Pranoé 12, CH-1985 Villa, Switzerland; gilbert.maitre@vtxnet.ch

Abstract

The integration of outdoor camera images with three-dimensional (3D) geographic information on the observed scene is of interest for many video acquisition applications. To solve this data fusion problem, camera images have to be matched with the 3D geometry provided by a geographic information system (GIS). Considering a camera with a known geographical position, this paper proposes the use of a dense local azimuth–elevation map (LAEM) derived from a gridded digital elevation model (DEM) to represent the data and thus facilitate the matching of GIS and image data. To each regularly sampled azimuth and elevation angle pair, this map assigns the geographic point derived from the DEM viewed in this direction. The problem of computing the LAEM from the DEM is closely related to that of surface rendering, for which solutions exist in computer graphics. However, rendering software cannot be used directly in this case, since their view directions are constrained by the pinhole camera model and the apparent colour, rather than the position of the viewed point, is assigned to the viewing direction. Therefore, this paper also proposes a specific algorithm for the computation of the LAEM from the DEM. A MATLAB® implementation of the algorithm is also provided, which is tailored to process the DEM dataset swissALTI3D from the Swiss Federal Office of Topography swisstopo.

Keywords: gridded DEM; DEM rendering; image georeferencing; geometric camera calibration; ray casting

Academic Editors: Wolfgang Kainz, Zhenyu Yu, Mohd Yamani Idna Idris, Yu Li and Aleksandar Dj Valjarević

Received: 29 December 2025
Revised: 8 March 2026
Accepted: 10 March 2026
Published: 16 March 2026

1. Introduction

We consider a situation where a pan–tilt–zoom (PTZ) camera with a fixed position is used to monitor a landscape. In such a case, there is often interest in associating each camera pixel with the geodetic coordinates (e.g., WGS84) [1,2] of the observed physical point. This process involves the georegistration of the image or the georeferencing [3] of the image pixels. Once the geodetic coordinates of an observed point have been associated, a plethora of interesting geographical data from a geographical information system (GIS) become usable [4].

Digital elevation models (DEMs) provide information on the three-dimensional shape of the Earth's surface in the form of elevation values (vertical datum) on a GIS raster (horizontal datum) [5]. Several governmental mapping agencies provide this type of topographical data. For example, the Swiss Federal Office of Topography swisstopo provides the swissALTI3D [6] raster DEM, and other DEMs are available at the continental level for Europe, such as the EuroDEM dataset [7].

Regarding geometry, a camera transforms a 3D geometric space representing the physical world to a 2D space represented in an image. This transformation has been

widely studied in the fields of photogrammetry [8], computer vision [9,10], and computer graphics [11]. In most cases of modelling a real camera, this transformation is, as a first approximation, represented as a true-perspective projection [9]. This is often called the pinhole camera model, referring to an ideal camera where all light rays pass through a pinhole. The transformation can be seen to comprise three concatenated geometric transformations [9]. The coordinates of the source point in a 3D Cartesian world coordinate system are first transformed into their coordinates in a similar but camera-centric coordinate system, where the Z-axis is aligned with the optical axis of the camera. This transformation depends on the camera's extrinsic parameters: its pose, which is composed of its position and its orientation. Then, the 3D space is projected to the image plane, which is perpendicular to the optical axis, yielding dimensionless 2D coordinates. Finally, these last coordinates are transformed into pixel coordinates through a 2D transformation based on the camera's intrinsic parameters: focal length, pixel size(s), and origin of the sensor's coordinate system.

Except for wide-angle cameras [12], real cameras can be modelled with only a minor modification of the pinhole model [9]: a supplementary transformation is introduced between the second and third transformations mentioned above in order to take into account the lens distortions. This consists of transforming the dimensionless coordinates resulting from the projection into distorted coordinates by adding a distortion term. Since at least the 1980s [13], decomposing the distortions into a radial (pure radial displacement) and tangential component has become commonplace. A reason for this is that the radial component is often more important than the tangential one; therefore, the latter is often neglected. Furthermore, radial and tangential displacements, increasing with the distance from the image centre, are modelled using polynomial functions of the normalised coordinates.

Let us now assume that the model parameters of a camera monitoring a landscape are known and a DEM of this landscape is available. Then, georeferencing the image pixels is similar to rendering the surface spanned by the DEM with the camera of which we have a model. Indeed, in both cases, a point or a small patch of the observed surface is assigned to each pixel. In surface rendering, a colour is then attributed to the pixel according to the light emitted or re-emitted by the surface patch, while, in georeferencing, the geodetic coordinates of the observed surface point or patch are attributed to the image pixels. Rendering solutions exist (in particular, see [14]) and can be adapted for georeferencing.

Hence, in the case where there is a DEM of the landscape being monitored by a camera, georeferencing the image pixels mainly comes down to finding the parameters of the camera model. This problem, known as geometric camera calibration, has been widely studied in the fields of photogrammetry [15,16] and computer vision [9,17,18], and a large number of methods have emerged from both fields. What all methods have in common is that the model parameters are estimated from a set of geometric primitives in the physical world and their corresponding geometric entities in the image. In probably the majority of methods, the world's geometric entities are points at the surface of physical objects and their corresponding entities are image pixels or, when more precision is required, points with sub-pixel coordinates. Prior to the actual computation of the parameter estimates, the first problem to be solved is the selection of calibration points, i.e., points in the scene and in the image that correspond to each other.

This situation of a camera monitoring a landscape for which we have a DEM is a particular camera calibration case in which we also have a 3D geometric model of the scene at hand. Thus, the correspondence to be established is between the image and 3D model, rather than between the image and physical world. This problem is, in its general formulation, difficult. Early solutions have worked on 2D shapes generated from 3D geometry, e.g., silhouettes [19] or shadows [20]. More recently, several authors [21–24]

tackled the difficult problem of finding cross-modal similarities between an intensity image and a 3D geometric model, and they proposed interesting solutions.

This paper does not present such an advanced method for finding correspondences between the camera image and 3D model, which is a DEM in this case; rather, a representation of the DEM, called a local azimuth–elevation map (LAEM), is proposed to ease this cross-modal matching. This approach takes advantage of the ease of determining the geographical position of a fixed camera through other means, such as with a hand-held global positioning system, orthophoto, or land survey map. Given the position, the method downsizes the set of candidates of the DEM points for the cross-modal matching and arranges them in a 2D array consistent with the 2D image array. Figure 1 shows how the LAEM and its construction are involved in the whole georegistration procedure.

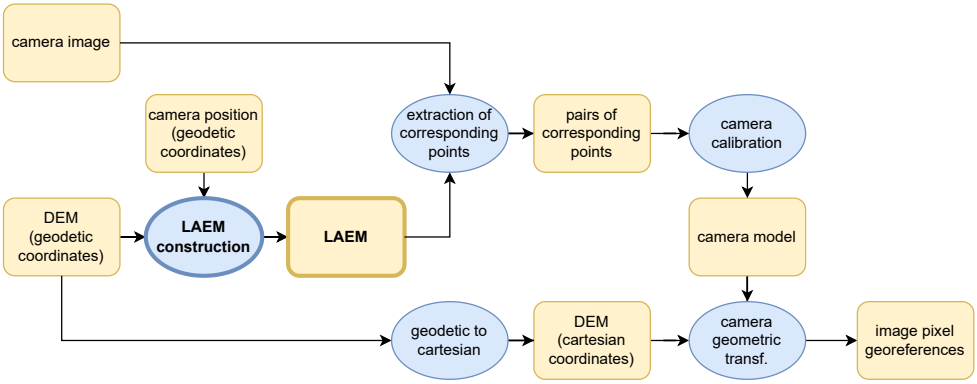


Figure 1. Location of LAEM within georegistration procedure.

The remainder of this paper is organized as follows: Section 2 presents related works. Then, in Section 3, the LAEM is presented as a particular case of 3D to 2D mapping, and a general method is described to compute an LAEM from a DEM in geodetic coordinates. Section 4 is concerned with the method's implementation and experimental results. The implementation adapts the general method to the specificities of the swissALTI3D [6] DEM. Finally, Section 5 summarises the advantages of the particular DEM representation created using the LAEM and provides research directions for how correspondences between an image and the LAEM could be found.

2. Related Works

To the best of my knowledge, the problem of automatically selecting points in the camera image and DEM that correspond to each other has been rarely addressed. Therefore, as the aim is to perform image georegistration of oblique views, this section presents related work from this broader perspective, emphasising the determination of the camera pose in a geographical coordinate system. Note that the latter is sometimes referred to as the geolocalisation of the image, although this is often understood to concern only the camera position, without its orientation [25]. In the following, unless otherwise specified, geolocalisation considers both the position and orientation of the camera. The related work is organised according to the data used as reference for the georegistration, or geolocalisation:

1. Orthophoto or georeferenced aerial images;
2. Georeferenced 3D point cloud;
3. Digital elevation model (DEM).

Instead of covering numerous publications, this overview prioritizes detailing specific examples in each category. Occasionally, personal comments are added.

2.1. Georegistration/Localisation with Orthophotos or Georeferenced Aerial Images

In the software developed by Rameau et al. [26], the selection of calibration points in the camera image and in an orthographic satellite image are selected semi-automatically. However, there are two limiting assumptions in this case: the camera must have a mostly top-down view of the scene and the calibration points must lie on a more or less planar surface of the scene. The selection is semi-automatic, since the software user has to first rotate the satellite image until it is almost aligned with the camera image. Then, a region of interest (ROI) has to be drawn, where the calibration points are selected automatically.

Shan et al. [27] proposed a solution for the georegistration of ground images using georeferenced aerial images. They assumed that the ground images have a GPS tag and hence an approximate geoposition. The method they proposed aims to improve the precision of the geolocation and then incorporate the orientation to the position. As there are several ground images of the same scene, a camera self-calibration is performed, which provides at the same time a 3D model of the scene. Then, the camera's intrinsic parameters are known, but not its pose in a geographic coordinate system. The aerial views are not exactly vertical as in an orthophoto, however the difference between the angle of the views of both types of images (terrestrial and aerial) is still sufficiently large to make direct matching between images unsuccessful. The ground image, 3D model, and approximate geolocation of the ground image are used to build a synthetic image from the aerial point of view. Then, by matching points between the synthetic and aerial images, the camera pose is determined in the geographical coordinates system, thereby achieving a more precise geolocalisation of the ground image. Comment. Shan et al. have shown impressive results with ground images of landmarks in Rome and aerial images of the city. These are images of man-made shapes with a rather simple geometry. The method may be less successful with natural or rural landscapes.

2.2. Georegistration/Localisation with Georeferenced 3D Point Clouds

Li et al. [28] proposed a method for the geolocalisation of images using georeferenced 3D point clouds. This is an extension to the geolocalisation methods of previous work [29] or similar work by Sattler et al. [30], where localisation is performed in a restricted area, typically a town. What is called a 3D point in these works, is in fact not a pure geometric primitive represented by coordinates in a 3D coordinate system but comprises also descriptors taken from the images used to construct the geometric point. Hence, to establish the correspondence between pixels in the image to be localised and some 3D points of the cloud, basically the same methods as for matching pixels between images are used. However, the existing geometric information accompanying the image descriptors creates constraints helping to select the right corresponding 3D points in the vastness of the point cloud. With the corresponding image pixels on one side and 3D points on the other side, the image can be localised with a camera calibration method. In the more recent work of Li et al. [28], a 3D point cloud had a worldwide extension, comprising 3D points from many places (cities) all around the world. The first difference, compared to the previous work, is the significantly larger size of the cloud, which makes the search for corresponding 2D–3D point pairs more difficult and necessitates a more performant solution. Another difference is that localisation must be performed with respect to a global coordinate system, i.e., image geolocalisation is now required. The 3D points are georeferenced using the geotag (latitude and longitude) of the images with which the 3D point was constructed. As in the case of localisation in a restricted area, the whole camera pose (3D position and orientation) is determined; however, the proposed geolocalisation method seems to consider only the horizontal coordinates of the position, since the authors used precisely geotagged photos to test their method's precision.

2.3. DEM-Based Georegistration/Localisation

Härer et al. [31] as well as Milosavljević et al. [32] proposed solutions for georeferencing images with a DEM. However, for the step of selecting calibration points, they used an orthophoto in addition to the DEM. Moreover, a human operator performs the selection. Comment. The manual selection of calibration points is time-consuming and can even be difficult, particularly with natural landscapes. For example, the ridge summits and passes in a camera image can be difficult to identify and localise precisely on an orthophoto or even on a topographical map. In the opposite direction, singular points detected in the orthophoto can be difficult to identify and localise in the camera image. This is because the camera's view of the landscape is oblique, i.e., quite far from the exactly vertical view of an orthophoto.

Portenier et al. [33] proposed a solution for the georegistration of camera images with a DEM for the purposes of measuring snow cover. The camera position is assumed to be known a priori with sufficient precision, but not its orientation. As they used a simple camera model, without lens distortions and with the optical image centre located in the middle of the digital image, only four parameters have to be estimated to calibrate the camera. Three are extrinsic—the orientation components—and one is intrinsic—the field of view (FOV), which is equivalent to the focal length presented in the Introduction. The calibration process is automatic except for the first step, which involves selecting at least one image with good sky–ground contrast, called the master image. In this image, the visible horizon is detected, and the polyline indicating the horizon is taken as a reference. Then, the space of camera model parameters is discretised within the potential value ranges. Next, the DEM is rendered for each discrete value of the parameter vector, and the visible horizon is detected and compared with the reference. Finally, the parameters selected for modelling the camera are those that minimise the difference with the reference horizon. Portenier et al. noticed that the camera model has not a permanent validity. Changes in weather conditions and/or human interactions, intentional or not, may induce small changes in camera orientation, requiring a model update. Because the calibration described above is a cumbersome process, the model is not updated through re-calibration. A newly captured camera image is matched with the master image; then, the camera rotation and hence new orientation are inferred from the pairs of corresponding image points.

2.4. Final Comments

The main difficulty in geolocalisation with orthophotos and georeferenced aerial images is that the image to be geolocalised can differ significantly from the reference image(s). This is due to the difference in not only views but also capture times. In the time interval between the two captures (the reference and to-be-geolocalised images), drastic changes may have occurred in the scene (e.g., land coverage) or illumination (e.g., day/night). Because of these drastic changes, the images can be so different that geolocalisation becomes impossible.

Geolocalisation with georeferenced 3D point clouds begins with a large dataset of georeferenced photos taken by individuals, from which a georeferenced 3D point cloud is first generated. The 3D points serve to determine the camera pose of the image being geolocalised. But to achieve this, the correspondence between image pixels and some of the 3D points has to be performed and it relies on the similitude of this image with those used to build the 3D points. Hence, geolocalisation with georeferenced 3D point clouds is subject to the same problem mentioned before resulting from differences in capture time between images. However, in this case, the problem can be mitigated by having many images of the same scene at hand, taken under different lighting and weather conditions. As the 3D point clouds are based on photos taken by individuals during their leisure time, the use of

this geolocalisation method is restricted to popular locations with frequent visitors, e.g., city landmarks.

DEM-based geolocalisation can be associated with geolocalisation with georeferenced 3D point clouds, as a DEM is a georeferenced 3D point cloud. However, the physical reality modelled by the point cloud differs. 3D point clouds in section 2.2 model popular man-made spaces, like city landmarks. On the contrary, DEMs model the ground of large regions—scientifically speaking [5], the interface between the atmosphere and lithosphere. Therefore, these approaches are actually complementary.

Another difference between both approaches is that DEMs are not built with images from multiple views and therefore do not hold image information that can be used to find the correspondences between the image to be geolocalised and the DEM. It is true that there often exists an orthophoto that can be associated point by point with the DEM. However, as mentioned in the above paragraph discussing geolocalisation with orthophotos, because of the problematic large difference in views, the similarity between the orthophoto and image to be geolocalised is poor. In Milosavljević et al.'s method [32], the operator undoubtedly uses his understanding of the image to be geolocalised and of the orthophoto to select the calibration points. New methods have to be proposed for the automatic selection of corresponding points between the image to be geolocalised/georegistered and a DEM, such as the attempt made by Portenier et al. [33] and the one proposed in this paper.

3. Local Azimuth–Elevation Map at Camera Position

3.1. Scenario

As explained in the Introduction, we consider the case of a PTZ camera fixed in position with an oblique view of a natural landscape for which we have a DEM. The typical camera set-up, which is the one used for the experiments (Section 4), is presented in Figure 2 (the camera CAD model in the figure was downloaded from [34] and rendered with Blender 4.5). The camera pan axis is more or less vertical, and the pan rotation allows a 360° view around the camera position. The proposed method is not exclusive to PTZ cameras. However, it is more useful in cases where the camera has changes in orientation and field of view, and is especially well suited for the camera set-up shown in Figure 2.

Since the camera has a fixed position, its geodetic coordinates only have to be determined once, which can be performed during camera installation or even earlier. This is an easy task, especially if the observed landscape is relatively distant from the camera, as in this case, the required positioning accuracy is low. The camera geodetic coordinates can be determined, e.g., using a hand-held global positioning system (GPS), or read on a digital map, e.g., on Google Maps [35].

Knowing a priori the camera position and assuming that the DEM is available in the same geodetic coordinate system as the camera position, we will see that we can represent the DEM in a local coordinate system, where the origin is at the camera position.

3.2. The ENU and AER Local Coordinate Systems

Readers can refer to Figure 3 to follow along with the text below.

Let us consider a point P_0, the camera position in our case, with geodetic coordinates [1,2,36]: we have geodetic latitude ϕ_0, longitude λ_0, and geodetic or ellipsoidal height h_0. Let S_0 be the orthogonal projection of P_0 on the ellipsoid modelling the Earth. h_0 is hence the Euclidian norm of the vector $\overrightarrow{S_0P_0}$. Let π_0 be the plane tangential to the Earth ellipsoid at S_0 and π be the plane parallel to π_0 and containing P_0.

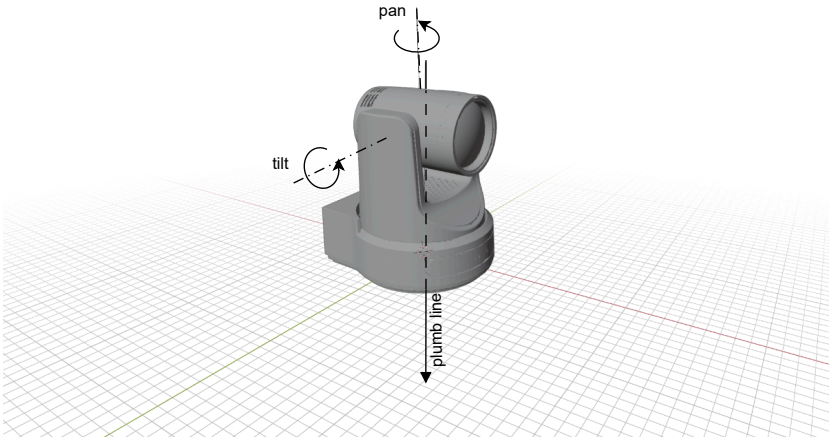


Figure 2. Typical set-up of the PTZ camera.

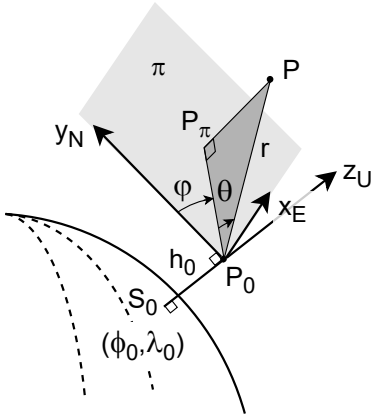


Figure 3. The ENU and AER coordinate systems.

The local East-North-Up (ENU) [1,36] or Local Geodetic Horizon (LZH) [2] coordinate system is a Cartesian coordinate system with origin P_0 and the following three axes, which are orthogonal to each other:

- Up is the direction of the vector $\overrightarrow{S_0P_0}$, denoted by z_U in Figure 3.
- North is the direction at S_0 of the meridian, i.e., the ellipse through the geodetic north, and is taken to be positive towards the geodetic north. In Figure 3, the north axis is represented in the plane π parallel to π_0, and denoted by y_N.
- East is the direction orthogonal to the other two, making ENU is a right-handed system. It is denoted by x_E in Figure 3.

Since ENU is a Cartesian coordinate system, it can be used to represent 3D points for camera calibration.

The formulas used to transform the coordinates of a point in the geodetic coordinate system to its coordinates in the local ENU coordinate system use an intermediate representation in the Earth-centred, Earth-fixed (ECEF) coordinate system. They can be found, e.g., in [1,2]. They are implemented in the MATLAB® Mapping Toolbox since R2012b through the function `geodetic2enu` [37] and in the PyMap3D Python module since at least v2.3.1 through the function `pymap3d.enu.geodetic2enu` [38].

The coordinate transformation is invertible, although there is no explicit form for the geodetic coordinates in the function of the ECEF coordinates [1]. The transformation from ENU coordinates into geodetic coordinates is implemented in the MATLAB® Mapping Toolbox through the function `enu2geodetic` [39] and in the PyMap3D Python module through the function `pymap3d.enu.enu2geodetic` [40].

The Azimuth–Elevation–Range (AER) coordinate system is the spherical companion of the ENU system [36]. Azimuth and elevation angles constitute a widely used coordinate system in astronomy to represent the direction from an observer position [41]. The elevation angle is sometimes called the altitude angle. There are some variations in how these angles are measured. In the AER coordinate system, the coordinates of a point P also include the range (or slant range), i.e., the distance from P_0 to P. In the following, we use the same conventions as in [36] for the AER coordinate system. Let P_π be the orthogonal projection of point P on plane π; then, we state the following:

- The azimuth φ is the clockwise angle from the axis y_N to the direction $\overrightarrow{P_0P_\pi}$.
- The elevation (or altitude) θ is the angle between the plane π and direction $\overrightarrow{P_0P}$; positive up, negative down.
- The (slant) range r is the Euclidean distance between points P_0 and P.

As in [36], the angles are expressed hereafter in degrees, in the range $[0°, 360°]$ for the azimuth φ and in the range $[-90°, 90°]$ for the elevation θ.

The following formulas are used to compute the AER coordinates from ENU coordinates:

$$\varphi = \begin{cases} \text{atan2}(x_E, y_N) & \text{if } x_E \geq 0, \\ \text{atan2}(x_E, y_N) + 360° & \text{if } x_E < 0, \end{cases} \tag{1a}$$

$$\theta = \text{atan2}(z_U, \sqrt{x_E^2 + y_N^2}), \tag{1b}$$

$$r = \sqrt{x_E^2 + y_N^2 + z_U^2}. \tag{1c}$$

Note that the azimuth φ is undefined when both x_E and y_N are null. The elevation θ is undefined when all three ENU coordinates are null, i.e., for the coordinate system origin P_0.

The functions `geodetic2aer` [42] from the MATLAB® Mapping Toolbox and `pymap3d.aer.geodetic2aer` [43] from the PyMap3D Python module allow us to directly and completely transform geodetic coordinates to AER coordinates.

The following formulas are used to compute the ENU coordinates from AER coordinates:

$$x_E = r \cos\theta \sin\varphi, \tag{2a}$$

$$y_N = r \cos\theta \cos\varphi, \tag{2b}$$

$$z_U = r \sin\theta. \tag{2c}$$

The functions `aer2geodetic2` [44] from the MATLAB® Mapping Toolbox and `pymap3d.aer.aer2geodetic` [45] from the PyMap3D Python module allow us to directly and completely transform AER coordinates to geodetic coordinates.

3.3. Local Azimuth–Elevation Map (LAEM)

3.3.1. Mapping to the Azimuth–Elevation Space

The local azimuth–elevation map (LAEM) is defined as the result of mapping from the 3D geometric space $\mathbb{R}^3$ to the 2D space $\mathbb{R}^2$, where the azimuth angle φ and elevation angle θ of a source point are represented in a Cartesian coordinate system. Figure 4 presents this mapping graphically for a point P_n. Since the azimuth φ is undefined if both x_E and y_N are null, the points belonging to the z_U-axis do not have an image. Furthermore, the image of $\mathbb{R}^3$ is a limited domain of $\mathbb{R}^2$. With the conventions adopted in Section 3.2 for the azimuth and elevation angles, the image domain is given by the following ranges on φ and θ values: $(\varphi \in [0°, 360°[) \wedge (\theta \in]-90°, 90°[)$. Note that the domain does not include the elevation values $-90°$ and $90°$, because they correspond to undefined azimuth values (points on the z_U-axis).

Let us now point out the following feature of the image domain, which arises due to a well-known property of the inverse trigonometric functions. Two points close to each other in the 3D space, one with a positive x_E value and one with a negative x_E value, have distant images. The first image point has a φ value slightly larger than 0°, while the second one has a φ value slightly smaller than 360°.

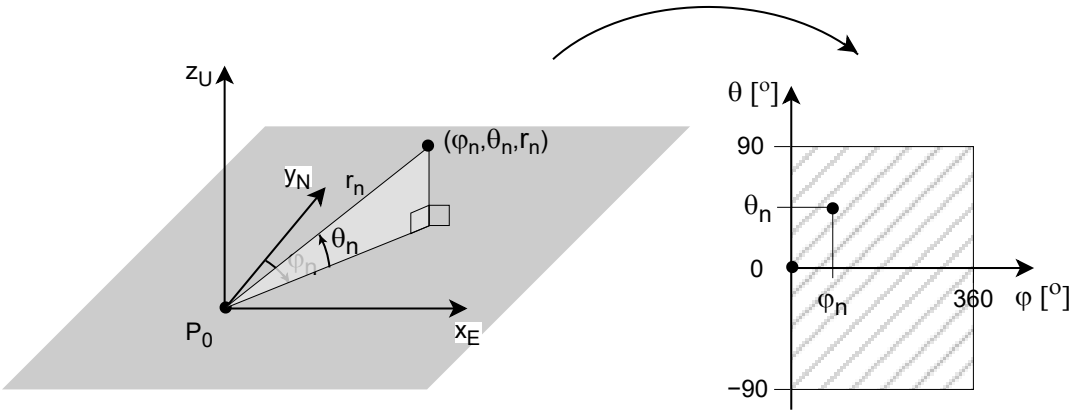


Figure 4. Mapping from the 3D geometric space to the azimuth–elevation space.

The mapping introduced above is similar to the perspective projection realised with a camera at position P_0, in the sense that all points of a half-line from P_0 are mapped to a single point. However, there are at least two differences:

1. The formula used to derive the image coordinates (φ, θ) from the AER coordinates of the 3D point is different from that for deriving the normalised camera image coordinates (x, y) from the camera-centric Cartesian coordinates (X, Y, Z).
2. A line segment in 3D space, the endpoints of which map to different points, does not map to a line segment but to another curve.

3.3.2. LAEM of a Surface

The LAEM of a surface in 3D space is the mapping of the points belonging to the surface, but there is an exclusion condition: from the points belonging to the same half-line from P_0, only the one nearest to P_0, i.e., the "visible" one, is mapped. With this condition, the mapping becomes injective. As a consequence, for each pair of values (φ, θ), there is a maximum of one surface point mapped to this image point. In Figure 5, points P_n and P'_n have the same azimuth φ_n and elevation θ_n values. However, only point P_n with smaller range value r_n is mapped to the image point (φ_n, θ_n). As a consequence, the unique surface point P_n can be associated with values φ_n and θ_n.

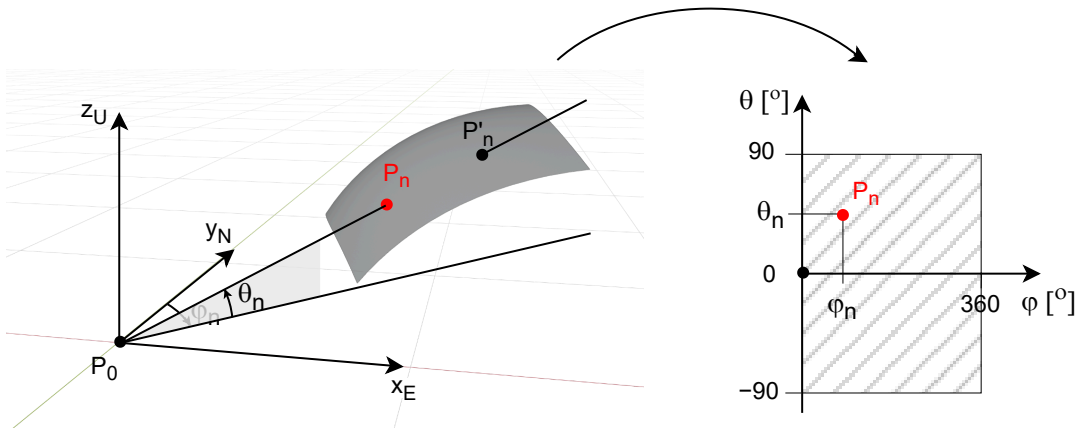


Figure 5. LAEM of a surface.

3.3.3. Discrete LAEM

For an implementation in a technical system, the LAEM has to be discrete, i.e., be a countable set of values. Moreover, the set has to be finite. Among the various possibilities of discretising the LAEM, let us propose the following, which is driven, at least partly, by the goal of representing a discrete LAEM as a digital image for easy storage and processing.

We use a constant sampling interval $\Delta\varphi$ for the azimuth as well as a constant sampling interval $\Delta\theta$ for the elevation. For convenience, we require that $\Delta\varphi$ and $\Delta\theta$ divide $90°$. Formally, this can be expressed as follows:

$$\frac{90°}{\Delta\varphi} = \left\lfloor \frac{90°}{\Delta\varphi} \right\rfloor, \tag{3a}$$

$$\frac{90°}{\Delta\theta} = \left\lfloor \frac{90°}{\Delta\theta} \right\rfloor. \tag{3b}$$

In practice (see Section 4), this constraint is not restrictive, since we use $\Delta\varphi$ and $\Delta\theta$ values as small as $0.01°$.

As it is usual to index the rows and columns of a digital image with i and j, respectively, starting from top and from left, respectively, we index the discrete azimuth values φ_j with j, starting with the smallest value, and θ_j with j, starting with the largest value. This is presented in Figure 6, where x and y, the commonly used image coordinates [46,47], are also shown. Most programming languages start table indexes with 0; this is also assumed for i and j. However, MATLAB® starts table indexes with 1. We use identifiers i' and j' in this case. The relationships between both types of indexes are of course simply given by the following:

$$i' = i + 1, \qquad i = i' - 1, \tag{4a}$$

$$j' = j + 1, \qquad j = j' - 1. \tag{4b}$$

In the envisioned camera-to-landscape configurations, only a limited portion of the elevation range $[-90°, 90°]$ is significant. Let $\theta_{\min}$ and $\theta_{\max}$ be the minimal and maximal elevation values of interest, respectively. For convenience, we require $\theta_{\min}$ and $\theta_{\max}$ to both be integer multiples of $\Delta\theta$. Formally, this can be expressed as follows:

$$\theta_{\min} = k_1 \cdot \Delta\theta, \;\; k_1 \in \mathbb{Z}, \tag{5a}$$

$$\theta_{\max} = k_2 \cdot \Delta\theta, \;\; k_2 \in \mathbb{Z}. \tag{5b}$$

With these conditions, the number N_θ of discrete elevation values θ_i is

$$N_\theta = \frac{\theta_{\max} - \theta_{\min}}{\Delta\theta} + 1 \tag{6}$$

and the discrete elevation values θ_i are given by the equation

$$\theta_i = \theta_{\max} - i \cdot \Delta\theta. \tag{7}$$

Regarding the azimuth, there may be cases where the full range of values $[0°, 360°]$ is of interest. However, there also cases where only a segment of the possible range is of interest. In some cases, the segment of interest will be over the transition from $359.\bar{9}°$ to $0°$. In the case where the full range of azimuth values is of interest, one could decide to easily handle the angle wrapping, by expanding the LAEM with an overlap corresponding to the horizontal field of view of the camera. In order to handle all of these cases in a simple manner, we propose the use of unwrapped angle values, i.e., angles values that are always increasing. Let φ' denote the unwrapped azimuth and $\varphi'_{\min}$ and $\varphi'_{\max}$ the minimal and

maximal unwrapped angle values, respectively. For the full range with overlap mentioned above, one would set, for example, $\varphi'_{\min} = -\frac{\alpha}{2}$ and $\varphi'_{\max} = 360^\circ + \frac{\alpha}{2}$, where α is the horizontal field of view of the camera . For convenience, we require $\varphi'_{\min}$ and $\varphi'_{\max}$ to both be integer multiples of $\Delta\varphi$. Formally, this can be expressed as follows:

$$\varphi'_{\min} = k_1 \cdot \Delta\varphi, \;\; k_1 \in \mathbb{Z}, \tag{8a}$$

$$\varphi'_{\max} = k_2 \cdot \Delta\varphi, \;\; k_2 \in \mathbb{Z}. \tag{8b}$$

With these conditions, the number $N_{\varphi'}$ of discrete unwrapped azimuth values φ'_j is

$$N_{\varphi'} = \frac{\varphi'_{\max} - \varphi'_{\min}}{\Delta\varphi} + 1 \tag{9}$$

and the discrete unwrapped azimuth values φ'_j are given by the equation

$$\varphi'_j = \varphi'_{\min} + j \cdot \Delta\varphi. \tag{10}$$

The discrete wrapped azimuth angle φ_j corresponding to φ'_j is then simply given by

$$\varphi_j = \mathrm{mod}(\varphi', 360^\circ), \tag{11}$$

where the mod function is defined by

$$\mathrm{mod}(a, b) = a - b \cdot \left\lfloor \frac{a}{b} \right\rfloor. \tag{12}$$

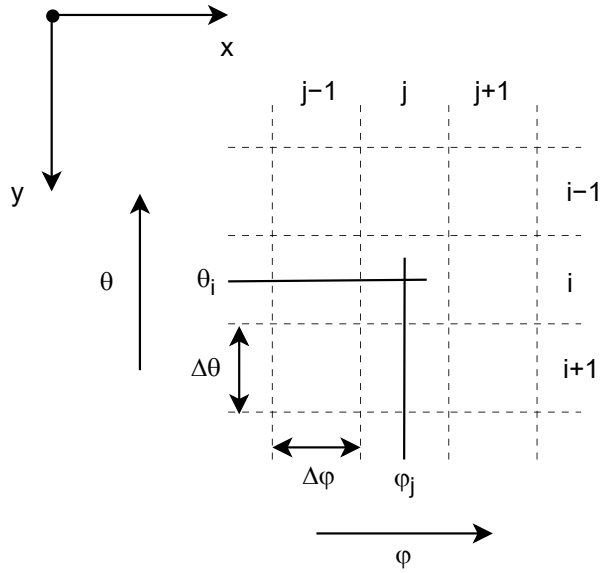


Figure 6. Coordinates of the discrete LAEM.

3.4. Computing a Discrete LAEM of a Gridded DEM

3.4.1. Methods from Surface Rendering

Sections 3.3.1 and 3.3.2 show that computing a discrete LAEM of a surface in 3D space is similar to rendering this surface with a camera model, which is a basic operation in computer graphics [11], and we note that a DEM is basically a surface. Therefore, whether methods used for surface rendering can be adapted to the computation of a discrete LAEM is worth considering.

Surface rendering starts by establishing correspondence between image points, i.e., pixels, and points belonging to the surface. To achieve this, there are basically two opposite

approaches. The first one proceeds from the scene to the image. The surface is modelled with polygonal patches, either as an exact model or as an approximation obtained via tessellation. Then, the image of the polygon vertices is computed using the true perspective projection modelling the camera. Finally, since linear segments are imaged into linear segments, the polygon edges and interior areas are drawn. This first approach is less precise than the second one because the correspondence is not established individually for each pixel within a patch. However, it is less intensive in terms of computation. It is therefore preferred when the computation time is critical. Despite the advantages of this approach, it cannot be adapted to LAEM computation, since, as mentioned in Section 3.3.1, line segments are not mapped into line segments in this case.

The second approach proceeds from the image to the scene. This method is known as ray casting [48], where a ray is traced from the camera centre through each image pixel and the corresponding surface point is determined as the intersection of the ray with the surface. In the case of an LAEM, points in 3D space with a constant elevation and azimuth also build rays from a fixed point, which is the coordinate system origin in this case. The only difference with respect to ray casting lies in the distribution of rays in the space. In surface rendering, rays are regularly spaced according to the angle tangent, while in LAEM, they are regularly spaced with respect to the angle value itself. The ray casting method can therefore be easily adapted to LAEM computation. In this paper, we only mention this possibility, emphasising the method presented in the following section.

3.4.2. Principles of the Proposed Method

The method proposed in this paper for computing a discrete LAEM is different from the surface rendering methods discussed in the previous section. It proceeds mainly from the scene (DEM) to the image (LAEM), but also partly from the LAEM to the DEM. Moreover, it is not linked to the rendering of the surface represented by the DEM, as it makes use of the volume upper-bounded by the DEM.

Let $P_{m,n}$ be a particular element of the DEM with position indices m and n and let us take one of the four-neighbourhood elements, e.g., $P_{m-1,n}$. The elementary surface patch used for the LAEM computation is a delimited cross-section of the volume upper-bounded by the DEM, as shown in Figure 7. The delimiting edges are line segment $[P_{m-1,n}, P_{m,n}]$, the two segments of vertical lines from $P_{m-1,n}$ and $P_{m,n}$ to the minimal elevation value $\theta_{\min}$, and the line segment joining these two last points.

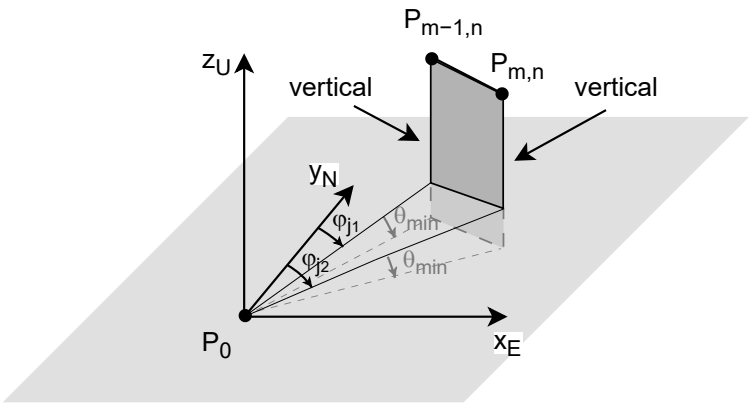


Figure 7. Elementary surface patch used for DEM computation.

The motivation for taking vertical lines is that a DEM provides elevation values (vertical datum) on a GIS raster (horizontal datum) [5]. At the origin of the local ENU coordinate system, the vertical direction of the DEM is the Up-axis. However, the more distant a DEM point is from the ENU origin, the larger the deviation of the vertical direction

at that point from the Up-axis of the ENU coordinate system. For a DEM where the elevation is the geodetic height, as we have assumed, the vertical direction coincides with the normal to the underlying ellipsoid. Determining the exact vertical direction at any point of a DEM is, however, cumbersome, and there is generally no need to be very precise in the vertical direction. To have a reference value for the required precision, let us take a DEM grid resolution of 2 m, like the DEM used in the experiment, and the greatest purely vertical drop on Earth [49], which is 1200 m. The angle formed with respect to the vertical by two consecutive DEM points, one at the top and one at the bottom of the drop, is $\arctan(2/1200)$, which is equal to 0.0955°. This, rounded to 0.1°, provides the precision which is sufficient for determining the vertical direction. Given this tolerance, to determine the vertical direction, we propose using one of the following two degrees of approximation for the DEM's horizontal datum: planar and spheric (see Section 3.4.3).

As explained in the introductory paragraph of this section, the method also proceeds from the LAEM to the DEM: Let φ'_{j1} and φ'_{j2} be the discrete azimuth values corresponding to $P_{m-1,n}$ and $P_{m,n}$, respectively, as shown in Figure 7. If the indices j_1 and j_2 are apart by more than one—formally, $|j_2 - j_1| > 1$—then for any in-between value j, i.e., $j_1 < j < j_2$, a point P_j is determined on the line segment $[P_{m-1,n}, P_{m,n}]$ as the intersection of the plane defined by all points of the same azimuth φ'_j and the line passing through points $P_{m-1,n}$ and $P_{m,n}$ (see Figure 8).

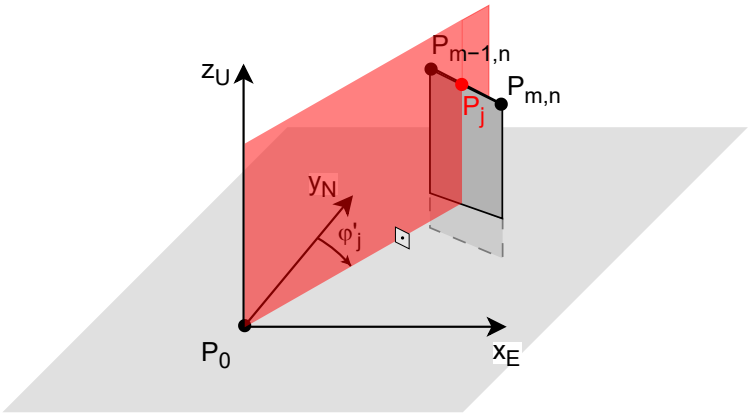


Figure 8. DEM interpolation point at discrete azimuth value φ_j.

The ENU coordinates of point P_j can be easily computed from φ'_j and the ENU coordinates of $P_{m-1,n}$ and $P_{m,n}$ by solving a system of linear equations. Let (x_1, y_1, z_1) be the coordinates of $P_{m-1,n}$ in the ENU coordinate system and (x_2, y_2, z_2) be those of $P_{m,n}$. Then, the coordinates (x_j, y_j, z_j) of P_j in the ENU coordinate system can be found by solving the following system of equations:

$$-\cos\varphi'_j \cdot x_j + \sin\varphi'_j \cdot y_j = 0, \tag{13a}$$

$$x_j = \lambda(x_2 - x_1) + x_1, \tag{13b}$$

$$y_j = \lambda(y_2 - y_1) + y_1, \tag{13c}$$

$$z_j = \lambda(z_2 - z_1) + z_1. \tag{13d}$$

Equation (13a) denotes the plane determined by φ'_j, while Equations (13b)–(13d) are the parametric equations of the line through points $P_{m-1,n}$ and $P_{m,n}$. Equation (13a) can be replaced with the following explicit equation for λ:

$$\lambda = \frac{\cos\varphi'_j \cdot x_1 + -\sin\varphi'_j \cdot y_1}{-\cos\varphi'_j \cdot (x_2 - x1) + \sin\varphi'_j \cdot (y2 - y1)}. \tag{14}$$

Once point P_j is determined, the patch is "filled" by going downward from this point in the vertical direction.

3.4.3. Computing DEM Vertical Lines

As explained in Section 3.4.2, to compute vertical lines from a DEM point, we propose using approximations of the DEM horizontal datum. We consider two possible approximations: planar and spheric.

Planar Horizontal Datum

If the DEM horizontal datum is the local horizontal plane, as shown in Figure 4, the DEM vertical direction is the normal to this plane. Hence, all points on a vertical line passing through a DEM point have the same coordinates x_E and y_N as this point, as well as the same azimuth φ and horizontal distance $\varrho = \sqrt{x_E^2 + y_N^2}$.

Spheric Horizontal Datum

At P_0, the origin of the local coordinate system, the curvature of the ellipsoid underlying the DEM depends on the direction considered. According to [1], the minimal radius of curvature is in the meridian, i.e., in the y_N-axis direction, while the maximal is in the perpendicular direction, i.e., in the x_E-axis, which is called curvature in the prime vertical. The values of these two extreme local radii of curvature depend on the geodetic latitude ϕ_0 of P_0. The formulae for the radius of curvature in the meridian $\rho(\phi_0)$ and for the radius of curvature in the prime vertical $\nu(\phi_0)$ are

$$\rho(\phi_0) = \frac{a(1-e^2)}{(1-e^2\sin^2\phi_0)^{3/2}}, \tag{15a}$$

$$\nu(\phi_0) = \frac{a}{(1-e^2\sin^2\phi_0)^{1/2}}, \tag{15b}$$

where a is the length of the ellipsoid major axis, being 6,378,137.0 m for GRS80, and e^2 is the ellipsoid eccentricity squared, being 0.006694380022901 for GRS80.

As an approximation of the DEM horizontal datum around P_0 with a sphere, we propose using a radius R between the radius of curvature in the meridian $\rho(\phi_0)$ (minimal value) and the radius of curvature in the prime vertical $\nu(\phi_0)$ (maximal value), augmented, of course, by the geodetic height h_0 of P_0. Since the geometric mean of both curvatures has a simpler formula than the arithmetic mean, we prefer the first option. The formula for computing R from the geodetic latitude ϕ_0 and height h_0 is

$$R = \sqrt{\rho(\phi_0)\nu(\phi_0)} + h_0, \tag{16a}$$

$$= \frac{a\sqrt{1-e^2}}{1-e^2\sin^2\phi_0} + h_0. \tag{16b}$$

Given a DEM point P_n, the vertical line from this point is normal to the sphere and hence crosses the z_U-axis at the sphere centre C. Let us consider a plane containing P_n and parallel to the horizontal plane through P_0. We can draw a rectangle in this plane, as shown in Figure 9a. The lengths of its edges are the coordinates x_E and y_N of P_n. For any point on the vertical line from P_n, we can do the same: take a parallel plane and draw the rectangle. The edge lengths of each rectangle are the coordinates x_E and y_N of the corresponding point on the vertical line. All rectangles are similar to each other. Consequently, the coordinates x_E and y_N of the points on the vertical line have a constant ratio x_E/y_N or y_N/x_E, and therefore a constant azimuth, equal to the one of P_n, that is, φ_n.

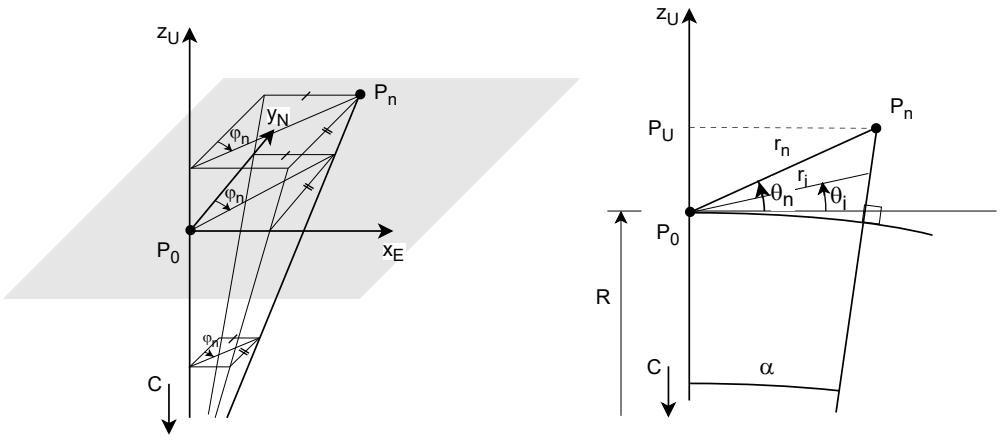


Figure 9. Line from P_n to sphere centre C: (**a**) 3D view; (**b**) view in the plane CP_0P_n.

Figure 9a shows the vertical line from P_n in the plane CP_0P_n. Considering the triangle CP_nP_U in Figure 9b, it is obvious that the angle α between the vertical line from P_n and z_U-axis depends on the elevation angle θ_n and the range r_n of P_n according to the following formula:

$$\tan\alpha = \frac{r_n \cos\theta_n}{R + r_n \sin\theta_n}. \tag{17}$$

The elevation and range of any point on the vertical line from P_n satisfy the same equation, particularly the points corresponding to the discrete elevation values θ_i. The implicit equation for θ_i and corresponding range r_i can be rewritten as

$$r_i(\cos\theta_i - \tan\alpha \sin\theta_i) = R\tan\alpha, \tag{18}$$

which can be converted into the following explicit equation for r_i:

$$r_i = \frac{R\tan\alpha}{\cos\theta_i - \tan\alpha \sin\theta_i}. \tag{19}$$

Maximal Errors in the Vertical Direction

We conclude this section by proposing an estimated upper bound for the errors in the vertical direction resulting from each of the two horizontal datum approximations presented above.

Let us consider a point P_n in the local horizontal plane distant from P_0 by d. This is a particular case in Figure 9b, representing the vertical line from P_n for a spheric model of radius R. Equation (17), which determines the angle α between the vertical line from P_n and z_U-axis, takes the simpler form $\tan\alpha = d/R$.

In the case of planar approximation, the vertical direction at P_n given by the model is that of the z_U-axis. If we assume that the right vertical direction is the one given by the sphere of radius R, the error is measured by the angle α. For a given distance d, α_R is maximal when R is minimal. The minimal value for R is the radius of curvature in the meridian at the equator, i.e., $\rho(0^\circ)$. Hence, we consider $\alpha_{\rho(0^\circ)}$ an upper bound of the error in the vertical direction resulting from the use of planar approximation.

Around P_0, the ellipsoid modelling the Earth lies between a sphere of radius $\rho(\phi_0)$ and a sphere of radius $\nu(\phi_0)$. The angle error in estimating the vertical direction with a sphere of mean radius, as with spheric approximation, should be smaller than the difference between the vertical directions determined by both spheres. This difference is maximal

at the equator. Therefore, we consider $(\alpha_{\rho(0^\circ)} - \alpha_{\nu(0^\circ)})$ an upper bound of the error in the vertical direction resulting from the use of spheric approximation.

Table 1 lists the numerical values of $\alpha_{\rho(0^\circ)}$ and $(\alpha_{\rho(0^\circ)} - \alpha_{\nu(0^\circ)})$ at different distances d of P_0. The used reference ellipsoid is GRS80. Given the tolerance of 0.1° for a grid resolution of 2 m, as computed in Section 3.4.2, it follows from the numerical values of Table 1 that the planar approximation can be used up to a distance of about 10km and that the spheric approximation has no practical limitation for this tolerance.

Table 1. Estimated upper bound for the error in the vertical direction on the GRS80 ellipsoid.

d	**Planar Approximation** $\alpha_{\rho(0^\circ)}[^\circ]$	**Spheric Approximation** $(\alpha_{\rho(0^\circ)} - \alpha_{\nu(0^\circ)})[^\circ]$
100 m	0.000904	6.054193×10^{-6}
1 km	0.009044	6.054193×10^{-5}
10 km	0.090437	0.000605
100 km	0.904294	0.006053

3.4.4. Detailed Method Description

We use a scalar two-dimensional array to represent the LAEM. Hereafter, we denote an array element by $d_{i,j}$ and the whole array by $\mathbf{D}$. In the case of planar approximation, d is the horizontal distance $\varrho = \sqrt{x_E^2 + y_N^2}$, while in the case of spheric approximation, d is the slant range r. In either case, the triplet $(\varphi_j, \theta_i, d_{i,j})$ represents a point of the 3D space, the coordinates of which can be computed in any coordinate system: AER, ENU, geodetic, ...

The pseudo-code in Listing 1 details how the LAEM $\mathbf{D}$ is computed from a gridded DEM, the elements of which are denoted by $P_{m,n}$. We only provide additional information where we deem necessary, referring to the code by line number.

Listing 1. LAEM computation.

```
1   % compute coordinates of DEM points P_{m,n}
2   for each P_{m,n}:
3       compute j_{m,n}, i_{m,n}, d_{m,n}
4   % add coordinates of interpolation points P_j
5   for each j_{m,n}:
6       for each j | j_{m,n}<j<j_{m+1,n} and each j | j_{m,n}<j<j_{m,n+1}:
7           compute i_j, d_j making use of (14), (13b), (13c) and (13d)
8   % fill the LAEM D
9   d_{i,j} = +∞, ∀i, j
10  for each (j_P, i_P, d_P) ∈ {(j_{m,n}, i_{m,n}, d_{m,n})} ∪ {(j, i_j, d_j)}:
11      if d_P < d_{i_P,j_P}:
12          for each i ∈ {i_P, i_P + 1, ··· , N_θ − 1}:
13              % spheric approximation          %planar approximation
14              compute r_i with Equation (19)   nop
15              if r_i < d_{i,j_P}               if d_P < d_{i,j_P}
16                  d_{i,j_P} = r_i                  d_{i,j_P} = d_P
17              else                             else
18                  break                            break
```

Line 3: The values $j_{m,n}$, $i_{m,n}$, and $d_{m,n}$ result directly from the representation of $P_{m,n}$ in the AER coordinate system. $d_{m,n}$ represents either the horizontal distance (planar approximation) or slant range (spheric approximation) of $P_{m,n}$. $j_{m,n}$ and $i_{m,n}$ determine the position of $P_{m,n}$ in the LAEM. They are derived from the azimuth φ and elevation θ of $P_{m,n}$ according to the choices made in

Section 3.3.3 for the discretisation of the (φ, θ) space. The equations are as follows:

$$j = \left\lfloor \frac{\varphi' - \varphi'_{\min}}{\Delta\varphi} + \frac{1}{2} \right\rfloor, \qquad \text{if } \varphi'_{\min} - \frac{\Delta\varphi}{2} < \varphi' \leq \varphi'_{\max} + \frac{\Delta\varphi}{2}, \tag{20a}$$

$$i = \left\lfloor \frac{\theta_{\max} - \theta}{\Delta\theta} + \frac{1}{2} \right\rfloor, \qquad \text{if } \theta_{\min} - \frac{\Delta\theta}{2} < \theta \leq \theta_{\max} + \frac{\Delta\theta}{2}, \tag{20b}$$

where φ' is the unwrapped azimuth angle, related to the azimuth angle φ by

$$\varphi' = \varphi + k \cdot 360^\circ, k \in \{-1, 0, 1\}. \tag{21}$$

Line 7: The interpolation is explained in Section 3.4.2 and illustrated in Figure 8. j determines the discrete unwrapped azimuth angle φ'_j. The explicit Equations (14), (13b), (13c) and (13d) allow us to compute from φ'_j the ENU coordinates of the interpolation point P_j between $P_{m,n}$ and $P_{m+1,n}$ and between $P_{m,n}$ and $P_{m,n+1}$. From the ENU coordinates of the interpolation point P_j, the corresponding θ_j and d_j values are derived with the ENU-to-AER coordinate transform. Finally, the elevation index i_j is computed from θ_j using Equation (20b). This new coordinate triplet (j, i_j, d_j) is added to the set (or list) of coordinates that will be considered to fill the LAEM.

lines 9 and 11: According to Section 3.3.2, from the points having the same LAEM position, we keep only the "visible" one, i.e., the one with the smallest d value. The LAEM is hence initialised at line 9 with a specific value, which is larger than any possible d value; here, the choice is $+\infty$. Furthermore, at line 11, the position (i_P, j_P) of the currently evaluated point is only updated if the value d_P is smaller than the value stored in $\mathbf{D}$ at this position.

Lines 12 to 18: i_P together with j_P determine the position of the currently evaluated DEM point. According to Section 3.4.3, points on the vertical line from a DEM point have the same azimuth, hence here the same j_P value. i values larger than i_P, each together with j_P, determine hence the LAEM positions of 3D points located on the vertical line down from the DEM point. In the case of planar approximation, points on the vertical line have the same horizontal distance ϱ value as the DEM point, i.e., d_P. In the case of spheric approximation, points on the vertical line have a slant range r_i computed from θ_i using Equation (19). The LAEM is updated at positions (j_P, i) with the values d_P and r_i, respectively, but only as long as they are smaller than the existing value d_{i,j_P}.

3.4.5. DEM Downsampling as Preprocessing

The discrete LAEM as defined in Section 3.3.3 has a constant azimuthal resolution $\Delta\varphi$. The corresponding horizontal spatial resolution Δs, as defined in Figure 10, varies with the horizontal range ϱ according to the following formula:

$$\Delta s = 2\varrho \sin\left(\frac{\Delta\varphi}{2}\right) \tag{22}$$

For an azimuthal resolution of $\Delta\varphi = 0.01^\circ$, as used in the experiment (Section 4), the spatial resolution Δs takes the value 0.1745 m at a horizontal distance ϱ of 1 km, 1745 m at 10 km, and 17.45 m at 100 km. At a distance of about 10 km, the spatial resolution Δs is hence about the same as the 2 m grid resolution of the used DEMs. Figure 11a shows

the case with a significantly shorter distance than the balance distance of equal resolutions, while Figure 11b shows the case with a significantly longer distance.

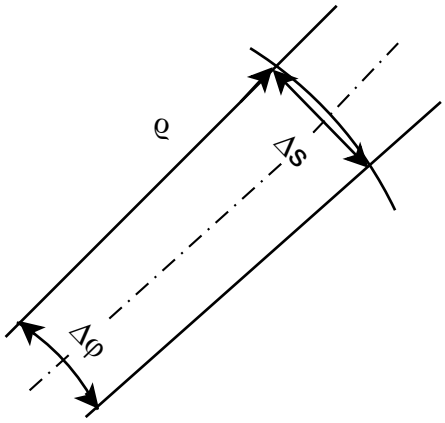
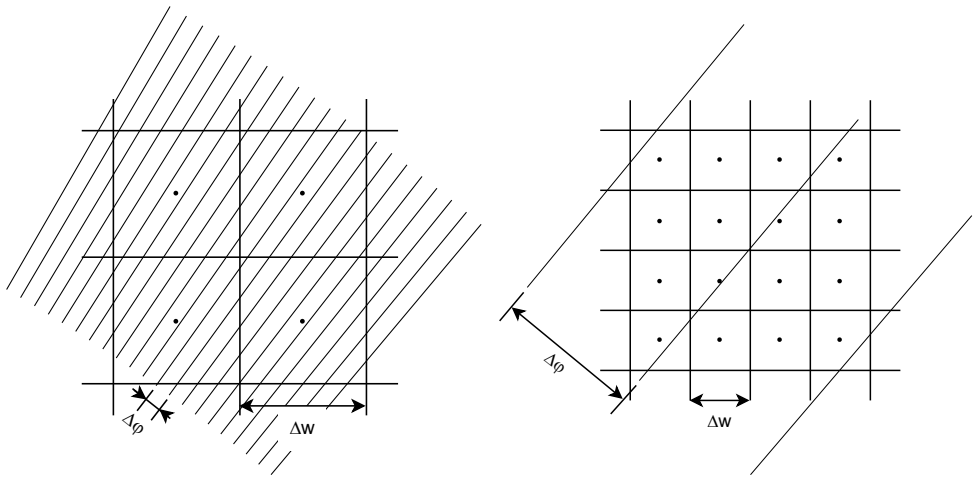


Figure 10. Azimuthal and spatial resolution of the LAEM.

In the situation shown in Figure 11a, interpolation ensures that the LAEM obtains a value at all discrete azimuth values, as well as at those situated between DEM points. In the situation shown Figure 11b, several neighbouring DEM points are mapped to the same j value. However, the condition in Listing 1, line 11 selects only the nearest one, thus implicitly subsampling the DEM points, but without any prior aggregation of those points into a representation of the group. In digital signal processing [50] and image processing [9], such subsampling is known as causing aliasing, which is an undesirable effect, and has to be preceded by low-pass filtering to avoid it. Here, we only suggest to introduce a preprocessing which correctly downsamples the DEM where needed. However, thus far, we have not worked out a solution where such downsampling is integrated into the LAEM computation process presented in Section 3.4.4.



(**a**) (**b**)

Figure 11. LAEM variable spatial resolution compared to constant DEM resolution Δw: (**a**) near the reference point, (**b**) far from the reference point.

4. Implementation and Experimental Results

4.1. The swissALTI3D DEM

swissALTI3D [6] is a raster digital elevation model (DEM) currently available in the Swiss LV95 [51] (for grid) and LN02 [52] (for height) coordinate systems. All coordinates are in meters. The data have two grid resolutions: 0.5 and 2 m.

The DEM is free to download from [6]. It is split into files (tiles) corresponding to a terrain square of 1 km by 1 km, available in two formats: Cloud Optimized GeoTIFF (COG) and zipped ASCII X,Y,Z. ESRI ASCII GRID is also available upon request. Tiles are selected by clicking on a zoomable map of Switzerland (default); alternatively, rectangle and polygonal selection can be employed, as well as selection by political regions: canton or municipality. The whole dataset may also be downloaded. For a small number of selected tiles (e.g., a square of 6 by 6 tiles), the files can be downloaded individually. For larger selections, a csv file with a list of file links, e.g., https://data.geo.admin.ch/ch.swisstopo.swissalti3d/swissalti3d_2019_2594-1119/swissalti3d_2019_2594-1119_2_2056_5728.tif (accessed on 21 December 2025), is provided for direct download from the web page. The individual files are then accessible by the links.

More information on the DEM is available in a technical report in German [53] and French [54].

4.2. Camera Set-Up and Images

The camera used for the experiment was an AXIS M5525-E PTZ Dome Network Camera, featuring 360° endless pan, 0–90° tilt, and 10× zoom capabilities. For other camera features, the interested reader can refer to the product support web page [55]. The camera was installed on the roof of building ENP23 of the HES-SO Valais-Wallis in the town of Sion. This high position gives a 360° view above the neighbouring buildings on the side slopes of the Rhône Valley.

While dome cameras are often installed on ceilings to monitor the half-space below the ceiling. In this set-up, the camera was installed upside down to monitor the half-space above the building roof. To obtain a view above surrounding objects on the roof, the camera was fixed at the top of a 2000 mm high, 800 mm wide frame, as shown in Figure 12a. The frame was levelled with a spirit level. This mechanical structure was selected to easily allow, on demand, the installation of other monitoring sensors. As shown in Figure 12b, a crown-shaped hardware mask was 3D printed and glued on the camera casing to avoid collecting privacy-sensitive data from neighbouring buildings while taking pictures.

(**a**) (**b**)

Figure 12. Mechanical camera set-up. (**a**) View at distance: the dome camera is within the red circle. (**b**) Close view. With respect to (**a**), this view was taken from behind the camera.

Figure 13 shows a typical image from the camera. The black area at the bottom of the image is due to the privacy mask. Pictures were systematically taken with this camera over a whole year at a mean rate of two sessions per day. A subset of the sessions is publicly available from the B2SHARE repository [56] under the "Creative Commons Attribution 4.0 International" license. When demanding access to the whole dataset, use the contact information given at [56].

Figure 13. Typical camera image in daylight and good weather. The image is dated 2023-09-06T12:08:59 (UTC). The camera parameters were as follows: pan 12.0°, tilt 6.9°, and zoom 4×.

4.3. Determining the Camera Position

As a first step, the camera position was determined on the online digital map of Switzerland, made available by the Swiss Federal Office swisstopo [57].

The procedure used is summarised graphically in Figure 14, where, to avoid confusion between the local ENU and LV95 coordinate systems, u (instead of x) is used for the easting LV95 coordinate and v (instead of y) is used for the northing LV95 coordinate.

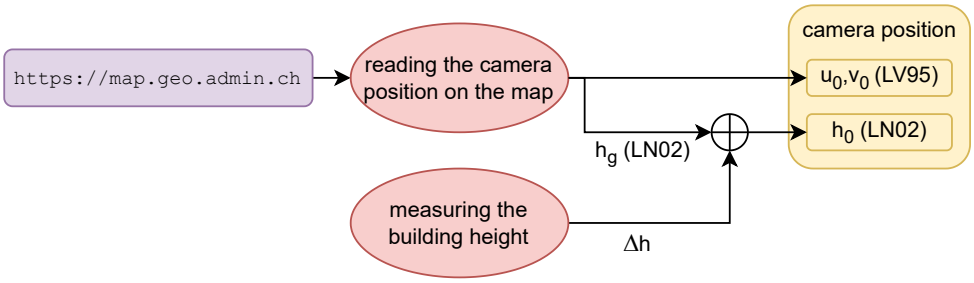


Figure 14. Procedure used to determine the camera position.

When selecting the background of aerial imagery, it is easy to identify the camera location on the building roof and position the mouse cursor at the camera location. The position coordinates are displayed when right-clicking. In Figure 15, the camera location is at the top of the small white triangle and the coordinates can be read in the displayed text.

The given elevation value, in the LN02 coordinate system, is of course not the altitude of the camera, but that of the ground level. To determine the height of the camera position with respect to the ground level, we used a laser distance meter Leica Disto X4 and found the LN02 altitude of the camera to be 517.5 m.

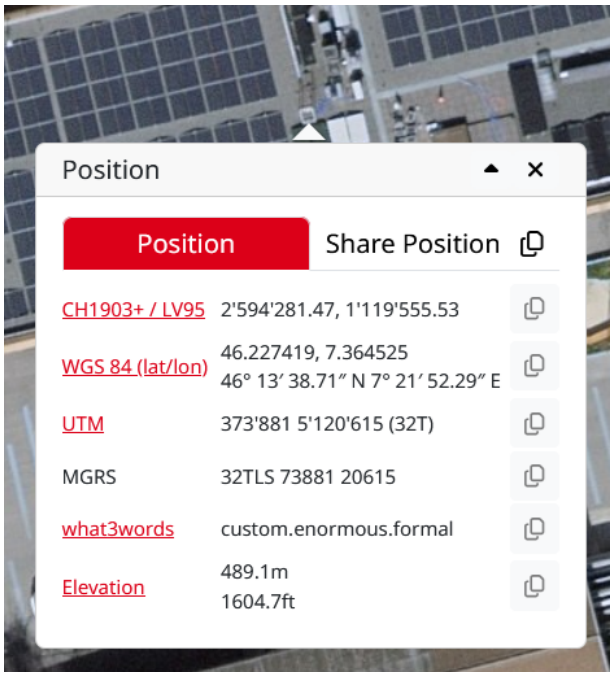


Figure 15. Camera position from map.geo.admin.ch.

4.4. Preselection and Download of the DEM Tiles

We chose the DEM in COG file format at a 2 m grid resolution, as we considered it quite sufficient for modelling the scene monitored by the camera.

Figure 16 summarizes how the DEM tiles were preselected and downloaded. Because the part of the Rhône Valley where the camera was located is also part of the Canton Valais, the tiles were simply preselected by choosing the option "Selection by canton" and then entering the name "Valais". This generated a csv file to be downloaded. We downloaded the file, which contained, in a single column, a list of 5562 hyperlinks to the individual tile files. The files themselves were downloaded with the help of a MATLAB® m-file reading the csv file, performing an http request for each hyperlink in the list, and saving the received data as a file in a local directory.

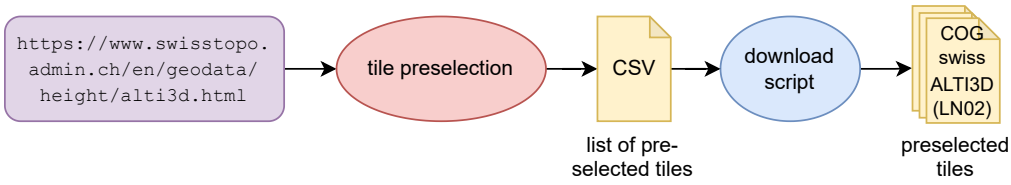


Figure 16. Procedure for preselecting and downloading the DEM tiles.

Even if it is not significant, we want to mention that the downloaded DEM files do not fully cover the landscape visible from the camera, since a very small portion of the visible terrain is in France and not surveyed by the Swiss Office of Topography. The region of interest is in a very small angle of view and at at a distance of more than 40 km from the camera.

4.5. Transformation to Geodetic Coordinates

As mentioned in Section 4.1, the swissALTI3D DEM uses Swiss-specific coordinate systems for both the horizontal and vertical data. In Section 4.3, the camera position was determined in the same coordinate systems. The transformation described in Section 3.2, and on which the computation of the LAEM is based, is however from a geodetic coordinate

system into the local AER coordinate system. In order to use this transformation, we have therefore first to transform the DEM and camera position from the Swiss-specific coordinate systems into a geodetic coordinate system.

The LN02 elevation values are orthometric heights. They cannot be converted into geodetic heights using a mathematical formula, because they are of different natures. The first elevation is linked to gravity, while the second is linked to geometry. Conversion into geodetic heights necessitates the official Swiss geoid model CHGeo2004 [58]. The Swiss Federal Office of Topography swisstopo provides the free online service REFRAME for coordinate transformation, particularly the transformation of LN02 heights into geodetic heights on the Bessel 1841 ellipsoid, which is what we need.

Swisstopo offers different possibilities for using the REFRAME service. First, the transformation can be performed through the web page [59], where a file can be uploaded, the transformation selected, and the resulting file downloaded. This is an easy way for transforming one or a few files. It is also possible to transform coordinates using the REST API [60], but one http request has to be made for each pair or triplet of coordinates. Swisstopo also provides software libraries (DLL, JAR) [61] to enable the use of the REFRAME service within software applications. However, the provided functions also perform the coordinate transform for a single point only. As a final option, the use of the GeoSuite software for Windows [62] is provided freely by swisstopo. This software allows us to transform coordinates in batches of multiple files. The REFRAME module and the type of transformation can be easily selected, as shown in Figure 17. The only drawback is that the module does not accept swissALTI3D files in GeoTIFF format for the needed altitude transform.

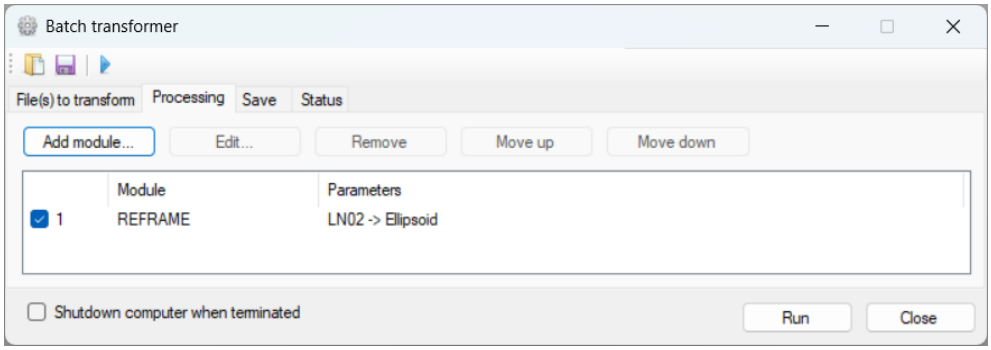


Figure 17. Geosuite batch transform.

For the transformation of the camera altitude from LN02 to geodetic, we used the REFRAME REST API, with the following http request:

https://geodesy.geo.admin.ch/reframe/ln02tobessel?easting=2594281.5&northing=1119555.5&altitude=517.5

We received back the following data:

```
{"type": "Point", "coordinates": [2594281.5, 1119555.5, 518.1399090414052]}.
```

The camera altitude on the Bessel 1841 ellipsoid was then recorded as 518.14 m.

For the altitude transformation of the swissALTI3D files, we chose the batch ability of the GeoSuite software, combined with a transformation of the swissALTI3D files from GeoTIFF to XYZ before the altitude transformation, and from XYZ to GeoTIFF after the transformation. We conducted a test with a batch of 500 files and noticed that the transformation was very resource-demanding. Since the difference between the LN02 and geodetic height has a magnitude no greater than 10 m, we decided to start the processing with the LN02 heights and perform the transformation only for tiles that actually remained after the selection described in Sections 4.6.1 and 4.6.2.

For the transformation of LV95 coordinates into geodetic latitude and longitude on the Bessel 1841 ellipsoid, for both the camera position and DEM files, we simply used the `projinv` function [63] of the MATLAB® Mapping Toolbox. The projected Coordinate Reference System (CRS) object to be passed to the function as the first parameter can simply be retrieved from any swissALTI3D COG file when reading the file with the function `readgeoraster` [64]. If `R` is the variable receiving the second output argument of `readgeoraster`, the value to be passed to `projinv` is simply `R.ProjectedCRS`.

The LN02 heights and the camera LV95 coordinates were transformed once, prior to the LAEM construction. Figure 18 summarises these transformations. However, the DEM LV95 coordinates were transformed during the LAEM construction in a block with the transformation from geodetic to local AER coordinates (see Figure 19).

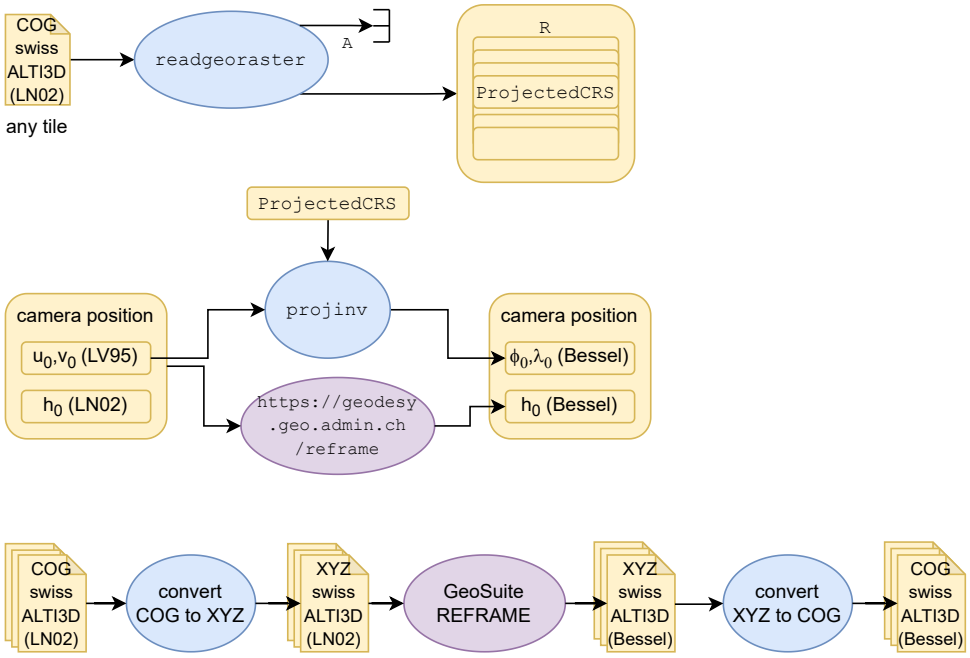


Figure 18. Transformation to geodetic coordinates on the Bessel 1841 ellipsoid performed prior to the LAEM computation.

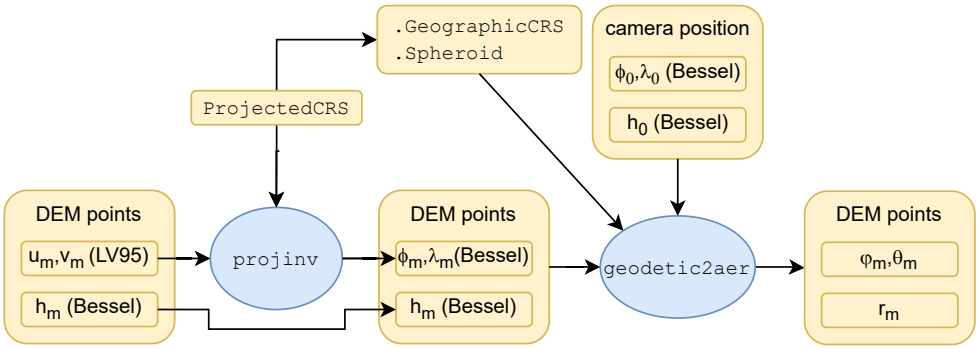


Figure 19. Composed transformation from original DEM coordinates to AER coordinates.

4.6. LAEM Computation

4.6.1. Tile-Level Processing

The preselection of the swissALTI3D tiles performed in Section 4.4 is quite broad. Many of the preselected tiles model landscape, which is not at all visible from the camera position. To avoid useless processing, a processing of lower computational complexity was performed at the tile level, extracting information with which it will be possible, while

filling the LAEM tile by tile, to decide whether a new tile has to be processed in the manner shown in Listing 1 or not, because it contains no point visible from the camera position.

In Section 4.5, we have explained that the transformation of LN02 heights into geodetic heights using the swisstopo REFRAME service was too resource-demanding to perform for all preselected files. Ideally, the tile-level processing should be performed with tiles in geodetic heights. But, due to the practical limitations, we performed it with tiles in LN02 heights.

A swissALTI3D tile is rectangular in the LV95 coordinate system. Let $u_{\min}$, $u_{\max}$ and $v_{\min}$, $v_{\max}$ be the extremal easting and northing coordinates, respectively. For adjacent tiles in the east direction, $u_{\min}$ of the eastern tile is strictly larger than $u_{\max}$ of the western tile. The difference is equal to the DEM resolution (2 m in our case). The same holds for the northing coordinate v.

The pseudo-code in Listing 2 describes the processing performed at the tile level. We only provide additional, complementary information where necessary, referring to the code by line number. In addition, Figure 20 presents a graphical representation of the processing performed on tiles that do not contain the camera position (lines 5 to 8 of the pseudo-code). Note that the processing block "convert UVH to AER" performs the coordinate transformation shown in Figure 19.

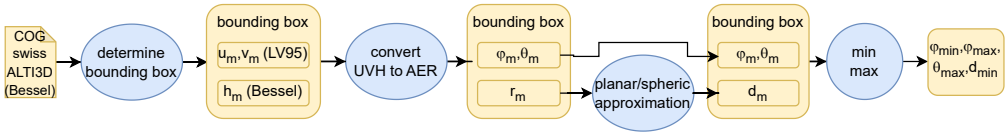


Figure 20. Processing of a tile that does not contain the camera position.

Listing 2. Tile-level processing.

```
1  for each tile T_i:
2      determine u_min, u_max, v_min, v_max
3      if (u_min ≤ u_0 ≤ u_max) ∧ (v_min ≤ v_0 ≤ v_max), current tile is T_0
4  for each tile in {T_i} − {T_0}:
5      determine h_min, h_max
6      build B according to Equation (23)
7      compute φ, θ, d for each (u, v, h) ∈ B
8      compute φ_min, φ_max, θ_max, d_min with Equation (24)
```

Line 2: $u_{\min}, u_{\max}, v_{\min}$, and $v_{\max}$ are easy to determine. If the tile format is ASCII XYZ, they can be determined directly from the file name. If the tile format is COG, they can also be found in the file metadata.

Line 3: If the LV95 coordinates of the camera (u_0, v_0) satisfy this condition, the camera position is within the tile currently considered. This tile is denoted by T_0. As explained before, adjacent tiles do not have common extremal coordinates, so the camera position cannot be in more than one tile. However, the camera position may be between two adjacent tiles. In this case, $\{T_0\}$ is simply the empty set $\emptyset$.

Line 5: Extremal height values $h_{\min}$ and $h_{\max}$ can be easily determined by scanning the file once.

Line 6: With the extremal coordinate values $u_{\min}, u_{\max}, v_{\min}, v_{\max}, h_{\min}$, and $h_{\max}$, we build a tile bounding box, which is a rectangular cuboid in the coordinate system LV95/height. For reasons that will become clear, we use not only the eight vertices

of the rectangular cuboid but also the edges. The set of (u, v, h) coordinates is hence

$$B = (\{u_{\min}, u_{\min} + \Delta u, \cdots, u_{\max}\} \times \{v_{\min}, v_{\max}\}) \cup (\{u_{\min}, u_{\max}\} \times \{v_{\min}, v_{\min} + \Delta v, \cdots, v_{\max}\})) \times \{h_{\min}, h_{\max}\}. \quad (23)$$

Line 7: The (u, v, h) coordinates of the points belonging to B are transformed into the values φ, θ, and d of the local coordinate system, where d is the horizontal distance ϱ in the case of the plane approximation of the DEM horizontal datum and d is the slant range r in the case of spheric approximation.

Line 8: The following extremal φ, θ, and d values are computed for the points belonging to B:

$$\varphi^T_{\min} = \min_{(u,v,h)\in B} \varphi(u, v, h), \quad (24a)$$

$$\varphi^T_{\max} = \max_{(u,v,h)\in B} \varphi(u, v, h), \quad (24b)$$

$$\theta^T_{\max} = \max_{(u,v,h)\in B} \theta(u, v, h), \quad (24c)$$

$$d^T_{\min} = \min_{(u,v,h)\in B} d(u, v, h). \quad (24d)$$

We use ^{T} to denote a tile in order to not confuse these φ and θ values with the extreme LAEM azimuth and elevation values introduced in Section 3.3.3.

Without demonstration, we claim that the values φ_P, θ_P, and d_P of any tile point P satisfy the following:

$$(\varphi^T_{\min} \le \varphi_P \le \varphi^T_{\max}) \wedge (\theta_P \le \theta^T_{\max}) \wedge (d_P \ge d^T_{\min}) \quad (25)$$

Note that this statement is not true if B is reduced to the eight vertices of the rectangular cuboid.

4.6.2. LAEM Computation with Tile Sorting and Filtering

In this section, we describe how the LAEM is computed tile by tile, using the information extracted from the tiles, to decide whether the current tile has to be processed in the manner shown in Listing 1 or not at all, because it does not contain any point visible from the camera position.

The condition for this decision uses the four values computed with Equation (24)—$\varphi^T_{\min}$, $\varphi^T_{\max}$, $\theta^T_{\max}$, and $d^T_{\min}$—and, from the LAEM, the apparent horizon, i.e., the coordinates of the highest elevation for which the LAEM value is not $+\infty$. The azimuth values $\varphi^T_{\min}$ and $\varphi^T_{\max}$ determine an azimuthal sector $\varphi^T_{\min} \le \varphi \le \varphi^T_{\max}$, which we denote by Φ^T. Then, a tile T is processed if the maximal tile elevation $\theta^T_{\max}$ is larger than the minimal elevation of the apparent horizon in the sector Φ^T or the minimal tile distance $d^T_{\min}$ is shorter than the maximal LAEM value on the apparent horizon in the sector Φ^T. A formal description of the condition is provided in the pseudo-code of Listing 3, and accompanying comments.

We keep the information on the apparent horizon in a one-dimensional array $\check{\mathbf{I}}$, the length of which is equal to the number $N_{\varphi'}$ of LAEM columns, containing the indexes of the highest discrete elevations for which the LAEM value is not $+\infty$. The array element at position j, denoted by $\check{i}_j$, is hence formally defined by the following equation:

$$\check{i}_j = \begin{cases} \min(i \mid d_{i,j} < +\infty), & \text{if } \exists i \mid d_{i,j} < +\infty), \\ \text{NaN}, & \text{otherwise.} \end{cases} \quad (26)$$

The minimum function is used instead of the maximum, because of the choice made in Section 3.3.3 of indexing the discrete elevations from top (highest) to bottom (lowest).

The pseudo-code in Listing 3 describes the LAEM computation tile by tile, checking each for whether they have to be processed because they may contain visible data. We only provide complementary information where necessary, referring to the code by line number.

Listing 3. LAEM computation with tile sorting and filtering.

```
1  %Initialise D and Ǐ
2  d_{i,j} = +∞, ∀i, j,      ǐ_j = NaN, ∀j
3  update D and Ǐ with T_0
4  for each tile in {T_i} − {T_0} sorted ascending according to d^T_min:
5      compute j^T_min, j^T_max, i^T_min
6      compute î_{j^T_min, j^T_max}, d̂_{j^T_min, j^T_max}
7      if (i^T_min < î_{j_min, j_max}) ∨ (d^T_min < d̂_{j_min, j_max}):
8          update D and Ǐ with T_i
```

Lines 3 and 8: **D** is updated as described in Listing 1, and $\check{\mathbf{I}}$ by inserting the following instruction after line 11 in Listing 1: if $d_{i_P,j_P} == +\infty, \check{i}_{j_P} = i_P$.

Line 4: Processing without sorting the tiles would yield the same final LAEM but more tiles would need to be processed.

Line 5: From $\varphi^T_{\min}$ and $\varphi^T_{\max}$, the indexes of the corresponding discrete azimuth values $j^T_{\min}$ and $j^T_{\max}$ are computed using Equations (21) and (20a). From $\theta^T_{\max}$, the index of the corresponding discrete elevation value $i^T_{\min}$ is computed using Equation (20b).

Line 6: These are extremal values in the section going from $j^T_{\min}$ to $j^T_{\max}$ of the apparent horizon $\check{\mathbf{I}}$. $\hat{\check{i}}_{j^T_{\min}, j^T_{\max}}$ is the minimal elevation in the section. Formally, this can be expressed as follows:

$$\hat{\check{i}}_{j^T_{\min}, j^T_{\max}} = \begin{cases} \max\limits_{j^T_{\min} \le j \le j^T_{\max}} (\check{i}_j \mid \check{i}_j \neq \text{NaN}), & \text{if } \exists j \mid (j^T_{\min} \le j \le j^T_{\max}) \wedge (\check{i}_j \neq \text{NaN}), \\ \text{NaN}, & \text{otherwise.} \end{cases} \tag{27}$$

$\hat{d}_{j^T_{\min}, j^T_{\max}}$ is the maximal LAEM value in the section, which can be formally expressed as

$$\hat{d}_{j^T_{\min}, j^T_{\max}} = \begin{cases} \max\limits_{j^T_{\min} \le j \le j^T_{\max}} (d_{\check{i}_j, j} \mid \check{i}_j \neq \text{NaN}), & \text{if } \exists j \mid (j^T_{\min} \le j \le j^T_{\max}) \wedge (\check{i}_j \neq \text{NaN}), \\ \text{NaN}, & \text{otherwise.} \end{cases} \tag{28}$$

Line 7: This is the formal expression of the condition expressed with words at the beginning of the section.

4.7. Results

The MATLAB® implementation and experimental results were obtained during the project time (December 2021–September 2023). Only the planar approximation of the DEM horizontal datum was implemented. The LN02-to-geodetic height transformation with REFRAME was tested, as well as the JAR library, but neither were performed on the whole set of preselected tiles, or on the set of those selected in the end.

Figure 21, dating from Sept. 2023, shows the greyscale-coded LAEM obtained from the camera position with an angular resolution of 0.01° in azimuth $\Delta\varphi$, as well as in elevation $\Delta\theta$. The range in azimuth $[\varphi'_{\min}, \varphi'_{\max}]$ is $[0°, 360°]$ and that in elevation $[\theta_{\min}, \theta_{\max}]$ is $[0°, 20°]$. The LAEM array size is 2001 rows by 36,000 columns. It is the size of a 72 megapixel image. A few more than half of the LAEM elements have terrain data, and the rest (in white) represent the sky.

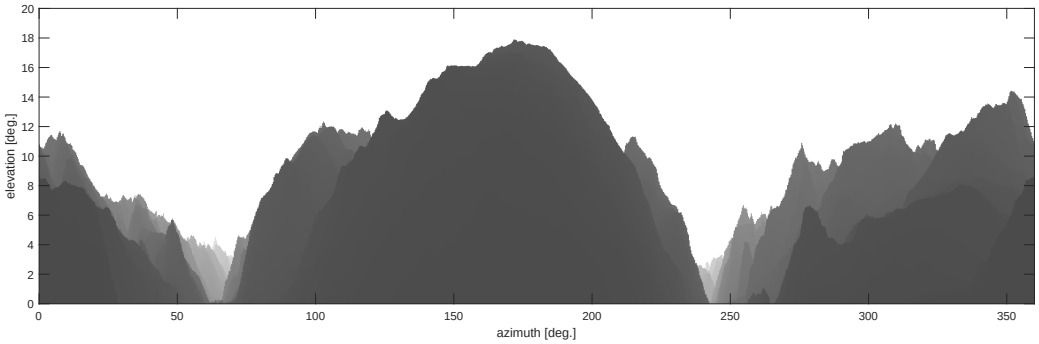


Figure 21. Greyscale-coded LAEM (the darker the shade, the shorter the distance) (see more in Supplementary Materials).

The north–east (azimuth 60°) to south–west (azimuth 240°) orientation of the Rhône Valley can be easily recognised. The highest elevation of the apparent horizon has a value of about 18° and is southwards, at an azimuth of about 170°.

Figure 22 presents an image of the preselected tiles, as well those selected in the end. The LV95 coordinates are presented along the horizontal (west–east) axis and along the vertical (south–north) axis. Each tile is represented by a small square. The tile containing the camera position is the isolated white square, close to coordinates (2,600,000, 1,120,000). The tiles not containing the camera position, which have been fully processed, are in red, and there are 805 of these tiles. The preselected tiles but not processed in the end to fill in the LAEM are in black.

In order to complement the results section during the paper review, the MATLAB® implementation of the LAEM was executed on a Dell Mobile Precision Workstation 3590 with Windows 11 to produce the LAEM presented in Figure 21. The tile-processing times were measured during one full execution (the times were not averaged over several computations of the LAEM). Table 2 summarises the results, showing that most of the computation time was spent processing some selected tiles (max. 7.93 s vs. mean 0.66 s). This concerns the tiles close to the camera position, where several interpolation points have to be computed between two adjacent DEM points, as shown in Figure 11a. The absolute computation times should not be taken as reference, because some parts of the implementation, which are specific to the LAEM computation, have not been optimised yet.

4.8. Result Evaluation

Since the LAEM is basically a coordinate transformation of the DEM points, the procedure used to evaluate the result correctness consists of transforming the LAEM back into the original coordinate system, i.e., LV95 for the horizontal datum and LN02 for the vertical datum, and then comparing the back-transformed LAEM with the original data. In general, the back-transformed points do not fall on the grid coordinates of the original DEM. We consider as correct DEM height at these off-grid horizontal coordinates, the value obtained by performing a linear interpolation on the DEM heights. We used the MATLAB® `interp2` function [65] for this interpolation. Then, the error is the difference in height between the back-transformation and interpolation results.

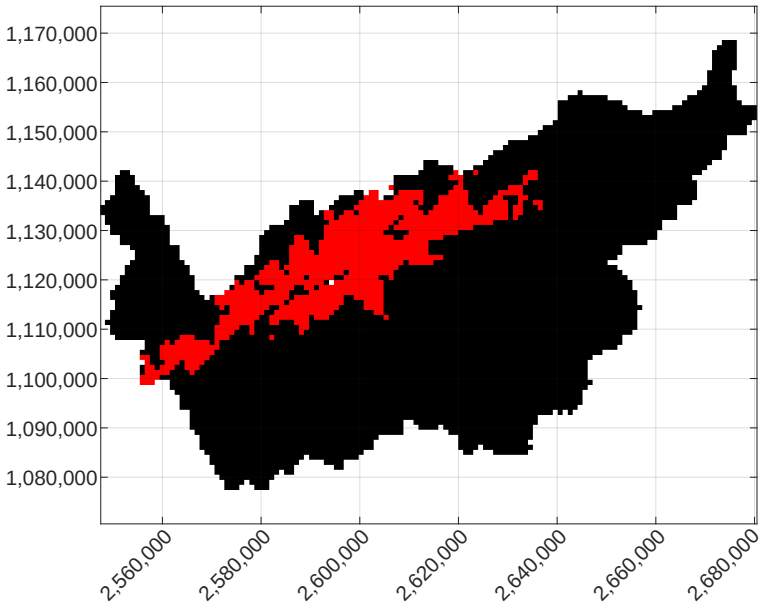


Figure 22. Preselected (black or red) and selected (red) tiles.

Table 2. Tile computation times in seconds.

		Min.	Max.	Mean	Total
Tile-level processing		0.011555	0.020217	0.014606	81.239482
LAEM construction	Selected	0.286338	7.930825	0.662097	536.960388
	Not selected	0.000002	0.001180	0.000012	0.055992

The comparison was performed tile by tile, selecting the back-transformed points based on their horizontal coordinates. Then, a global statistical analysis was performed. For the LAEM presented in Section 4.7, the 37,202,038 back-transformed points have been found to lie in 553 different tiles. There exist height differences larger than 100 m. We noticed that such extreme differences also exist when comparing the heights obtained via linear interpolation with those obtained using another interpolation method. This is to be expected for places where the terrain has cliffs. After truncating the 0.5‰largest negative differences and 0.5 ‰largest positive differences, we obtained a minimal value of −27.82 m, a maximal value of 28.57 m, and a trimmed mean of -0.25 m. Figure 23 shows the central distribution (limited to the interval [−5 m, 5 m]) of the height difference.

We also analysed how the height difference varies from a tile to another. For this, tiles containing a few back-transformed LAEM points were discarded. The minimal number of points was set to 2000. With this condition, the number of tiles was reduced from 553 to 362. For each tile, the 0.5% largest negative differences and the 0.5 % largest positive differences were truncated. The minimal, maximal, and mean were computed for the remaining values. Table 3 presents these statistical values for the 10 tiles closest to and the 10 tiles furthest from the camera. The tile ID is composed of the minimal LV95 coordinates of the tile. The whole table has been made available as an Excel file in the Supplementary Materials.

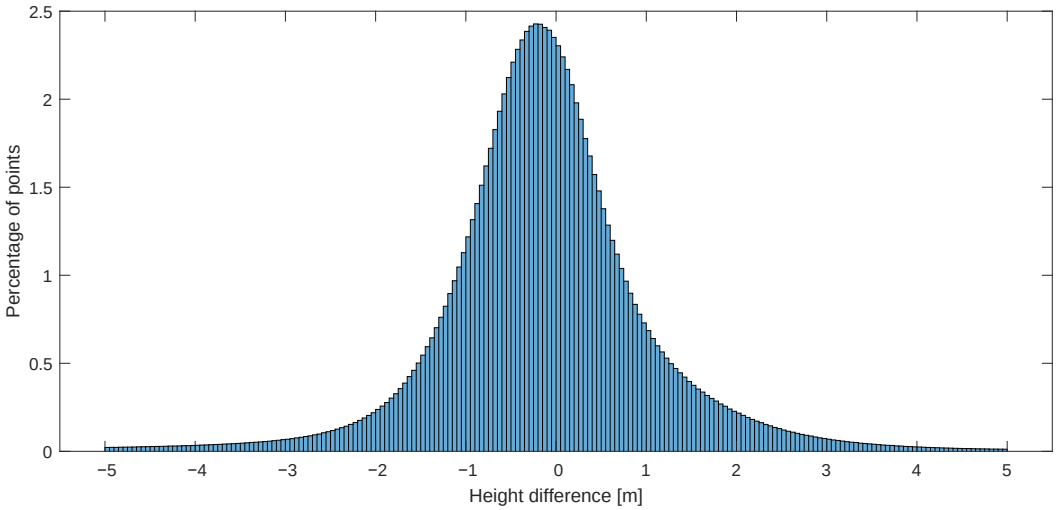


Figure 23. Distribution of the difference between back-transformed and interpolated heights.

Table 3. Height difference by tile (see more in Supplementary Materials).

Tile ID	d_{min} [m]	N pts	N trunc.	0.5% Min. [m]	0.5% Max. [m]	1% Mean [m]
2593-1120	521.9	2,057,832	20,578	−3.88	1.20	−0.35
2594-1120	522.3	2,762,000	27,620	−28.74	1.47	−3.31
2594-1118	630.2	6,283,020	62,830	−3.50	0.97	−0.67
2593-1118	631.5	1,523,272	15,232	−7.91	1.52	−0.72
2595-1119	838.2	251,398	2512	−1.48	0.49	−0.28
2595-1118	909.5	3,553,016	35,530	−3.49	1.19	−0.47
2592-1119	1358.2	657,121	6570	−2.31	1.81	−0.29
2592-1120	1358.9	797,997	7978	−2.78	1.60	−0.21
2592-1118	1403.8	13,677	136	−3.62	1.46	−0.36
2593-1121	1464.3	861,586	8614	−2.99	0.89	−0.42
...	...	...	...	...	...	...
2626-1134	34,854.5	10361	102	−5.86	36.28	7.05
2562-1103	34,947.9	3991	38	−4.44	19.13	4.23
2561-1105	35,027.7	3267	32	−1.07	11.61	4.45
2626-1135	35,281.4	5416	54	−1.59	26.18	6.59
2561-1103	35,848.9	10,000	100	−8.61	17.63	5.13
2628-1132	35,942	3260	32	−0.07	24.50	6.35
2627-1135	36,185.7	2885	28	−1.23	18.56	3.90
2631-1133	39,105.2	3967	38	−3.01	24.85	8.21
2631-1137	40,659.7	2876	28	−11.44	50.99	8.96
2631-1138	41,098.6	3157	30	−2.15	25.06	5.78

Figure 24 shows the 1% trimmed mean value at the geographic position of the tile. The camera position is represented as a black point close to the line of northing coordinate 1,112,000. Obviously, the mean height difference has a positive value that increases with the distance to the camera position. The error in height due to the quantisation of the elevation angle increases with the distance to the reference point, but this should not affect the mean value, only the deviation. The observed increase of the mean value is certainly due to the limitations of the planar approximation vs. the spheric approximation (see Section 3.4.3). We cannot verify this claim because the spheric approximation has not been implemented yet.

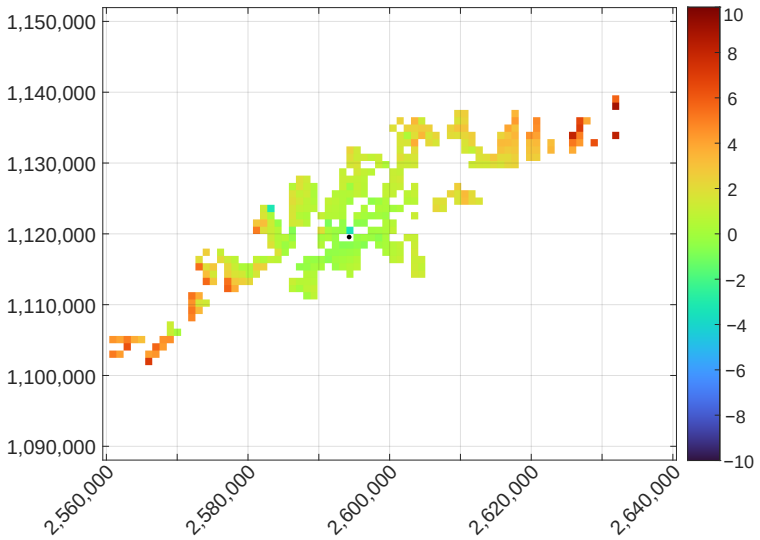


Figure 24. The 1% trimmed mean height difference by tile.

5. Discussion

In this paper, a new representation of a digital elevation model (DEM), called the local azimuth–elevation map (LAEM), is proposed, as well as a method for computing this representation. The method was implemented in MATLAB® and tested on the swissALTI3D DEM published by the Swiss Federal Office of Topography swisstopo. This implementation required completing the general method with some processing specific to the DEM provided by swisstopo. That is, the Swiss-specific coordinates were transformed into geodetic coordinates and the tiling of the DEM was worked around. Both the method principle and its implementation are described with sufficient details in order to analyse their correctness and ensure their precise reproduction by any interested reader.

The LAEM representation and the method for its computation were developed with the aim to to calibrate a PTZ camera using DEM data. Notwithstanding this goal, the LAEM representation offers the following advantages:

- It contains only the portion of a DEM that is visible from a given position.
- It is not a subset in any order, but DEM points are arranged with a spatial consistency.
- The dimensionality is reduced from 3D to 2D. The LAEM is comparable to an image of the DEM but with distance information (horizontal distance ϱ or slant range r) instead of apparent colour, and another 2D coordinate system.
- In the case using spheric approximation, the scalar value in the 2D space, the slant range r, is rotation-invariant.

Regarding the project goal (camera calibration), where the key problem is to find corresponding points in the camera image and the DEM, the new representation is an important step towards a problem solution for the following reason: it narrows the set of DEM candidates and arranges them in 2D in a manner consistent with the arrangement of their projection in the image.

An important point to mention concerning the targeted camera calibration is that the universally used camera matrix model assumes world points to be in a Cartesian coordinate system. However, the LAEM representing world points in a spheric coordinate system is not an obstacle since the LAEM points can easily be transformed to the local ENU Cartesian system if the camera position is assumed to be sufficiently precise, or to the global ECEF Cartesian system, as accomplished by Milosavljević et al. [32] if a more precise camera position is desired.

Of course, the LAEM usage can only be effective for camera calibration as long as the DEM is a valid model of the scene. A DEM like swissALTI3D is a terrain model that describes the Earth's surface without vegetation or development features [6]. Such a model is convenient when the scene is a natural landscape with no or little vegetation such as that in cold climates. It is also a valid model of a landscape that is far enough from the camera position so that the height of the vegetation or any anthropogenic objects do not significantly change the elevation angle. However, DEMs like the swissSURFACE3D Raster [66] are also available, which represent the Earth's surface, including visible landscape elements such as soil, natural cover, and all sorts of constructive work with the exception of power lines and masts.

Finally, this paper proposes a few ideas on how the correspondence between the camera image and LAEM could be established:

Depth map: The LAEM is a depth map, similar to the output of a LiDAR. Fusion techniques for LiDAR and camera data [67] can be used. In particular, one could work on the similarity between the C^0 discontinuities [68] of the LAEM and image edges [9]. Note that this is a generalisation of the approach used by Portenier et al. [33], who used the similarity of the apparent horizon in the image and the one in a synthetic image generated by the DEM.

GIS-augmented LAEM: In this case, the end goal of camera calibration is to georeference a camera image so that is enriched with GIS information. Changing the ordering of the procedure could be helpful in finding correspondence between the camera image and LAEM, i.e., the LAEM is first enriched with GIS information, e.g., roads, public lights, ... An orthophoto could also be used to complement the LAEM in this case; note that the drawback of doing so was discussed in Section 2.

Self-calibration: Self-calibration can be performed in the case of a PTZ camera. Doing so generates a 3D point cloud. The camera pose is relative to this 3D point cloud. It is possible to align the 3D point cloud with the LAEM and, thereby, obtain the pose of the camera in a geographic coordinate system. To align the 3D point cloud generated through the self-calibration with the LAEM, the well-known RANSAC algorithm [9] can be used. Another idea is to transform the 3D point cloud into AER coordinates and use the rotation invariance of the range component r to preselect points of same r-value in the 3D point cloud and in the LAEM that could correspond to each other.

Finally, in addition to researching solutions for establishing correspondences between the camera image and LAEM, the author intends to improve the LAEM computation method by integrating a downsampling procedure, as mentioned in Section 3.4.5.

Supplementary Materials: The following supporting information can be downloaded at: https://www.mdpi.com/article/10.3390/ijgi15030131/s1, the file ch.swisstopo.swissalti3d-Valais_2 20220330.csv containing the list of swissALTI3D files used in the experiments; the MATLAB® m-file SWISSGD_download_maps_listed_in_csv_file.m to download the swissALTI3D files to a local folder, the m-files ALTI3D_convert_COG_to_XYZ.m and ALTI3D_convert_XYZ_to_COG.m to convert swissALTI3D files between COG and XYZ formats; the MATLAB® MAT-file D.mat containing the LAEM shown in Figure 21: Greyscale-coded LAEM; the Excel file statistics_height_differences_by_tile.xlsx containing the full table, from which the first 10 and last 10 rows are shown in Table 3: Height difference by tile.

Funding: This research was conducted from December 2021 to September 2023 when I was with the HES-SO Valais-Wallis Institute of Systems Engineering. This work was funded by the HES-SO University of Applied Sciences and Arts Western Switzerland and its Engineering and Architecture faculty under the title "Calibration d'une caméra PTZ à l'aide d'un modèle numérique de terrain" and grant number 115161. The paper summary was written at that time. The contents of this article were drafted in late 2025 and revised in early 2026 without external funding.

Data Availability Statement: The original data from the Swiss Federal Office of Topography swisstopo used in the experiments are openly available at https://www.swisstopo.admin.ch/en/height-model-swissalti3d (accessed on 1 November 2025). A set of images that were taken with the camera set-up described in the experimental section is freely available from the B2SHARE repository at: https://doi.org/10.23728/b2share.1474166f2e024413a181089253d58a83.

Acknowledgments: I would like to thank the persons who encouraged me to write this publication, especially my wife. Additional thanks go to the reviewers, who pointed out issues in my original text and helped me improve this publication.

Conflicts of Interest: The author declares no conflicts of interest.

Abbreviations

The following abbreviations are used in this manuscript:

2D	two-dimensional
3D	three-dimensional
AER	azimuth–elevation–range
ASCII	American Standard Code for Information Interchange
COG	Cloud Optimized GeoTIFF
DEM	Digital Elevation Model
ECEF	Earth-centred, Earth-fixed
ESRI	Environmental Systems Research Institute, Inc.
FOV	field of view
GIS	geographic information system
GRS80	Geodetic Reference System 1980
LGH	local geodetic horizon
LGHS	local geodetic horizon system
LAEM	local azimuth–elevation map
LN02	Swiss national levelling network 1902
LV95	Swiss national survey coordinate system 1995
PTZ	pan–tilt–zoom
ROI	region of interest
UTC	Coordinated Universal Time

References

1. Meyer, T. Grid, ground, and globe: Distances in the GPS era. *Surv. Land Inf. Syst.* **2002**, *62*, 179–202.
2. Van Sickle, J. *Basic GIS Coordinates*, 3rd ed.; CRC Press, Taylor & Francis Group: Boca Raton, FL, USA, 2017.
3. Hastings, J.T.; Hill, L.L. Georeferencing. In *Encyclopedia of Database Systems*; Liu, L., Özsu, M.T., Eds.; Springer: Boston, MA, USA, 2009; pp. 1246–1249. [CrossRef]
4. Shekhar, S.; Xiong, H.; Zhou, X. (Eds.) *Encyclopedia of GIS*, 2nd ed.; Springer: Cham, Switzerland, 2017. [CrossRef]
5. Guth, P.L.; Van Niekerk, A.; Grohmann, C.H.; Muller, J.P.; Hawker, L.; Florinsky, I.V.; Gesch, D.; Reuter, H.I.; Herrera-Cruz, V.; Riazanoff, S.; et al. Digital Elevation Models: Terminology and Definitions. *Remote Sens.* **2021**, *13*, 3581. [CrossRef]
6. Federal Office of Topography Swisstopo. swissALTI3D. Available online: https://www.swisstopo.admin.ch/en/height-model-swissalti3d (accessed on 1 November 2025).
7. EuroGeographics AISBL. Open Maps for Europe|Eurogeographics. Available online: https://www.mapsforeurope.org/datasets/euro-dem (accessed on 30 October 2025).
8. McGlone, J.C. (Ed.) *Manual of Photogrammetry*, 6th ed.; American Society for Photogrammetry and Remote Sensing: Bethesda, MD, USA, 2013.
9. Szeliski, R. *Computer Vision: Algorithms and Applications*, 2nd ed.; The University of Washington: Seattle, WA, USA, 2022. Available online: https://szeliski.org/Book (accessed on 3 November 2025).
10. Hartley, R.; Zisserman, A. *Multiple View Geometry in Computer Vision*, 2nd ed.; Cambridge University Press: Cambridge, UK, 2004. [CrossRef]
11. Akenine-Möller, T.; Haines, E.; Hoffman, N.; Pesce, A.; Iwanicki, M.; Hillaire, S. *Real-Time Rendering*, 4th ed.; A K Peters/CRC Press: Boca Raton, FL, USA, 2018; p. 1200.

12. Huai, J.; Shao, Y.; Jozkow, G.; Wang, B.; Chen, D.; He, Y.; Yilmaz, A. Geometric Wide-Angle Camera Calibration: A Review and Comparative Study. *Sensors* **2024**, *24*, 6595. [CrossRef] [PubMed]
13. Slama, C.C., Ed. *Manual of Photogrammetry*, 4th ed.; American Society for Photogrammetry: Falls Church, VA, USA, 1980.
14. Bösch, J.; Goswami, P.; Pajarola, R. RASTeR: Simple and Efficient Terrain Rendering on the GPU. In *Proceedings of the Eurographics 2009—Areas Papers*; Ebert, D., Krüger, J., Eds.; The Eurographics Association: Eindhoven, The Netherlands, 2009. [CrossRef]
15. Remondino, F.; Fraser, C. Digital Camera Calibration Methods: Considerations and Comparisons. In *Proceedings of the ISPRS Commission V Symposium 'Image Engineering and Vision Metrology'*; Maas, H.G., Schneider, D., Eds.; International Society for Photogrammetry and Remote Sensing: Dresden, Germany, 2006; pp. 266–272. Available online: https://www.isprs.org/proceedings/xxxvi/part5/paper/remo_616.pdf (accessed on 8 November 2025).
16. Hieronymus, J. Comparison of methods for geometric camera calibration. *ISPRS-Archives* **2012**, *XXXIX-B5*, 595–599. [CrossRef]
17. Salvi, J.; Armangué, X.; Batlle, J. A comparative review of camera calibrating methods with accuracy evaluation. *Pattern Recognit.* **2002**, *35*, 1617–1635. [CrossRef]
18. Long, L.; Dongri, S. Review of camera calibration algorithms. In *Proceedings of the Advances in Computer Communication and Computational Sciences*; Bhatia, S.K., Tiwari, S., Mishra, K.K., Trivedi, M.C., Eds.; Springer: Singapore, 2019; pp. 723–732.
19. Lensch, H.P.A.; Heidrich, W.; Seidel, H.P. A Silhouette-Based Algorithm for Texture Registration and Stitching. *Graph. Model.* **2001**, *63*, 245–262. [CrossRef]
20. Troccoli, A.J.; Allen, P.K. A Shadow Based Method for Image to Model Registration. In Proceedings of the 2004 Conference on Computer Vision and Pattern Recognition Workshop, Washington, DC, USA, 27 June–2 July 2004; p. 169. [CrossRef]
21. Feng, M.; Hu, S.; Ang, M.; Lee, G.H. 2D3D-MatchNet: Learning to Match Keypoints Across 2D Image and 3D Point Cloud. *arXiv* **2019**, arXiv:1904.09742. [CrossRef]
22. Pham, Q.H.; Uy, M.A.; Hua, B.S.; Nguyen, D.T.; Roig, G.; Yeung, S.K. LCD: Learned Cross-Domain Descriptors for 2D-3D Matching. *arXiv* **2019**, arXiv:1911.09326. [CrossRef]
23. Wang, B.; Chen, C.; Cui, Z.; Qin, J.; Lu, C.X.; Yu, Z.; Zhao, P.; Dong, Z.; Zhu, F.; Trigoni, N.; et al. P2-Net: Joint Description and Detection of Local Features for Pixel and Point Matching. *arXiv* **2021**, arXiv:2103.01055. [CrossRef]
24. Li, M.; Qin, Z.; Gao, Z.; Yi, R.; Zhu, C.; Guo, Y.; Xu, K. 2D3D-MATR: 2D-3D Matching Transformer for Detection-free Registration between Images and Point Clouds. *arXiv* **2023**, arXiv:2308.05667. [CrossRef]
25. Wilson, D.; Zhang, X.; Sultani, W.; Wshah, S. Image and Object Geo-Localization. *Int. J. Comput. Vis.* **2024**, *132*, 1350–1392. [CrossRef]
26. Rameau, F.; Choe, J.; Pan, F.; Lee, S.; Kweon, I. CCTV-Calib: A toolbox to calibrate surveillance cameras around the globe. *Mach. Vis. Appl.* **2023**, *34*, 125. [CrossRef]
27. Shan, Q.; Wu, C.; Curless, B.; Furukawa, Y.; Hernandez, C.; Seitz, S.M. Accurate Geo-Registration by Ground-to-Aerial Image Matching. In Proceedings of the 2014 2nd International Conference on 3D Vision, Tokyo, Japan, 8–11 December 2014; Volume 1, pp. 525–532. [CrossRef]
28. Li, Y.; Snavely, N.; Huttenlocher, D.; Fua, P. Worldwide Pose Estimation Using 3D Point Clouds. In *Proceedings of the Computer Vision—ECCV 2012*; Fitzgibbon, A., Lazebnik, S., Perona, P., Sato, Y., Schmid, C., Eds.; Springer: Berlin/Heidelberg, Germany, 2012; pp. 15–29. [CrossRef]
29. Li, Y.; Snavely, N.; Huttenlocher, D.P. Location Recognition Using Prioritized Feature Matching. In *Proceedings of the Computer Vision—ECCV 2010*; Daniilidis, K., Maragos, P., Paragios, N., Eds.; Springer: Berlin/Heidelberg, Germany, 2010; pp. 791–804. [CrossRef]
30. Sattler, T.; Leibe, B.; Kobbelt, L. Fast image-based localization using direct 2D-to-3D matching. In Proceedings of the 2011 International Conference on Computer Vision, Barcelona, Spain, 6–13 November 2011; pp. 667–674. [CrossRef]
31. Härer, S.; Bernhardt, M.; Corripio, J.G.; Schulz, K. PRACTISE - Photo Rectification And ClassificaTIon SoftwarE (V.1.0). *Geosci. Model Dev.* **2013**, *6*, 837–848. [CrossRef]
32. Milosavljević, A.; Rančić, D.; Dimitrijević, A.; Predić, B.; Mihajlović, V. A Method for Estimating Surveillance Video Georeferences. *ISPRS Int. J. Geo-Inf.* **2017**, *6*, 211. [CrossRef]
33. Portenier, C.; Hüsler, F.; Härer, S.; Wunderle, S. Towards a webcam-based snow cover monitoring network: Methodology and evaluation. *Cryosphere* **2020**, *14*, 1409–1423. [CrossRef]
34. PTZOptics. CAD Line Drawings and 3D Renders—PTZOptics. Available online: https://ptzoptics.com/cad-line-drawings/ (accessed on 25 November 2025).
35. Google. Google Maps. Available online: https://maps.google.com (accessed on 30 November 2025)).
36. The MathWorks, Inc. Comparison of 3-D Coordinate Systems—MATLAB & Simulink. Available online: https://www.mathworks.com/help/map/choose-a-3-d-coordinate-system.html (accessed on 19 November 2025).
37. The MathWorks, Inc. geodetic2enu—Transform Geodetic Coordinates to Local East-North-Up—MATLAB. Available online: https://www.mathworks.com/help/map/ref/geodetic2enu.html (accessed on 19 November 2025).

38. pymap3d.enu API Documentation. Available online: https://geospace-code.github.io/pymap3d/enu.html#pymap3d.enu.geodetic2enu (accessed on 19 November 2025).
39. The MathWorks, Inc. enu2geodetic—Transform Local East-North-Up Coordinates to Geodetic—MATLAB. Available online: https://www.mathworks.com/help/map/ref/enu2geodetic.html (accessed on 5 December 2025).
40. pymap3d.enu API Documentation. Available online: https://geospace-code.github.io/pymap3d/enu.html#pymap3d.enu.enu2geodetic (accessed on 5 December 2025).
41. Roy, A.E.; Clarke, D. *Astronomy: Principles and Practice*, 4th ed.; CRC Press, Taylor & Francis Group: Boca Raton, FL, USA, 2003.
42. The MathWorks, Inc. geodetic2aer—Transform Geodetic Coordinates to Local Spherical–MATLAB. Available online: https://www.mathworks.com/help/map/ref/geodetic2aer.html (accessed on 19 November 2025).
43. pymap3d.aer API Documentation. Available online: https://geospace-code.github.io/pymap3d/aer.html#pymap3d.aer.geodetic2aer (accessed on 19 November 2025).
44. The MathWorks, Inc. aer2geodetic—Transform Local Spherical Coordinates to Geodetic—MATLAB. Available online: https://www.mathworks.com/help/map/ref/aer2geodetic.html (accessed on 6 December 2025).
45. pymap3d.aer API Documentation. Available online: https://geospace-code.github.io/pymap3d/aer.html#pymap3d.aer.aer2geodetic (accessed on 6 December 2025).
46. The MathWorks, Inc. Image Coordinate Systems—MATLAB & Simulink. Available online: https://www.mathworks.com/help/images/image-coordinate-systems.html (accessed on 2 December 2025).
47. OpenCV Team. OpenCV: Operations with images. Available online: https://docs.opencv.org/4.12.0/d5/d98/tutorial_mat_operations.html#autotoc_md342 (accessed on 2 December 2025).
48. Roth, S.D. Ray casting for modeling solids. *Comput. Graph. Image Process.* **1982**, *18*, 109–144. [CrossRef]
49. Wikimedia Foundation. "Extremes on Earth—Wikipedia". Available online: https://en.wikipedia.org/wiki/Extremes_on_Earth#Greatest_vertical_drop (accessed on 14 December 2025).
50. Holton, T. *Digital Signal Processing: Principles and Applications*; Cambridge University Press: Cambridge, UK, 2021. [CrossRef]
51. Federal Office of Topography Swisstopo. The Swiss Coordinates LV95. 2024. Available online: https://www.swisstopo.admin.ch/en/the-swiss-coordinates-system (accessed on 21 December 2025).
52. Federal Office of Topography Swisstopo. Swiss National Levelling Network LN02. 2025. Available online: https://www.swisstopo.admin.ch/en/swiss-national-levelling-network-ln02 (accessed on 21 December 2025).
53. Bundesamt für Landestopographie Swisstopo. swissALTI3D: Das Hoch Aufgelöste Terrainmodell der Schweiz. Technical Report, Bundesamt für Landestopographie Swisstopo. 2022. Available online: https://www.swisstopo.admin.ch/dam/de/sd-web/D6hcUzfZuiQc/swissALTI3D-ProdInfo-DE.pdf (accessed on 14 February 2026).
54. Office Fédéral de Topographie Swisstopo. swissALTI3D: Le Modèle de Terrain à Haute réSolution de la Suisse. Technical Report, Office fédéRal de Topographie Swisstopo. 2022. Available online: https://www.swisstopo.admin.ch/dam/fr/sd-web/D6hcUzfZuiQc/swissALTI3D-ProdInfo-FR.pdf (accessed on 14 February 2026).
55. Axis Communications AB. AXIS M5525-E PTZ Network Camera—Product Support | Axis Communications. Available online: https://www.axis.com/products/axis-m5525-e/support (accessed on 17 December 2025).
56. Maître, G. PTZcalDB Public (1.0). EUDAT Collaborative data infrastructures. *EUDAT* **2024** . [CrossRef]
57. Federal Office of Topography Swisstopo. Maps of Switzerland—Swiss Confederation—map.geo.admin.ch. Available online: https://map.geo.admin.ch (accessed on 18 December 2025).
58. Federal Office of Topography Swisstopo. Geoid: The Swiss Geoid Model CHGeo2004. Available online: https://www.swisstopo.admin.ch/en/geoid-en (accessed on 19 December 2025).
59. Federal Office of Topography Swisstopo. REFRAME. Available online: https://www.swisstopo.admin.ch/en/coordinate-conversion-reframe (accessed on 19 December 2025).
60. Federal Office of Topography Swisstopo. REST Web Geoservices (REFRAME Web API). Available online: https://www.swisstopo.admin.ch/en/rest-api-geoservices-reframe-web (accessed on 19 December 2025).
61. Federal Office of Topography Swisstopo. REFRAME DLL/JAR. Available online: https://cms.geo.admin.ch/ogd/geodesy/reframedll.zip (accessed on 19 December 2025).
62. Federal Office of Topography Swisstopo. GeoSuite (LTOP/REFRAME/TRANSINT). Available online: https://www.swisstopo.admin.ch/en/geodetic-software-geosuite (accessed on 19 December 2025).
63. The MathWorks, Inc. Projinv—Unproject x-y Map Coordinates to Latitude-Longitude Coordinates—MATLAB. Available online: https://www.mathworks.com/help/map/ref/projcrs.projinv.html (accessed on 19 December 2025).
64. The MathWorks, Inc. Readgeoraster–Read Geospatial Raster Data File—MATLAB. Available online: https://www.mathworks.com/help/map/ref/readgeoraster.html (accessed on 20 December 2025).
65. The MathWorks, Inc. interp2—Interpolation for 2-D Gridded Data in Meshgrid Format—MATLAB. Available online: https://www.mathworks.com/help/matlab/ref/interp2.html (accessed on 14 February 2026).

66. Federal Office of Topography Swisstopo. swissSURFACE3D Raster. Available online: https://www.swisstopo.admin.ch/en/height-model-swisssurface3d-raster (accessed on 13 February 2026).
67. Thakur, A.; Rajalakshmi, P. LiDAR and Camera Raw Data Sensor Fusion in Real-Time for Obstacle Detection. In Proceedings of the 2023 IEEE Sensors Applications Symposium (SAS), Ottawa, ON, Canada, 18–20 July 2023; pp. 1–6. [CrossRef]
68. An, Y.; Shao, C.; Li, Z.; Zhuang, Y.; Yan, Y. Discontinuity Identification from Range Data Based on Similarity Indicators. *IFAC Proc. Vol.* **2011**, *44*, 3153–3158. [CrossRef]