International Journal of Geo-Information

MDPI

Article

Generating Orthorectified Multi-Perspective 2.5D Maps to Facilitate Web GIS-Based Visualization and Exploitation of Massive 3D City Models

Jianming Liang [1,2], Jianhua Gong [1,2,*], Jin Liu [1,3,4], Yuling Zou [2], Jinming Zhang [1,5], Jun Sun [1,2] and Shuisen Chen [6]

1 State Key Laboratory of Remote Sensing Science, Institute of Remote Sensing and Digital Earth, Chinese Academy of Sciences, Beijing 100101, China; ljm355@163.com (J.L.); 13512007804@163.com (J.L.); zjmwh@vip.sohu.com (J.Z.); sjunme@126.com (J.S.)
2 Zhejiang-CAS Application Center for Geoinformatics, Zhejiang 314100, China; elaine63@139.com
3 University of Chinese Academy of Sciences, Beijing 100049, China
4 National Marine Data and Information Service, Tianjin 300171, China
5 Institute of Geospatial Information, Information Engineering University, Zhengzhou 450052, China
6 Guangdong Open Laboratory of Geospatial Information Technology and Application, Guangzhou Institute of Geography, Guangzhou 510070, China; css@gdas.ac.cn
* Correspondence: gongjh@radi.ac.cn; Tel.: +86-10-6484-9299

Academic Editors: Bert Veenendaal, Maria Antonia Brovelli, Serena Coetzee, Peter Mooney and Wolfgang Kainz
Received: 5 August 2016; Accepted: 7 November 2016; Published: 11 November 2016

Abstract: 2.5D map is a convenient and efficient approach to exploiting a massive three-dimensional (3D) city model in web GIS. With the rapid development of oblique airborne photogrammetry and photo-based 3D reconstruction, 3D city models are becoming more and more accessible. 3D Geographic Information System (GIS) can support the interactive visualization of massive 3D city models on various platforms and devices. However, the value and accessibility of existing 3D city models can be augmented by integrating them into web-based two-dimensional (2D) GIS applications. In this paper, we present a step-by-step workflow for generating orthorectified oblique images (2.5D maps) from massive 3D city models. The proposed framework can produce 2.5D maps from an arbitrary perspective, defined by the elevation angle and azimuth angle of a virtual orthographic camera. We demonstrate how 2.5D maps can benefit web-based visualization and exploitation of massive 3D city models. We conclude that a 2.5D map is a compact data representation optimized for web data streaming of 3D city models and that geometric analysis of buildings can be effectively conducted on 2.5D maps.

Keywords: massive 3D city model; 2D GIS; web GIS; 2.5D map; oblique airborne photogrammetry

1. Introduction

Urban landscapes are highly diversified in both the horizontal and vertical dimensions [1] with the presence of water bodies, plants, buildings, and other artificial structures. The inadequacy of satellite imagery and digital elevation models (DEMs) for representing the vertical dimension led to the widespread use of 3D city models [2] in urban GIS, which typically takes the form of a virtual geographic environment [3].

Although 3D GIS is already widely used for interactive exploration of 3D city models, its use in a web context faces the following technical challenges: (1) Internet data transfer limits. For example, it takes 17 min to download 1 GB of 3D model data with an internet transfer rate of 1 MB/s. This means that users will likely experience data latency under limited bandwidth. (2) Cross-platform hardware and software compatibility. The devices and platforms that host a 3D GIS application need to meet

the minimum hardware and software requirements for graphics processing unit (GPU) support. Mobile devices do not always meet the computational requirements of a 3D GIS [4]. Deployment of a 3D GIS across various platforms, such as Linux, Windows, Android, iOS, and Mac, requires extra development cost and effort, since 3D graphics APIs rarely support direct cross-platform migration. Additionally, web-based 2D GIS remains the dominant paradigm in operational business applications. Many business GIS applications, such as the hotel booking website Agoda (http://www.agoda.com/), rely solely on web-based 2D GIS for operational functionality. 2.5D maps, also known as perspective maps or bird's-eye view maps, are represented in an oblique perspective. 2.5D cartography dates back to ancient times (Figure 1). Modern 2.5D cartography, known as pixel art in some contexts, integrates computer-aided design (CAD)-based photorealistic or artistic renderings into maps. 2.5D maps can facilitate visualization and exploitation of 3D city models in web-based 2D GIS applications. The 2.5D maps provider Edushi produces pixel art-style maps from CAD models and delivers these maps to a browser-based web GIS (Figure 2A), which can run on various devices and platforms. These CAD-based pixel art-style maps are also widely used in campus map GIS (Figure 2B) (http://maps.colostate.edu/, https://www.udayton.edu/map/, http://map.wfu.edu/, https://www.asu.edu/map/interactive/), which makes high-quality 3D scenes available to users under conditions of limited bandwidth and computational power. If large-scale 3D city models can be used in web GIS as shown in those examples (Figure 2A,B), they will attract a wider audience and benefit a wider scope of applications. Therefore, 2.5D maps have the potential to augment the value and extend the use of existing 3D city models.

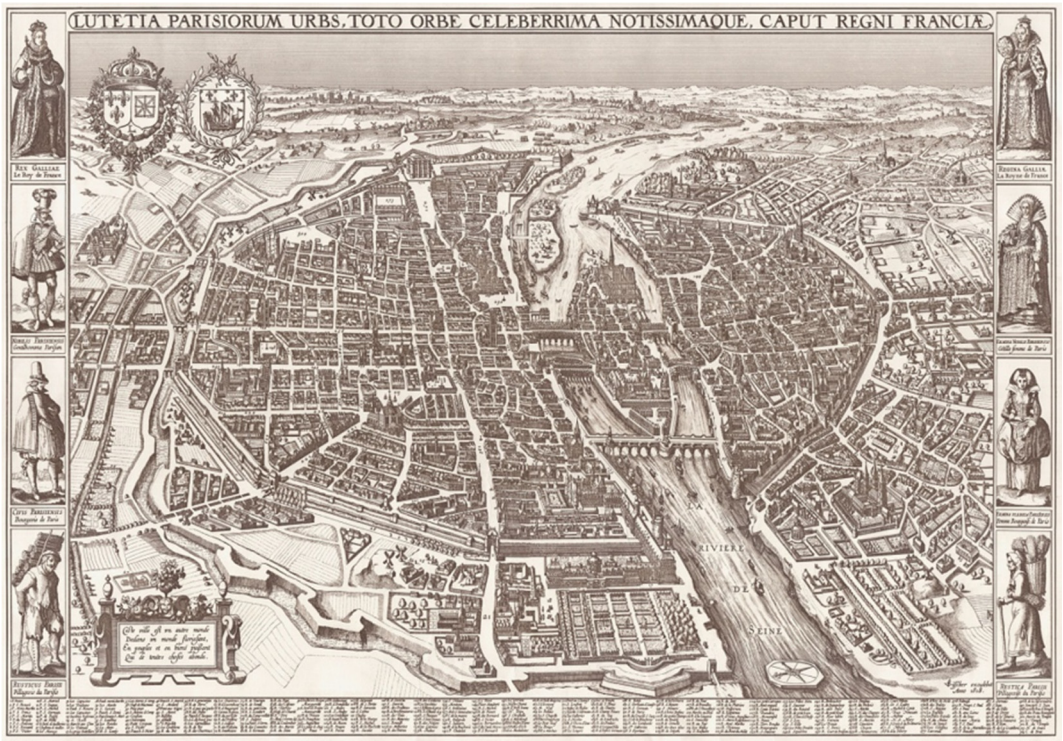


Figure 1. Hand-drawn perspective map by Claes Janszoon Visscher (http://en.wikipedia.org/wiki/Pictorial_maps).

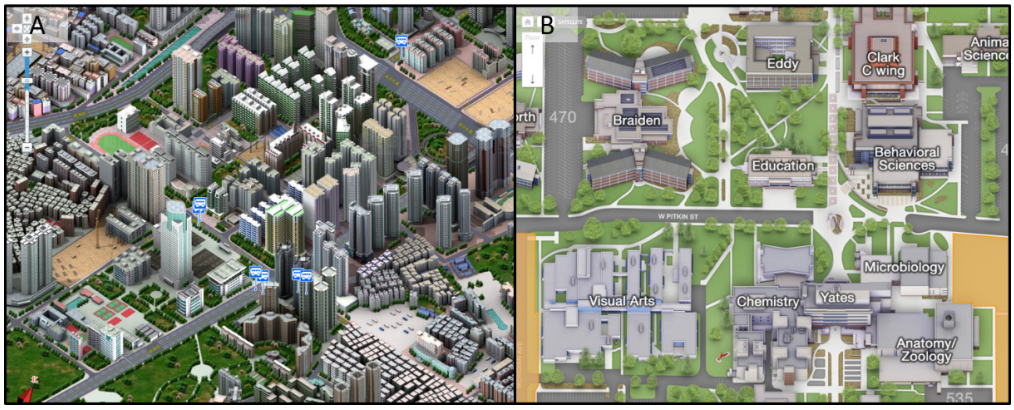


Figure 2. Application of 2.5D maps in (**A**) a web-based urban GIS (http://sz.edushi.com/); (**B**) an online campus GIS (http://maps.colostate.edu/) for efficient streaming of 3D city models.

3D city models can be derived from a variety of sources. One of the most simplified forms of a 3D city model is obtained by extruding building footprints into polyhedra [5]. A LiDAR point cloud can accurately capture architectural geometry, but the textural information needs to be retrieved separately [6]. Photorealistic building models can be constructed in a CAD environment, but the intensive labor required to create the geometric details and to acquire street-level façade photos is challenging. Recently, with the rapid development of unmanned aerial vehicle (UAV) technology and computer vision-based 3D reconstruction [7], oblique airborne photogrammetry-based 3D city models (OAP3Ds) are becoming increasingly mature and affordable [8]. OAP3Ds combine the advantages of LiDAR and close-range photography, and thus are enriched in both geometric and textural detail.

In the early stages of 3D GIS development, the use of 3D city models was limited primarily to visualization and urban planning [9,10], but recently it has expanded well beyond this scope. A bibliometric survey [11] identified 29 distinct use cases from different application domains and grouped these use cases into two broad categories, i.e., visualization-centered and non-visualization-centered. Out of the 29 use cases, 24 are visualization-centered and only five are non-visualization-centered (Table 1). This suggests that the core value of 3D city models is embodied primarily in visualization, although a variety of non-visualization purposes are also served.

Table 1. Overview of the use cases of 3D city models (adapted from Biljecki et al. [11]).

1.	Estimation of solar irradiation [12–15]
2.	Energy demand estimation [16,17]
3.	Aiding positioning
4.	Determination of floor space [18,19]
5.	Classifying building types [20]
6.	Geo-visualization and visualization enhancement [2,21]
7.	Visibility analysis [22]
8.	Estimation of shadows cast by urban features [23]
9.	Estimation of the propagation of noise in an urban environment
10.	3D cadaster [24]
11.	Visualization for navigation [25]
12.	Urban planning [8–10]
13.	Visualization for communication of urban information to citizenry [25,26]
14.	Reconstruction of sunlight direction
15.	Understanding synthetic aperture radar images
16.	Facility management [27]
17.	Automatic scaffold assembly civil engineering
18.	Emergency response [28]
19.	Lighting simulations
20.	Radio-wave propagation

Table 1. *Cont.*

21.	Computational fluid dynamics
22.	Estimating the population in an area [29]
23.	Routing [30]
24.	Forecasting seismic damage
25.	Flooding [31,32]
26.	Change detection [33]
27.	Volumetric density studies
28.	Forest management
29.	Archaeology

Massive 3D city models are embedded in a level-of-detail (LOD) structure optimized for view-dependent out-of-core rendering, and thus cannot be easily brought into a CAD environment for making 2.5D maps. Typical formats for representing massive 3D city models include CityGML and OAP3D. Additionally, the methods and applications of 2.5D cartography have not been systematically addressed in the literature.

In this study, we present a streamlined framework (Figure 3) for transforming massive 3D city models into 2.5D maps. In the proposed framework (Figure 3), we aim to achieve the following functional objectives:

1. Support for various 3D model formats. 3D models come in a wide variety of formats, including .3ds, .obj and .dae.
2. Support for LOD structure and spatial partitioning. Given limited computational power, a well-designed spatial-partitioning and out-of-core rendering scheme must be employed to generate 2.5D maps at a sufficiently high spatial resolution. By leveraging an LOD structure, a large 3D city model can be rendered into a grid of map tiles, but these map tiles need be accurately georeferenced so they can be stitched back together to form a seamless 2.5D mosaic.
3. Support for custom map perspectives. An oblique perspective is defined by a camera's azimuth and elevation angle. A multi-perspective set of 2.5D maps can potentially afford a complete view of a 3D city model.
4. Support for orthorectification. The presence of terrain and the use of an oblique perspective may subject a set of 2.5D maps to distortion and misalignment. Orthorectification brings a multi-perspective set of 2.5D maps back into a common reference system.

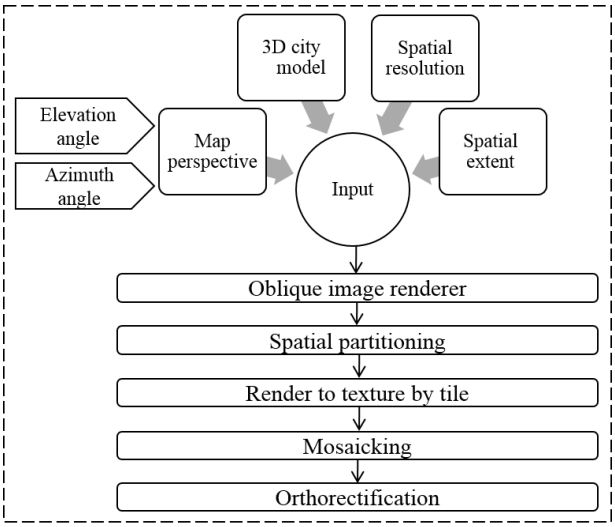


Figure 3. A framework for producing orthorectified 2.5D maps from massive 3D city models.

To orthorectify a multi-perspective set of 2.5D maps, the normal workflow is to identify and collect a set of ground control points (GCPs) in each of them and then apply a polynomial transformation using the set of GCPs to warp the image into alignment with the common reference image. This traditional orthorectifition procedure is not only labor-intensive but also disruptive to a streamlined workflow. Moreover, an efficient implementation of spatial partitioning and out-of-core rendering is also not a trivial task.

2. Methods

2.1. Constructing an Integrated Oblique Image Renderer

In a nutshell, the oblique image renderer is expected to accommodate various 3D model formats, various types of LOD structures, spatial partitioning, and custom camera configuration. To find out which 3D graphics API is most suitable for developing the oblique image renderer, we conducted a comparative review of the mainstream rendering APIs by categories (Table 2):

1. Low-level graphics API

Low-level graphics APIs offer limited support for external data formats. For example, DirecX3D has built-in support only for its native .X format. Additionally, they do not provide advanced capabilities related to out-of-core data management, image data exchange, and scene graph construction.

2. High-level 3D CAD studio

3D CAD studios such as Autodesk 3ds Max and Sketchup are intended for architectural design, model creation and offline rendering of photorealistic images. Autodesk 3ds Max and Sketchup have built-in support for a wide variety of 3D model formats and can be used to create and render individual 3D building models. Massive 3D city models, also known as large-scale 3D city models, are usually present in the form of a hierarchy of progressively simplified individual 3D building models. These 3D CAD studios are not equipped with built-in capability to manage a massive 3D city model with an LOD structure [8].

Table 2. Comparative review of mainstream 3D rendering APIs.

Category	Intended for	Product	Model Format Support
low-level graphics application programming interfaces (API)	providing direct access to GPU rendering pipeline.	OpenGL	no built-in support for any model format.
		DirectX	built-in support only for its native model format.
high-level 3D CAD studio	creating 3D models and serving photo-realistic off-line rendering.	Autodesk 3ds Max	native support for various 3D model formats
		Sketchup	native support for various 3D model formats.
intermediate-level integrated software development kit (SDK)	accelerating development of integrated 3D applications.	OGRE	built-in support only for its native model format.
		OpenSceneGraph	support for various 3D model formats and LOD structures.

3. Intermediate-Level Integrated SDK

OGRE and OpenSceneGraph are open-source 3D graphics libraries developed on top of low-level graphics APIs. They are used mainly by application developers in fields such as visual simulation, computer games, virtual reality, and scientific visualization and modeling. OpenSceneGraph provides

not only native support for a wide variety of 3D model formats, but also a well-organized scene graph that accommodates large-scale 3D city models with an LOD structure. Actually, OAP3D is usually represented in an OpenSceneGraph-based LOD format [8]. Moreover, OpenSceneGraph also affords access to many OpenGL functions such as rendering to texture and camera space transformation, which are crucial to generating oblique image.

Based on the advantages and disadvantages discussed above, OpenSceneGraph was chosen to develop the oblique image renderer, which serves the purpose of transforming a 3D city model into an oblique image for a given perspective. The oblique image renderer can be thought of as the rendering module of a CAD studio with extended support for massive 3D city models. There are typically two types of cameras in computer graphics: a perspective camera and an orthographic camera. A perspective camera acts like a real-world camera or the human visual system. However, an image generated by a perspective camera is subject to perspective distortion, which could adversely affect the geometric accuracy of map tile mosaicking and orthorectification in generating 2.5D maps. To avoid perspective distortion, an orthographic camera is used in the proposed oblique image rendering pipeline. With an orthographic projection, an object always appears the same size no matter how close or far away it is. The orthographic camera (Figures 4 and 5) used for generating an oblique image is adjusted so it covers the entire spatial domain of a 3D city model and is mainly constrained by three factors, i.e., the elevation angle φ, the azimuth angle θ, and the bounding box of the 3D city model.

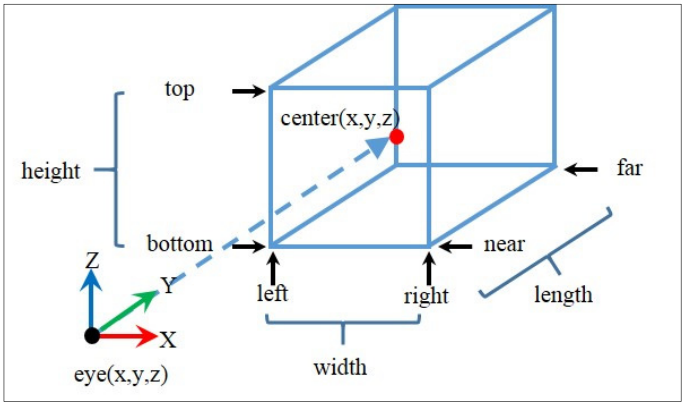


Figure 4. The view and projection transformations of an orthographic camera.

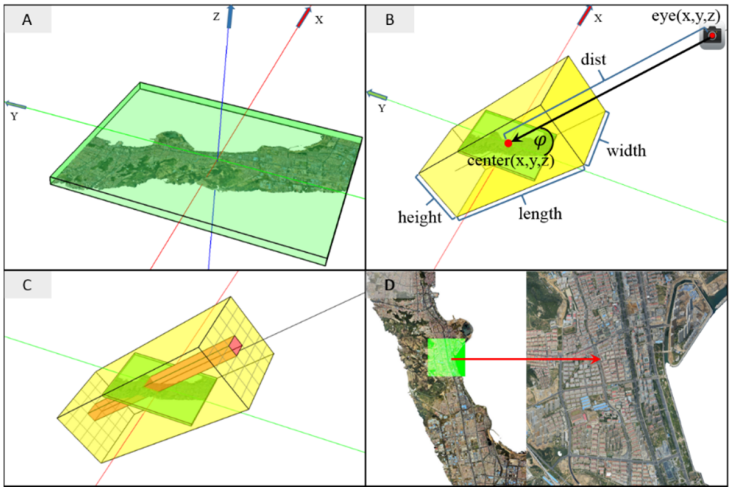


Figure 5. Constructing an oblique camera ($\theta = 0°$, $\varphi = 45°$) for rendering 2.5D maps: (**A**) bounding box of the 3D city models; (**B**) camera configuration; (**C**) spatial partitioning; and (**D**) a map tile rendered from a sub-camera.

The bounding box (Figure 5A) is given by {*xmin*, *ymin*, *zmin*, *xmax*, *ymax*, *zmax*} and has a radius *R*. We define the azimuth angle as the number of degrees clockwise from the negative Y axis. We rotate the 3D city model clockwise around the vertical direction (Z axis) by an amount equal to θ, and then re-compute the world-space bounding box. If θ is 0° in case the camera is oriented in the south–north direction, the rotation will not affect the model orientation and the world-space bounding box. By rotating the 3D city model instead of the camera itself, a fixed camera reference system can be maintained.

The virtual camera is constructed by defining the two transformations [34]:

$$ViewMatrix = \begin{bmatrix} vRight_x & vUp_x & vForward_x & eye_x \\ vRight_y & vUp_y & vForward_y & eye_y \\ vRight_z & vUp_z & vForward_z & eye_z \\ 0 & 0 & 0 & 1 \end{bmatrix}, \quad (1)$$

where *ViewMatrix* defines the camera's local coordinate system, $eye_{(x,y,z)}$ is the position of the camera (Figure 5B), and $vUp_{(x,y,z)}$ is the cross product of $vForward_{(x,y,z)}$ and $vRight_{(x,y,z)}$. The positive X axis, Y axis, and Z axis of the camera coordinate system are defined by $vRight_{(x,y,z)}$, $vUp_{(x,y,z)}$, and $vForward_{(x,y,z)}$ respectively. The four vectors in the *ViewMatrix* are given in Equations (9)–(12).

$$ProjectionMatrix = \begin{bmatrix} \frac{2}{right-left} & 0 & 0 & \frac{right+left}{right-left} \\ 0 & \frac{2}{top-bottom} & 0 & \frac{top+bottom}{top-bottom} \\ 0 & 0 & -\frac{2}{far-near} & -\frac{far+near}{far-near} \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (2)$$

The orthographic projection matrix is defined in Equation (2) by six parameters, which are given by Equations (13)–(18). The width, height, length, and center of the view frustum are given in Equations (4)–(7).

$$\begin{cases} shadowLen = (zmax - zmin) \times \tan\varphi & (3) \\ width = xmax - xmin & (4) \\ height = (ymax - ymin) \times \frac{\sin\varphi}{2} + shadowLen & (5) \\ length \in [R \times 4, \infty] & (6) \\ vCenter_{(x,y,z)} = \left\{ \frac{width}{2}, \frac{(ymax-ymin)}{2} + \frac{shadowLen}{2}, zmin \right\} & (7) \\ dist \in [length, \infty] & (8) \\ vRight_{(x,y,z)} = \{1, 0, 0\} & (9) \\ vForward_{(x,y,z)} = \{0, -\cos\varphi, \sin\varphi\} & (10) \\ vUp_{(x,y,z)} = vForward_{(x,y,z)} \times vRight_{(x,y,z)} & (11) \\ eye_{(x,y,z)} = vCenter_{(x,y,z)} + vForward_{(x,y,z)} \times dist & (12) \\ left = -width/2 & (13) \\ right = width/2 & (14) \\ bottom = -height/2 & (15) \\ top = height/2 & (16) \\ near = dist - length/2 & (17) \\ far = dist + length/2 & (18) \end{cases}$$

In Equations (3)–(7), φ is the elevation angle and $vCenter_{(x,y,z)}$ is translated from the center of the bounding box down to the ground plane and is the reference point that the camera looks at (Figure 5C). We set the ground plane at a minimum height, *zmin*. In Equation (3), *shadowLen* is the maximum possible projected length of a vertical line on the ground plane. We extend the camera frustum height by *shadowLen* and translate $vCenter_{(x,y,z)}$ toward the positive Y axis by *shadowLen*/2 so that tall buildings

(if there are any) on the northern edge remain within the field of view and the projected scene remains centered. The view frustum length in Equation (6) can be any value greater than the diameter of the bounding sphere. In Equation (8), *dist* denotes the distance from $vCenter_{(x,y,z)}$ to the camera; it can be any value greater than the view frustum length.

To overcome computational power limitations, a large scene needs to be spatially segmented into a grid of a given number of rows and columns, with each grid cell linked to a map tile. With the master orthographic camera constructed above, we need only to split the frustum into an array of smaller frustums so that each of these sub-frustums exactly covers the corresponding map tile (Figure 5C). In generating an oblique image of a 3D city model for a set of parameters, the group of sub-frustums is traversed and each sub-frustum camera is activated to render the 3D scene into an OpenGL Render Target Texture (RTT), which is saved as a map tile image immediately after the render loop comes to an end. After all the sub-frustums are traversed, the set of map tiles generated are mosaicked into a seamless raster (Figure 5D).

We exemplify the use of the proposed set of Equations (1)–(18) by parameterizing an oblique image renderer (Table 3). In this example, the building model (Figure 6A) used has a bounding box $\{xmin = -50, ymin = -50, zmin = -50, xmax = 50, ymax = 50, zmax = 50\}$ centered at the origin $\{x = 0, y = 0, z = 0\}$. The oblique camera looks in the south–north direction $\{x = 0, y = 0.707106781, z = -0.707106781\}$ given $\theta = 0^\circ$ and $\varphi = 45^\circ$ (Figure 6A,B). The set of parameters that define the *ViewMatrix* and *ProjectionMatrix* are summarized in Table 3.

Table 3. Parameterization of the view and projection matrix of an oblique image renderer.

Variable	Value
R	86.60253906
shadownLen	100
height	135.3553391
width	100
length	346.4101563
left	−50
right	50
bottom	–67.67766953
top	67.67766953
dist	346.4101563
near	173.2050781
far	519.6152344
vForward	{ 0, –0.707106781, 0.707106781}
vUp	{ 0, 0.707106781, 0.707106781}
vCenter	{ 0, 50, –50}

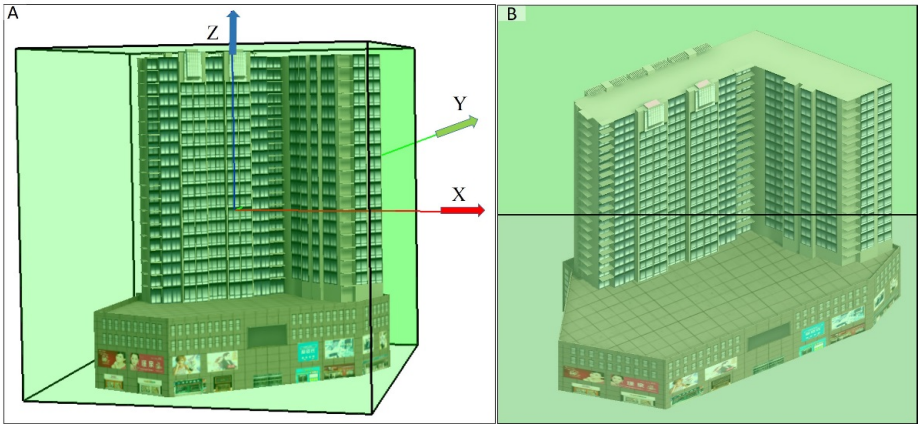


Figure 6. Generating an oblique image from the south–north direction at an elevation angle of 45° using an instantiated oblique image renderer: (**A**) the 3D building model used; (**B**) the oblique image generated.

2.2. Automating GCP Coordinates Retrieval for Orthorectification

In orthorectification, an oblique image is warped so that all ground pixels are mapped back to the orthographic space of the 3D city model. A larger number of GCPs are usually needed in cases with a rugged terrain. To orthorectify a multi-perspective set of 2.5D maps, the coordinates of each GCP need to be identified and collected separately in each map. Automatic retrieval of GCP coordinates can substantially reduce the labor required for the manual identification and collection of GCP coordinates.

We present a computer graphics-based method for automatically retrieving the GCPs' coordinates in an oblique image. Before this method can be applied to an oblique image, a set of GCPs with orthographic coordinates need to be collected from the 3D city model in a 3D GIS or from an orthoimage in a 2D GIS (Figure 7A). In collecting the GCPs, one must ensure that every GCP is sampled from the exposed ground (Figure 7A) instead of from other aboveground objects, e.g., buildings, trees, and vehicles. A four-byte RTT the same size as a map tile in the grid is allocated for rendering to texture. With the proposed method, the coordinates of each GCP in an oblique image can be correctly retrieved as follows (Figure 7):

1. Locate the row and column number of the map tile that contains the GCP by the orthographic coordinates of this GCP.
2. Create an OpenGL point primitive using the GCP coordinates and then render the point together with the 3D city model using the associated orthographic camera. In the GPU shading pipeline, the GCP primitive is shaded in red and the 3D city model in pure black with all textures and materials disabled (Figure 7B). In Figure 6B, the background pixels associated with the 3D city model are not discarded to show how the GCPs are displaced in an oblique view against the orthographic view.
3. Traverse the pixels in the RTT after the render loop is completed. The red pixels are retained (Figure 7C) while black ones are discarded. The center of the cluster of red pixels is assumed to be the coordinates of this GCP in the oblique space.

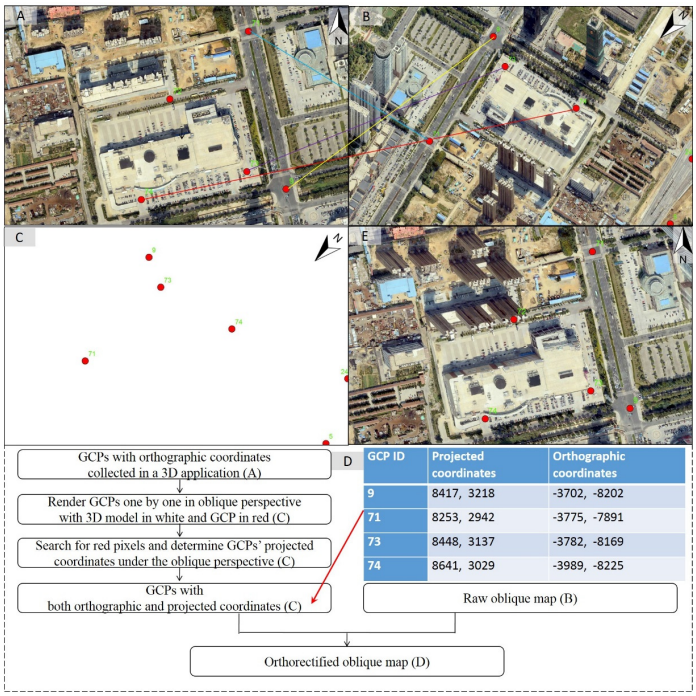


Figure 7. Illustration of the automatic GCP coordinates retrieval approach: (**A**) GCPs in an orthographic view; (**B**) GCPs in an oblique view; (**C**) GCPs rendered in an oblique view without background for identifying coordinates; (**D**) GCP table obtained for use in orthorectification; (**E**) oblique image orthorectified using the set of GCPs.

The retrieved GCPs are stored in a table (Figure 7D) for use in the final image warping (Figure 7E). With these GCPs, a polynomial transformation can be applied to warp the oblique image into alignment with the orthographic reference system. ArcMap normally comes with an image warping tool that can make use of externally created GCPs in the polynomial transformation. The open-source raster library GDAL also provides a free image warping tool, the "gdalwarp utility", which is used in the proposed streamlined framework because open-source programs can be more conveniently integrated into an automated workflow.

3. Application

Two OAP3D datasets were acquired using a combination of oblique airborne photogrammetry and photo-based 3D reconstruction (Figure 8). 2.5D maps are generated from the two 3D city models using the proposed 2.5D cartography framework. A comparison of the 2.5D maps and the 3D city model is made to show the potential of 2.5D maps for improving user experience in exploring 3D city models in a web environment. We assess the accuracy of building height measurement on the 2.5D maps against the 3D city model and then lay out the general framework for integrating 2.5D maps into web GIS. Two simple examples are presented to show how 2.5D maps can be integrated into real-world applications.

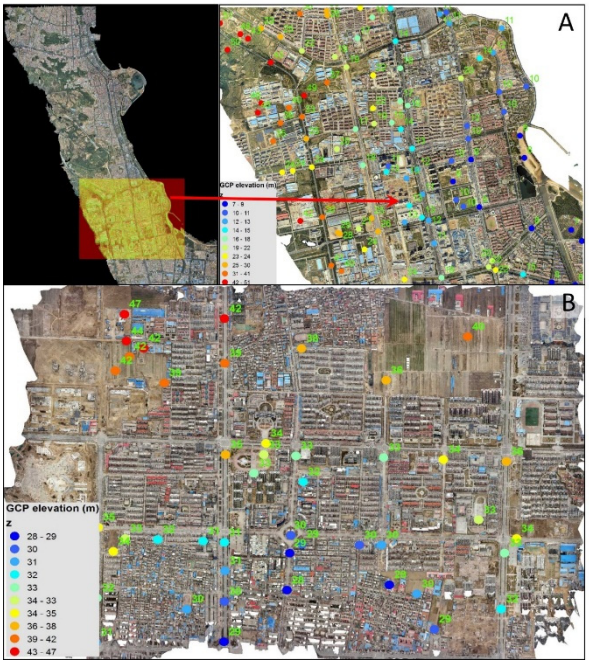


Figure 8. The distribution of GCPs in two OAP3D datasets used for generating 2.5D maps and assessing quality: (**A**) Dataset 1 in a rugged terrain; (**B**) Dataset 2 in a gentle terrain.

3.1. Generating 2.5D Maps from 3D City Models

2.5D maps at a spatial resolution of 25 cm were generated for eight perspectives, defined by the combinations of a 45° elevation angle and the eight azimuth directions, including north, south, east, west, northeast (NE), southeast (SE), southwest (SW), and northwest (NW). Collection of GCPs was conducted in a 3D interactive environment, in which the orthographic coordinates of a GCP were registered at the intersection point of a mouse click position and the ground plane. Dataset 2 (Figure 8B) covers an area of flat topography. By contrast, Dataset 1 is situated in rugged terrain (Figure 8A). In collecting the GCPs, areas surrounding buildings were not considered since they may be obstructed by the buildings when viewed from an oblique perspective. Consequently, a majority of the GCPs were placed on arterial road segments, which were clear of building obstructions.

In orthorectifying an oblique image, the oblique-space coordinates of each GCP were retrieved using the automatic GCP coordinates retrieval technique. Orthorectification with a third-order transformation was used to warp the oblique images into the orthographic reference system. To show the performance of the automatic GCP coordinates retrieval algorithm, all GCPs were used in the orthorectification process without being adjusted or screened.

As expected, the 2.5D maps created from Dataset 1 (Figure 8A) have a mean RMSD of 4.5 m, which is significantly larger than the RMSD (0.8 m) with Dataset 2 (Figure 8B). This large difference in the geometric accuracy of the orthorectified 2.5D maps was likely caused by terrain relief. We conducted a visual inspection to assess the quality of the orthorectified 2.5D maps. An evenly divided grid was draped on the orthographic map and the 2.5D maps from different perspectives for comparison of reference locations. We utilized the locations of some visually prominent ground-level features to assess the relative displacement between an orthographic map and an oblique map. At the given map scale, noticeable displacement (Figure 9) can be observed in areas of rugged relief (Figure 8A), but little displacement (Figure 10) appear in areas of gentle relief (Figure 8B).

Figure 9. 2.5D maps of different perspectives created from Dataset 1 for assessing geometric accuracy (follow the yellow arrows to observe displacements): (**A**) orthographic view; (**B**) $\theta = 0°$, $\varphi = 45°$; (**C**) $\theta = 45°$, $\varphi = 45°$; (**D**) $\theta = 135°$, $\varphi = 45°$.

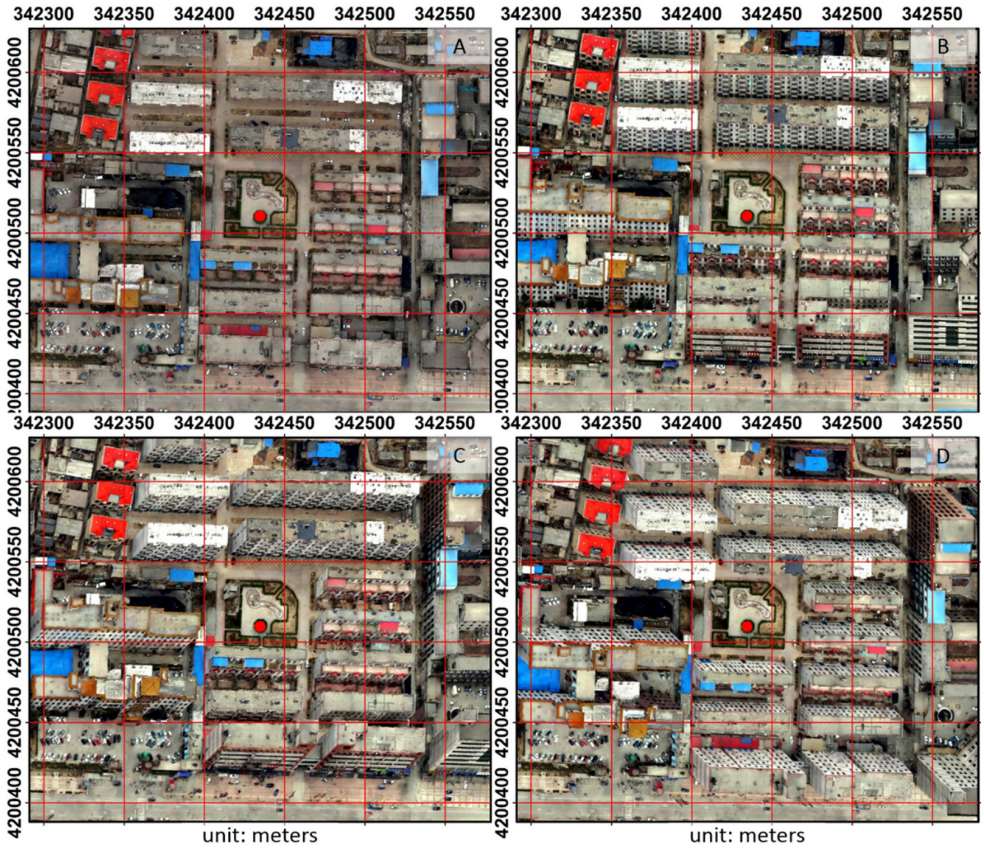


Figure 10. 2.5D maps of different perspectives created from Dataset 2 for assessing geometric accuracy: (**A**) orthographic view; (**B**) $\theta = 0°$, $\varphi = 45°$; (**C**) $\theta = 45°$, $\varphi = 45°$; (**D**) $\theta = 135°$, $\varphi = 45°$.

3.2. Comparison of 2.5D and 3D Representations in Web-Based Visualization of 3D City Models

Data transfer can become a challenging bottleneck for a web GIS if a large volume of data is involved while only limited Internet bandwidth is available. A compact data representation is conducive to a web GIS that focuses on interactive visualization and analysis. Datasets 1 and 2 in their original form take up 81.5 GB and 4.8 GB of disk space, respectively. By contrast, the eight-perspective sets of 2.5D maps in GeoTiff format with LZW compression were reduced to 1.6 GB and 0.14 GB, showing a compression ratio of 51:1 and 34:1, respectively.

To find out how 2.5D maps can improve the user experience in web-based visualization of 3D city models, Dataset 2 and the associated 2.5D maps were created on a local area network with a maximum transfer speed of 10 MB/s. This dataset covers a 3195 m by 2292 m area, which was divided into a grid of 50 m by 50 m sub-regions with 45 rows and 63 columns. On the client side, to simulate a user interactively exploring a 3D city model in full detail, a robot user was programed to zoom into the 2835 sub-regions one by one and would remain focused on a sub-region until the data download was completed and the area appeared in full detail. A 50 m by 50 m sub-region was used because the 3D city model has a spatial resolution higher than 25 cm and can show its content in full detail only when zoomed in on an area of approximately 50 m by 50 m or smaller. The robot user program was made to navigate through the 2835 sub-regions separately for the 3D and 2.5D visualization. Under the 3D visualization mode, a total of 2253 s were spent exploring the 2835 sub-regions with data retrieval and 3D rendering consuming 1648 and 605 s, respectively. Under the 2.5D visualization mode with a constant oblique perspective, a mere total of 20 s on average were spent, with 5 s attributed to data retrieval and 15 s attributed to image rendering. This shows that 2.5D maps can potentially improve the user experience in web-based visualization of 3D city models by shortening the time required for data retrieval and rendering.

3.3. Geometric Measurement on 2.5D Maps and Accuracy Assessment

In a 2D GIS, a 2.5D map can be utilized to measure the geometric features of buildings. In theory, the actual length of a linear feature that is perpendicular to the ground plane can be found by dividing the projected length by the tangent of the camera elevation angle. When the camera elevation angle is 45°, the projected length is assumed to be equal to the actual length. Therefore, direct measurement can be conducted on a 2.5D map with an elevation angle of 45°. For a 2.5D map with a larger or smaller elevation angle, measurement is still applicable with simple trigonometric calculations. To assess the measurement accuracy of geometric features on 2.5D maps, we picked out 20 reference buildings from Dataset 2. We measured the height of each of the 20 buildings in a real 3D GIS and then on the 2.5D maps for comparison (Figure 11). As shown in Table 4, the two sets of building heights agree well with each other with a root mean square deviation (RMSD) of 0.701 m. Besides height measurements, building façade area can also be approximated (Figure 12).

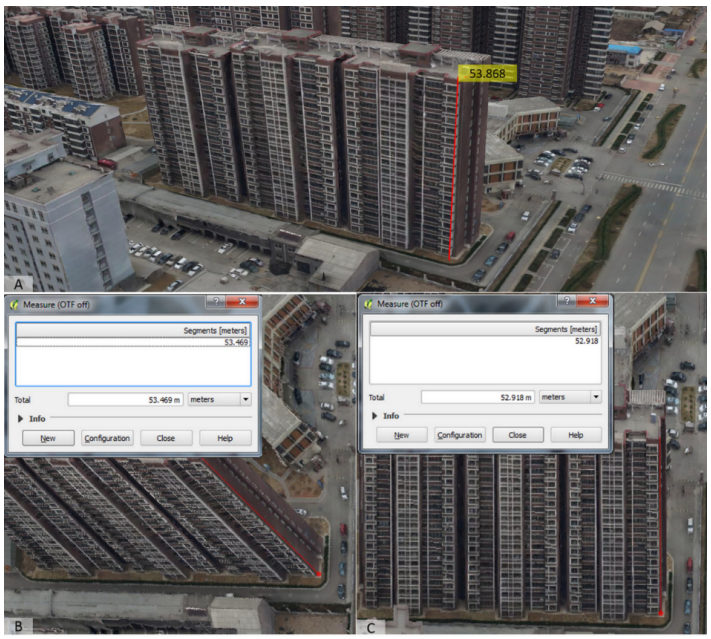


Figure 11. Measuring building height (**A**) in a real 3D environment; (**B**) on a 2.5D map with $\theta = 315°$, $\varphi = 45°$; (**C**) on a 2.5D map with $\theta = 0°$, $\varphi = 45°$.

Table 4. Comparison of the building heights measured on a 2.5D map and in a 3D GIS.

Building ID	3D Measurement (m)	2D Measurement (m)	Difference (m)
1	12.507777	13.125066	0.617289
2	13.713325	13.732419	0.019094
3	14.129196	14.155275	0.026079
4	16.866035	16.933496	0.067461
5	17.225675	17.833031	0.607356
6	18.953028	18.25633	–0.696698
7	18.758588	18.653542	–0.105046
8	19.039177	19.47366	0.434483
9	20.484453	19.157041	–1.327412
10	20.829912	20.002697	–0.827215
11	20.874955	21.008558	0.133603
12	20.955293	20.958611	0.003318
13	22.353933	22.861286	0.507353
14	22.754253	22.423482	–0.330771
15	22.520097	22.066294	–0.453803
16	23.235158	24.293483	1.058325
17	33.576494	31.089237	–2.487257
18	37.405119	36.844787	–0.560332
19	55.114582	54.690003	–0.424579
20	56.291389	56.197712	–0.093677

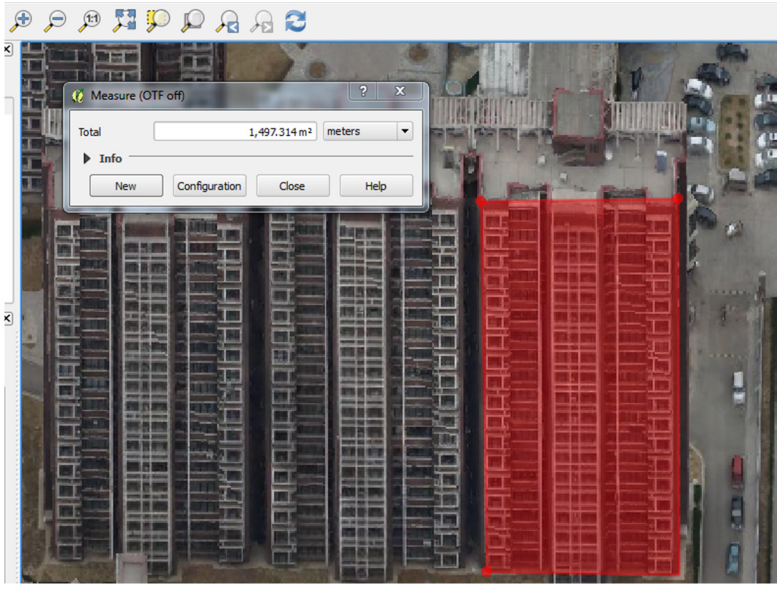


Figure 12. Measuring building façade area on a 2.5D map in a 2D GIS.

3.4. Workflow for Integrating 2.5D Maps into Web GIS

A multi-perspective 2.5D layer set is essentially a raster dataset, which can be published through standard map services such as WMS and TMS (Figure 13). Most open-source map servers, including Geoserver, Mapserver, and QGIS Server, support WMS service and accommodate raster datasets that come in a standard geoferenced raster format, such as GeoTIFF. This means a geoferenced 2.5D map in a standard format can be hosted as a WMS layer on these servers and thus multi-perspective 2.5D maps can be published as a set of WMS layers. Another option is to publish 2.5D maps through TMS service, which generates maps as a pyramid of tiles at multiple zoom levels. Open-source tools, such as GDAL2Tiles, can generate TMS tiles from a georeferenced 2.5D map. Viewing 2.5D maps in browsers on various devices and platforms has become an easy task with the availability of open-source mapping libraries such as OpenLayers, which can display map tiles, vector data, and markers loaded from a wide variety of sources.

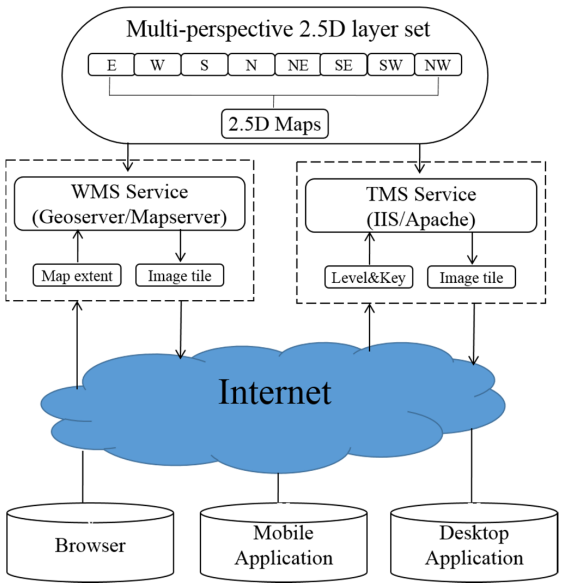


Figure 13. Integrating 2.5D maps into web GIS via map services.

3.5. Integrating Orthorectified 2.5D Images into a Street Map for Campus Navigation

Traditionally, 2.5D campus maps are usually rendered from 3D city models created in a CAD environment. The multi-perspective 2.5D maps generated using the proposed method can serve as an alternative data source for web-based 2.5D campus mapping (Figure 14). In contrast to manually created CAD models, OAP3D can be easily acquired and rapidly updated over a geographic domain much larger than a university campus. Thus, OAP3D-based 2.5D maps cannot only reduce the cost of 2.5D campus mapping, but also extend the use of 2.5D campus mapping to a wider scope of geographic environments, including industrial parks, high schools, hospitals, and amusement parks.

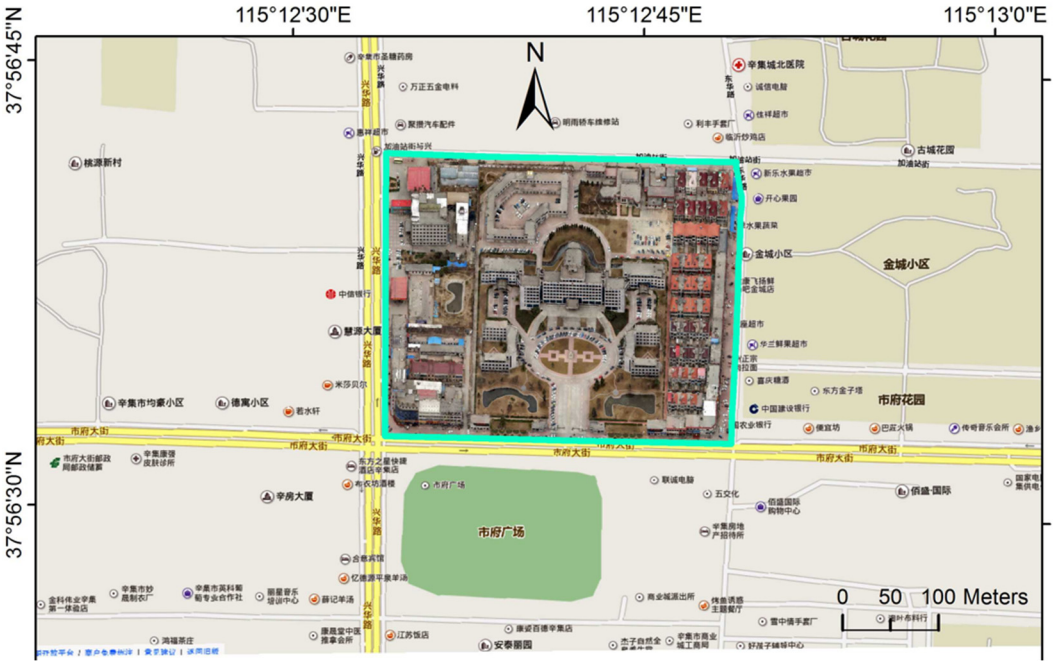
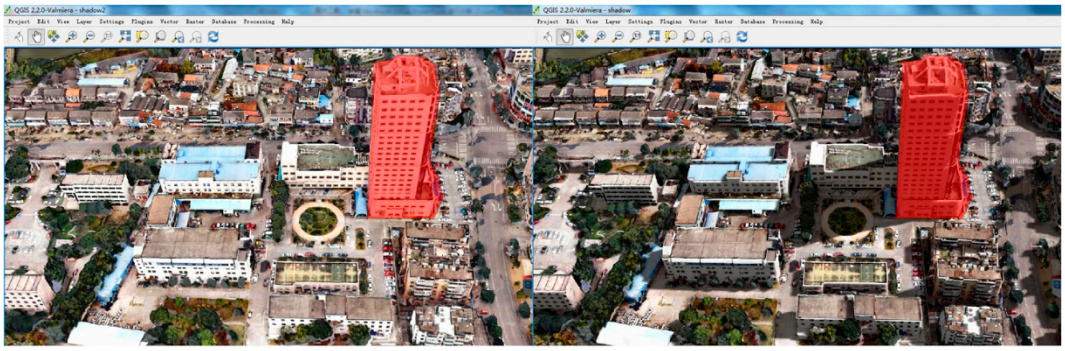


Figure 14. Integrating 2.5D images into a street map for campus navigation.

3.6. The Fusion of Scientific Data and Art in 2.5D Cartography

Cartography is the discipline that combines science, technology, and art to prepare and study all types of maps and charts [35]. Modern cartography relies on specialized data models in GIS to store, manage, symbolize, and render digitized geographic information [36,37]. Since the first operational GIS was introduced in 1960, cartography had been gradually integrated into GIS as an essential component that provides capabilities for the mapping of cartographic information [38]. Goodchild [39] commented that "Cartography also deals with geographic information, but in a manner that combines the scientific with the artistic". Krygier [35] contended that "art" and "science" serve a functionally similar role in cartography.

Pixel art-based cartography (Figure 2) is arguably more artistic than scientific, as the data content is not physically related to the real world. In contrast to the pixel art-based cartography (Figure 2), a 2.5D map from an OAP3D is a photogrammetric representation of an urban environment and is reasonably more scientific than artistic. Following the cartographic philosophy of "integrating art and science", we can improve the aesthetic quality and extend the use of OAP3D-based 2.5D maps by (1) employing computer graphics techniques to produce desired visual effects [40,41]; even simple shadowing effects can enhance the aesthetic quality of 2.5D maps (Figure 15); And (2) incorporating externally generated content, e.g., architectural and urban design [8], to achieve a wider scope of application.



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

Figure 15. Incorporating computer-generated virtual effect into a 2.5D map (**a**) without shadow effect; (**b**) with shadow effect.

4. Conclusions

2.5D mapping is a convenient and efficient approach to delivering massive 3D city models to 2D web GIS. With 2.5D cartography, existing massive 3D city models can be used by a wider audience and in a wider variety of contexts. A systematic 2.5D cartography framework was presented in this study. We have shown how the technical challenges, including spatial partitioning, camera construction, and orthorectification, can be effectively resolved by using a combination of GIS and computer graphics techniques. We have laid out a streamlined workflow for producing high-quality 2.5D maps using massive 3D city models. The geometric accuracy of the orthorectification under different topographic conditions has been assessed. In gentle terrain relief, we achieved an RMSD of 0.8 m, which suggests that the presented automatic GCP coordinates retrieval method is effective in orthorectifying multi-perspective 2.5D maps. Although a mean RMSD of 4.5 m was found in rugged topography, the orthorectification quality can be improved if more GCPs are used.

Effective utilization of 3D city models in 2D web GIS is mainly reflected in two functional aspects:

1. Interactive analysis. Geometric measurement is typical of interactive analysis in 3D city models. We have shown by example that the geometric measurement of buildings can be effectively conducted on 2.5D maps. The accuracy assessment revealed that measurement of building height on 2.5D maps is subject to minor errors. Although the RMSD is as small as 0.701 m, it must be considered in engineering activities such as cadastral survey. The uncertainty in geometric measurement on 2.5D maps may be related to the inaccurate positioning of a point, inaccurate alignment of lines, or insufficient map resolution.
2. Interactive visualization. We conclude that 2.5D maps are a compact data representation optimized for web data streaming and mapping. Our case study showed that a compression ratio of 51:1 was achievable by transforming an OAP3D of 81.5 GB into an eight-perspective set of 2.5D maps of 1.6 GB. Efficient streaming of high-resolution 2.5D maps to a client can ensure a high-quality visualization experience.

Acknowledgments: This research was supported and funded by the Foundation for Young Scientists of the State Key Laboratory of Remote Sensing Science (15RC-08), the Key Knowledge Innovative Project of the Chinese Academy of Sciences (KZCX2 EW 318), the National Key Technology R&D Program of China (2014ZX10003002), the National Natural Science Foundation of China (41371387), and the Science and Technology Program of Guangzhou (201604020117) (high-resolution estimation method of UAU-based urban building carbon emissions–key special subject in cooperation innovative combining learning with research and production of Guangzhou Municipality). We thank the three anonymous reviewers for their constructive comments and Maya Hutchins for her proofreading.

Author Contributions: Jianming Liang and Jianhua Gong conceived and designed the method; Jianming Liang implemented the method; all the authors reviewed and edited the manuscript.

Conflicts of Interest: The authors have no conflicts of interest to declare.

References

1. Bruse, M.; Fleer, H. Simulating surface–plant–air interactions inside urban environments with a three dimensional numerical model. *Environ. Model. Softw.* **1998**, *13*, 373–384. [CrossRef]
2. Huang, B.; Jiang, B.; Li, H. An integration of GIS, virtual reality and the Internet for visualization, analysis and exploration of spatial data. *Int. J. Geogr. Inf. Sci.* **2001**, *15*, 439–456. [CrossRef]
3. Lin, H.; Chen, M.; Lu, G. Virtual geographic environment: A workspace for computer-aided geographic experiments. *Ann. Assoc. Am. Geogr.* **2013**, *103*, 465–482. [CrossRef]
4. Li, W.; Gong, J.; Yu, P.; Duan, Q.; Zou, Y. A stream-based Parasitic Model for implementing Mobile Digital Earth. *Int. J. Dig. Earth* **2014**, *7*, 38–52. [CrossRef]
5. Ledoux, H.; Meijers, M. Topologically consistent 3D city models obtained by extrusion. *Int. J. Geogr. Inf. Sci.* **2011**, *25*, 557–574. [CrossRef]
6. Cheng, L.; Gong, J.; Li, M.; Liu, Y. 3D building model reconstruction from multi-view aerial imagery and lidar data. *Photogramm. Eng. Remote Sens.* **2011**, *77*, 125–139. [CrossRef]
7. Remondino, F.; Barazzetti, L.; Nex, F.; Scaioni, M.; Sarazzi, D. UAV photogrammetry for mapping and 3d modeling–current status and future perspectives. *Int. Arch. Photogramm. Remote Sens. Spat. Inf. Sci.* **2011**, *38*, 22–37. [CrossRef]
8. Liang, J.; Shen, S.; Gong, J.; Liu, J.; Zhang, J. Embedding user-generated content into oblique airborne photogrammetry-based 3D city model. *Int. J. Geogr. Inf. Sci.* **2016**. [CrossRef]
9. Ranzinger, M.; Gleixner, G. GIS datasets for 3D urban planning. *Comput. Environ. Urban Syst.* **1997**, *21*, 159–173. [CrossRef]
10. Zhang, X.; Zhu, Q.; Wang, J. 3D city models based spatial analysis to urban design. *Geogr. Inf. Sci.* **2004**, *10*, 82–86. [CrossRef]
11. Biljecki, F.; Stoter, J.; Ledoux, H.; Zlatanova, S.; Çöltekin, A. Applications of 3D city models: State of the art review. *ISPRS Int. J. Geo-Inf.* **2015**, *4*, 2842–2889. [CrossRef]
12. Biljecki, F.; Heuvelink, G.B.; Ledoux, H.; Stoter, J. Propagation of positional error in 3D GIS: Estimation of the solar irradiation of building roofs. *Int. J. Geogr. Inf. Sci.* **2015**, *29*, 2269–2294. [CrossRef]
13. Liang, J.; Gong, J.; Li, W.; Ibrahim, A.N. A visualization-oriented 3D method for efficient computation of urban solar radiation based on 3D–2D surface mapping. *Int. J. Geogr. Inf. Sci.* **2014**, *28*, 780–798. [CrossRef]
14. Liang, J.; Gong, J.; Zhou, J.; Ibrahim, A.N.; Li, M. An open-source 3D solar radiation model integrated with a 3D Geographic Information System. *Environ. Model. Softw.* **2015**, *64*, 94–101. [CrossRef]
15. Lukač, N.; Žalik, B. GPU-based roofs' solar potential estimation using LiDAR data. *Comput. Geosci.* **2013**, *52*, 34–41. [CrossRef]
16. Carrión, D.; Lorenz, A.; Kolbe, T.H. Estimation of the energetic rehabilitation state of buildings for the city of Berlin using a 3D city model represented in CityGML. *Int. Arch. Photogramm. Remote Sens. Spat. Inf. Sci.* **2010**, *38*, 31–35.
17. Saran, S.; Wate, P.; Srivastav, S.K.; Krishna Murthy, Y.V.N. CityGML at semantic level for urban energy conservation strategies. *Ann. GIS.* **2015**, *21*, 27–41. [CrossRef]
18. Boeters, R.; Arroyo Ohori, K.; Biljecki, F.; Zlatanova, S. Automatically enhancing CityGML LOD2 models with a corresponding indoor geometry. *Int. J. Geogr. Inf. Sci.* **2015**, *29*, 2248–2268. [CrossRef]
19. Shiravi, S.; Zhong, M.; Beykaei, S.A.; Hunt, J.D.; Abraham, J.E. An assessment of the utility of LiDAR data in extracting base-year floorspace and a comparison with the census-based approach. *Environ. Plan. B Plan. Des.* **2015**, *42*, 708–729.
20. Henn, A.; Römer, C.; Gröger, G.; Plümer, L. Automatic classification of building types in 3D city models. *GeoInformatica* **2012**, *16*, 281–306. [CrossRef]
21. Royan, J.; Gioia, P.; Cavagna, R.; Bouville, C. Network-based visualization of 3d landscapes and city models. *IEEE Comput. Graph. Appl.* **2007**, *27*, 70–79. [CrossRef] [PubMed]
22. Yang, P.P.J.; Putra, S.Y.; Li, W. Viewsphere: A GIS-based 3D visibility analysis for urban design evaluation. *Environ. Plan. B Plan. Des.* **2007**, *34*, 971–992. [CrossRef]
23. Hofierka, J.; Zlocha, M. A new 3-D solar radiation model for 3-D city models. *Trans. GIS.* **2012**, *16*, 681–690. [CrossRef]
24. Guo, R.; Li, L.; Ying, S.; Luo, P.; He, B.; Jiang, R. Developing a 3D cadastre for the administration of urban land use: A case study of Shenzhen, China. *Comput. Environ. Urban Syst.* **2013**, *40*, 46–55. [CrossRef]

25. Duan, Q.; Gong, J.; Li, W.; Shen, S.; Li, R. Improved Cubemap model for 3D navigation in geo-virtual reality. *Int. J. Dig. Earth.* **2015**, *8*, 877–900. [CrossRef]
26. Wu, H.; He, Z.; Gong, J. A virtual globe-based 3D visualization and interactive framework for public participation in urban planning processes. *Comput. Environ. Urban Syst.* **2010**, *34*, 291–298. [CrossRef]
27. Hijazi, I.H.; Ehlers, M.; Zlatanova, S. NIBU: A new approach to representing and analysing interior utility networks within 3D geo-information systems. *Int. J. Dig. Earth.* **2012**, *5*, 22–42. [CrossRef]
28. Kwan, M.P.; Lee, J. Emergency response after 9/11: The potential of real-time 3D GIS for quick emergency response in micro-spatial environments. *Comput. Environ. Urban Syst.* **2005**, *29*, 93–113. [CrossRef]
29. Lu, Z.; Im, J.; Quackenbush, L. A volumetric approach to population estimation using LiDAR remote sensing. *Photogramm. Eng. Remote Sens.* **2011**, *77*, 1145–1156. [CrossRef]
30. Hildebrandt, D.; Timm, R. An assisting, constrained 3D navigation technique for multiscale virtual 3D city models. *GeoInformatica* **2014**, *18*, 537–567. [CrossRef]
31. Amirebrahimi, S.; Rajabifard, A.; Mendis, P.; Ngo, T. A framework for a microscale flood damage assessment and visualization for a building using BIM–GIS integration. *Int. J. Dig. Earth.* **2016**, *9*, 363–386. [CrossRef]
32. Liu, J.; Gong, J.H.; Liang, J.M.; Li, Y.; Kang, L.C.; Song, L.L.; Shi, S.X. A quantitative method for storm surge vulnerability assessment—A case study of Weihai City. *Int. J. Dig. Earth* **2016**. [CrossRef]
33. Qin, R. Change detection on LOD 2 building models with very high resolution spaceborne stereo imagery. *ISPRS J. Photogramm. Remote Sens.* **2014**, *96*, 179–192. [CrossRef]
34. Shirley, P.; Ashikhmin, M.; Marschner, S. *Fundamentals of Computer Graphics*; CRC Press: Boca Raton, FL, USA, 2009.
35. Krygier, J.B. Cartography as an art and a science? *Cartogr. J.* **1995**, *32*, 3–10. [CrossRef]
36. Peuquet, D.J. An examination of techniques for reformatting digital cartographic data/Part 1: The raster-to-vector process. *Cartogr. Int. J. Geogr. Inf. Geovis.* **1981**, *18*, 34–48.
37. Peuquet, D.J. An examination of techniques for reformatting digital cartographic data/Part 2: The vector-to-raster process. *Cartogr. Int. J. Geogr. Inf. Geovis.* **1981**, *18*, 21–33.
38. Berry, J.K. Fundamental operations in computer-assisted map analysis. *Int. J. Geogr. Inf. Syst.* **1987**, *1*, 119–136. [CrossRef]
39. Goodchild, M. Twenty years of progress: GIScience in 2010. *J. Spat. Inf. Sci.* **2015**, *1*, 3–20. [CrossRef]
40. Liang, J.; Gong, J.; Li, Y. Realistic rendering for physically based shallow water simulation in Virtual Geographic Environments (VGEs). *Ann. GIS* **2015**, *21*, 301–312. [CrossRef]
41. Liu, P.; Gong, J.; Yu, M. Graphics processing unit-based dynamic volume rendering for typhoons on a virtual globe. *Int. J. Dig. Earth.* **2015**, *8*, 431–450. [CrossRef]