Web-based presentation of multisensoral 3D data on the example of the Church of St. Augustine in Warsaw

Jakub GRYZIO, Paweł CZERNIC, Łukasz WILK, Poland

Keywords: Geoinformation/GI, Laser scanning, Photogrammetry, Young surveyor, 3D visualization, web app

SUMMARY

The paper will aim to discuss the course of the project, including the measurement techniques, algorithms, and methodology used. As part of the research, measurements of the Church of St. Augustine in Warsaw were made. The three-dimensional point cloud was obtained using terrestrial laser scanning and short-range photogrammetry.
The first issue discussed will be the ability to combine data obtained using the methods mentioned above to create one coherent point cloud of the church. In the next part of the presentation, open-source 3D visualization web libraries will be considered. Visualization using the web browser is an exquisite way to present geospatial data of cultural heritage for the standard user. This processed 3D model can be easily adapted to modern video game engines or create virtual reality.
At the end of the presentation, the research results will be discussed: the accuracy of the obtained point cloud, the quality of generated mesh model, the effectiveness of 3D data web visualization, and the conclusions drawn during the project.

STRESZCZENIE

Referat będzie miał na celu omówienie przebiegu projektu, w tym zastosowanych technik pomiarowych, algorytmów i metodologii. W ramach badań wykonano pomiary kościoła św. Augustyna w Warszawie. Trójwymiarowa chmura punktów została pozyskana za pomocą naziemnego skaningu laserowego oraz fotogrametrii bliskiego zasięgu.
Pierwszym omawianym zagadnieniem będzie możliwość łączenia danych uzyskanych za pomocą wymienionych metod w celu stworzenia jednej, spójnej chmury punktów kościoła. W kolejnej części prezentacji zostaną omówione biblioteki internetowe do wizualizacji 3D typu open-source. Wizualizacja za pomocą przeglądarki internetowej jest doskonałym sposobem prezentacji danych geoprzestrzennych dotyczących dziedzictwa kulturowego dla standardowego użytkownika. Tak przetworzony model 3D można łatwo zaadaptować do nowoczesnych silników gier wideo lub stworzyć wirtualną rzeczywistość.
Na zakończenie prezentacji omówione zostaną wyniki badań: dokładność uzyskanej chmury punktów, jakość wygenerowanego modelu siatkowego, efektywność wizualizacji internetowej danych 3D oraz wnioski wyciągnięte w trakcie realizacji projektu.

Web-based presentation of multisensoral 3D data on the example of the Church of St. Augustine in Warsaw

Jakub GRYZIO, Paweł CZERNIC, Łukasz WILK, Poland

1. INTRODUCTION

The aim of the project was to test the reproducibility of simple buildings and buildings decorated with numerous architectural details. Therefore, a search was conducted for buildings in Warsaw that would meet these requirements. The second important criterion also guided the search - the object had to be located outside the zone closed for Unmanned Aerial Vehicle (UAV) air raids. In the search, several candidates were selected, but the most exciting position turned out to be the Church of Saint Augustine, located in the Warsaw district of Wola. The construction of this historic church began in 1891 and was completed in 1896. A fascinating fact is that the building was located within the Warsaw Ghetto for the Jewish population during the Second World War, whose grounds were almost destroyed during the war and the Warsaw Uprising. The church itself remained one of the only buildings preserved after the war. The shape of the building is based on a rectangle. The central brick nave is almost 70m long, while together with the side aisles, the whole church is about 40m wide. A gable cooper roof tops the building over its entire length, the dominant feature of which is the soaring 70m tower topped by a 5m cross. There is also a lower tower about 29m high at the back of the church. The entire building is surrounded by a courtyard about 20m wide with deciduous trees, while residential buildings surround the church grounds.

2. DATA ACQUISITION

2.1. TLS data acquisition

The fieldwork for TLS data acquisition was done on March 17, 2022, by students belonging to the Koło Naukowe Geodezji i Kartfografii. The terrestrial laser scanning was performed with a Leica RTC360 scanner, and its purpose was to record the lower parts of the church that would be hard accessible to UAVs due to the building's surroundings. Twenty-six setups distributed around the object and two placed at the main entrance during the fieldwork were recorded (Figure 1).

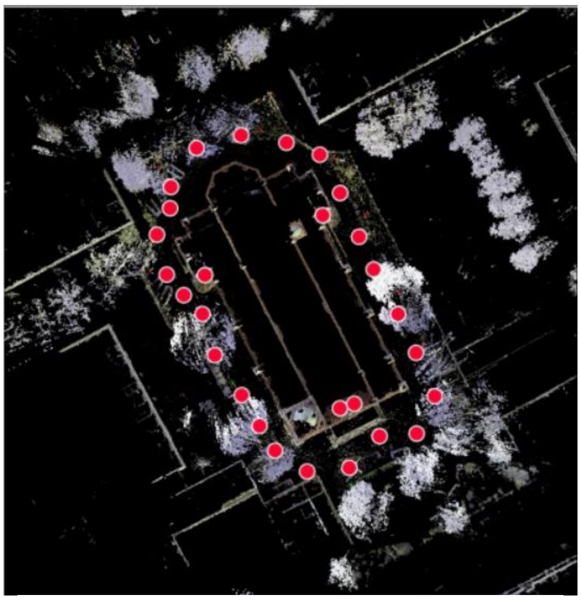


Figure 1: TLS setups map

Leica Cyclone Field 360 mobile software was used to operate the scanner, allowing scans to be pre-registered during fieldwork. All scans were taken at medium resolution (6mm at 10m) and HDR images. During the entire survey, a VIS system was used to allow for real-time viewing of the scans taken and the initial orientation between each other (Biason et al., 2019).
Two methods were used to signal the measured points: 4.5-inch printed signal targets on A4 sheets taped to the object's walls and four 4.5-inch Leica GZT21 targets repositioned as the fieldwork progressed. A Leica TS16 total station characterized by an angle measurement error of 1' and a mirrorless measurement error of 2+2ppm was used to determine the position of all targets. The position and orientation of the total station were determined using the free-station method on points determined using GNSS RTN technology with the Leica GS07 receiver without marking these points. This workflow is an efficient way to establish a ground reference in highly urbanized areas.
The average error of points measured in the RTN technique was 0.012m horizontally and 0.015m vertically. These errors translated into the RMS of the stations at the level of 0.010m horizontally and 0.009m vertically. The final RMS of the 3D targets was 0.015m.
All measurements were done in the official polish projected coordinate system – PL-2000 zone 7 (EPSG:2178) and PL-EVRF2007-NH in case of elevation.

2.2. UAV data acquisition

The photogrammetric raid over the church grounds was done with a DJI Phantom 3 Professional drone on January 16, 2020. Images were captured using a built-in 12 Mpx, 4mm focal length camera mounted on a gyro-stabilized hover. In order to obtain a consistent resulting 3D model, it was decided to take both nadir and oblique images. All images were acquired from approximately 78m, resulting in a 3.34 cm/pix GSD. The whole object was mapped on 519 images (123 nadirs + 396 obliques).

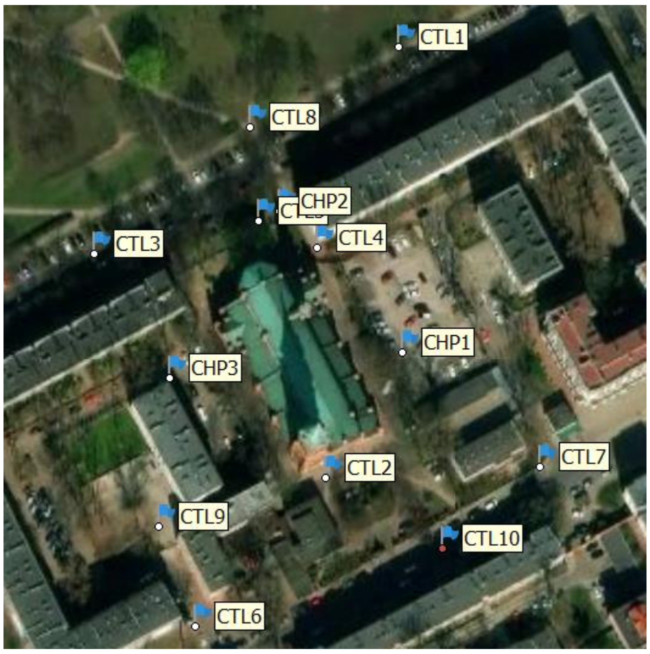


Figure 2: GCP distribution (CTL - control points, CHP - check points)

In order to accurately position the final 3D model of the object in the external coordinate system, it was also necessary to plan and measure the relevant ground control points (GCP). 13 natural (non-signaled, e.g., centers of sewer hatches) points were used to orient the photographs (Figure 2), divided into ten control points and three checkpoints. Field measurements of GCPs previously identified on the images were made later (simultaneously with the terrestrial measurements) using the same instruments and coordinate systems as those used for the TLS datum measurements.

3. DATA PROCESSING

3.1. TLS data processing

Registration of the acquired scans was performed using Leica Cyclone Register 360 software. The first step was to preview the TLS data and a specially prepared text file containing the coordinates of the reference points data. The text file must contain comma-separated-values, including point name, X coordinate, Y coordinate, and elevation.
Thanks to the pre-registration performed during the fieldwork, the preview data were immediately pre-oriented and linked, significantly reducing the processing time. Before importing complete TLS data, the user can choose to import parameters like an automatic cloud to cloud (C2C) registration, Smart Align, or automatic target detection. In our case, all of the listed options were used, except the pre-registration option.
After importing the complete data, the user could review the data and perform optimizations, including georeferencing. For the project in this paper, several manual manipulations had to be

made related to the links, especially those located at the main entrance of the church. Unfortunately, automatic target detection required great attention because it proposed many misidentified targets. This fact may have been due to the structure of the church walls, which were made of brick whose welds may have mistakenly resembled targets. After removing incorrectly identified points, adding a few undetected targets, and establishing consistent names using the Smart Label option, the entire point cloud was georeferenced using the Apply Control tool. Finally, the entire bundle was optimized using the Optimize Bundle tool, which allowed us to proceed to the final step before exporting the point cloud.
The final bundle includes 28 setups with 45 connections and 27 reference points. The overall result has an error of 0.003m, with a C2C error of 0.004m and a target error of 0.002m. These results can be considered satisfactory for our purposes.
The last step was to export the final point cloud to a single file in .e57 format, the oriented, separate positions in .e57 format, and the spherical panoramas. The second and third data will be used to perform the experiments more fully described in Section 3.3 of this paper.
Due to the file size and aesthetic values, the registered point cloud had to be cleaned of unnecessary surroundings. For this purpose, Autodesk ReCap software was used, in which, with the use of cutting tools, the part of the point cloud containing the church building was extracted. This extracted fragment was re-exported to .e57 format. However, it was inconvenient to use it for further experiments and integration due to the file size. In order to subsample it, it was loaded into CloudCompare software for using the Subsample a Point Cloud tool. It was filtered using the rule that points can be a minimum of 0.01m apart. This value was chosen empirically with the condition of significantly reducing the file without degrading the visual perception of the scanned object.

Table 1: Comparison of points clouds

Point cloud name	Points number	File size
Raw data	854 693 218	18.20 GB
Only church	355 645 548	9.31 GB
Subsampled Church (0.01m)	33 915 433	0.62 GB

3.2. UAV data processing

Popular software Agisoft Metashape 1.8.2 was used to process aerial photographs acquired using a UAV. First, all images were imported into the project, and a preliminary alignment was carried out at a reduced resolution of rasters. This type of alignment made it possible to filter out images where the selected GCPs were visible.
After GCPs in the images had been indicated, a final alignment of the image block was carried out. Two image block alignment approaches (Wiedemann, Moré J; 2012) were tested during the study:

1) Single-step alignment - for each pair of images in the block, the detection of tie-points and image matching was performed,
2) Multi-step alignment - the detection of binding points and image matching was performed for nadir images, and then the process was repeated for oblique images from

successive directions. Such a process is impossible in the base version of the software, so it had to be implemented using the Python programming language and the API provided by the manufacturer.

Simultaneously with aerotriangulation of the image block, internal orientation parameters and camera distortion coefficients were optimized. Initially, the optimization included all internal orientation elements, but due to significant reprojection errors, the coordinates of principal points (cx, cy) were finally taken as fixed. The alignment results (position RMSE obtained) are shown in Table 2, while Figure 3 presents per-point errors.

Table 2: Obtained accuracies

Alignment method	**Control Points**		**Check Points**	
	horizontal accuracy [m]	**vertical accuracy [m]**	**horizontal accuracy [m]**	**vertical accuracy [m]**
Single-step	0.034	0.117	0.041	0.140
Mutli-step	0.056	0.061	0.072	0.084

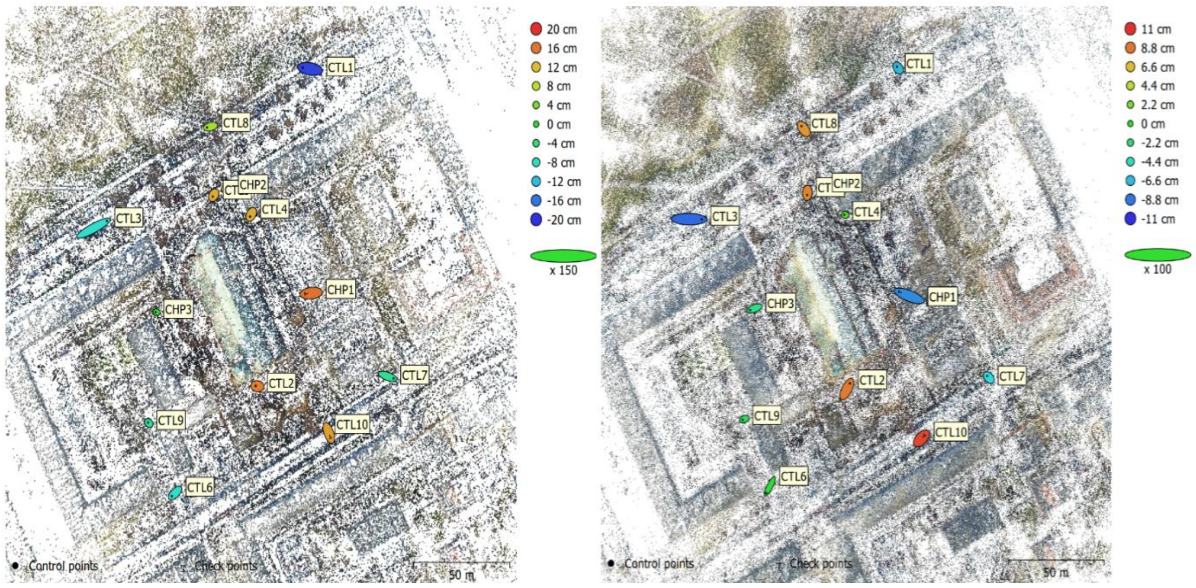


Figure 3: Single-step (left) and multi-step (right) GCP errors

Finally, using the multi-step alignment results, a dense point cloud of the church building and its immediate surroundings was generated, which served as data for constructing the final 3D model.

3.3. Multisensoral data integration

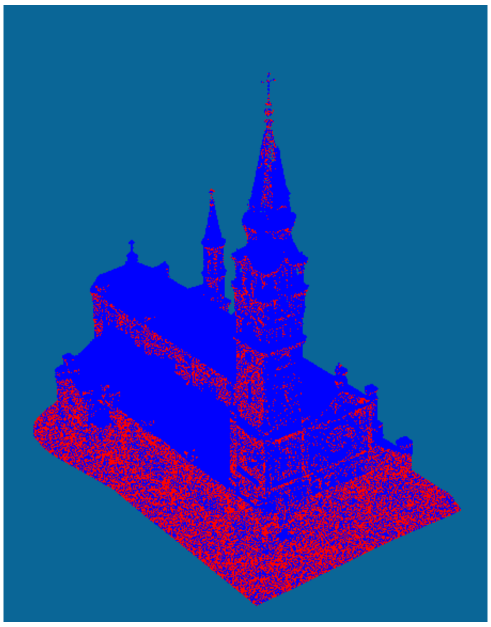


Figure 4: Merged point cloud visualized by data source: LiDAR (red) and UAV (blue)

Before integrating the acquired data, it is essential to note that the data are from periods with a time lag of 2 years. This was due to the start of the COVID-19 pandemic outbreak.

The integration process was done in the open-source software – CloudeCompare. After importing the UAV and TLS point clouds, the georeferenced point clouds were immediately oriented concerning each other. The first step was a visual consistency check of both clouds (Figure 4). Particular attention was paid to areas where the two clouds overlapped each other.

As shown in Figure 5, the original, unfiltered data have different density and measurement noise parameters. Figure 5 shows a section of the upper wall of the church's central nave where the two data types overlapped. It was cut using a cube of 1m side for analysis. A Gaussian fit plane was then fitted into the point cloud using the Fit Plane tool. The histogram shown in Figure 6 represents the distance of the points from the plane. The standard deviation is 0.008m. Two data types are visible: a compact, coherent, low noise LiDAR point cloud on the left and a sparser and higher noise cloud from photogrammetry.

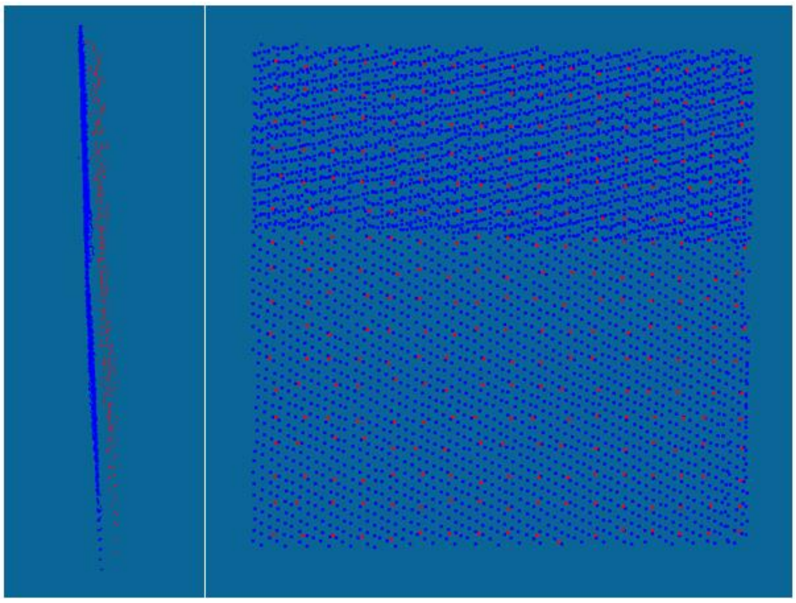


Figure 5: A fragment of the wall with both point clouds: LiDAR (blue) and UAV (red)

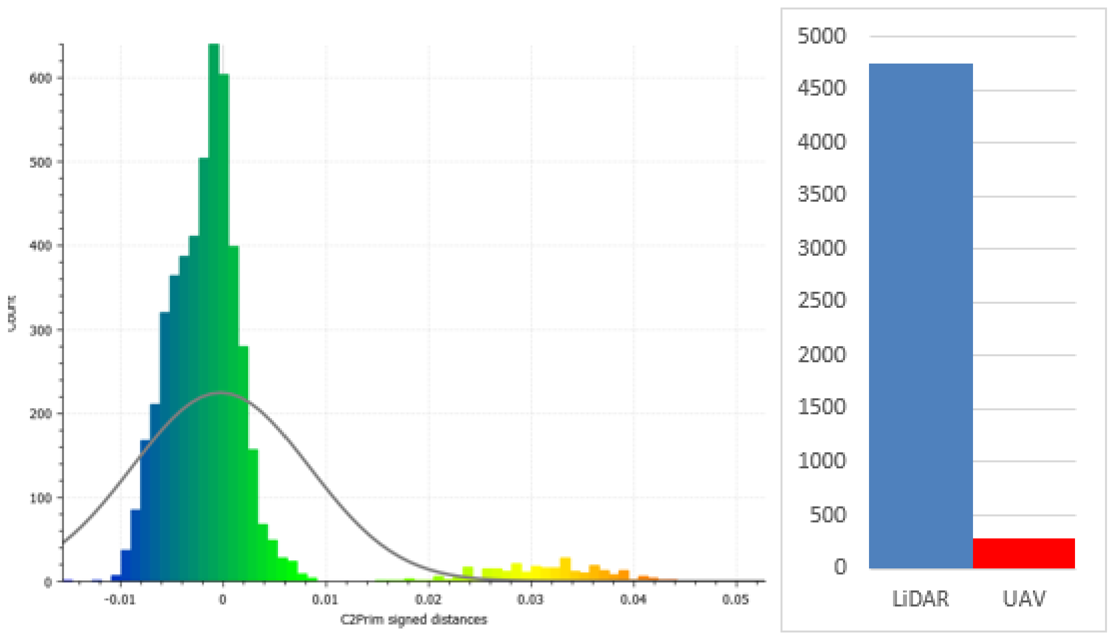


Figure 6: Histogram and diagram of point counts

The next step was to merge both point clouds into one coherent point cloud. For this purpose, the Merge multiple clouds function was used. In order to keep the point cloud entirely consistent, not only in terms of location, it was necessary to remove the saved points normals and then generate them from the beginning during the next steps. The consistency of the normals is crucial when generating 3D models, especially at locations common to both point clouds (bad and good features).

The prepared point cloud was re-imported into the project in Agisoft software containing oriented images. While importing the point cloud, consistent normals were generated for the entire point cloud, which was crucial for the correct construction of the mesh model. High parameters were used to build the mesh model, resulting in a model containing approximately 16 million planes. A surface type was Arbitrary 3D and enabled interpolation was also used as parameters, which helped to close holes caused by local missing data in the point cloud. The model included an uncleaned UAV cloud to texture the object of interest correctly. It was crucial to have a model of the surrounding trees that could obscure the main object of interest in the images. As confirmed by empirical experiments, the texturing process proceeded incorrectly in their absence. The model was also visually inspected at this stage, and sections generated incorrectly during the mesh model process were removed. This process was fundamental in the case at the tops of the towers capped with crosses (Figure 7). After this process, the Close Holes tool was used to close the holes in the mesh model created during the removal process.

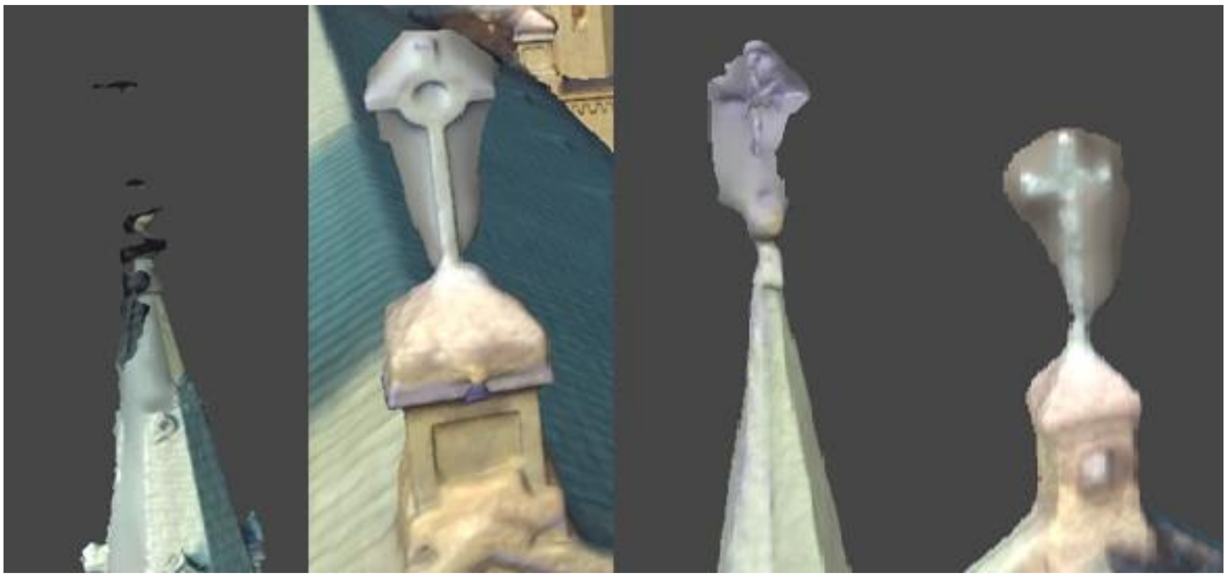
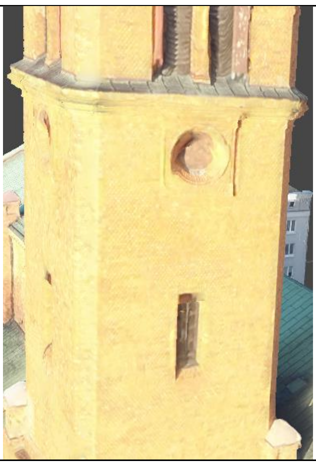
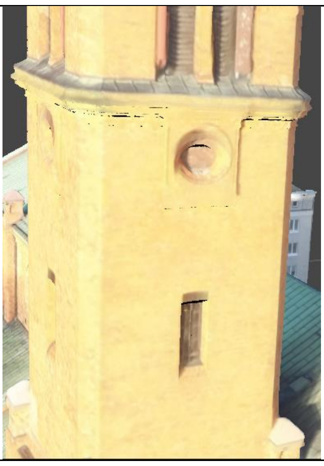
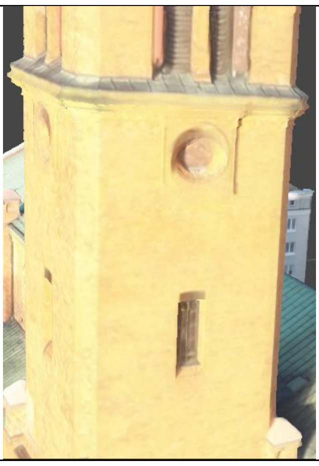


Figure 7: Incorrectly generated mesh model areas

The Build Texture process was used to generate the textures. In order to compare the quality, the model was textured using three processes differing in parameters, as shown in the table below (Table 3).

Table 3: Build Texture parameters

No.	I	II	III
Blending mode	Mosaic	Average	Average
Hole filling	No	No	Yes
Ghost filtering	Yes	-	-
Example			

As shown in the example above, textures I and III have minor differences, with a slight edge in sharpness in favor of option I, used in the final model.
One of the final phases of model processing was to remove the surroundings from the model so that the model includes only the building and the surrounding paved area. Finally, a Tiled Model was built from the mesh model and was exported to Cesium 3D Tiles format with a .zip extension.

4. WEB-BASED VISUALIZATION

4.1. WebGL API

WebGL (Web Graphics Library) is the JavaScript API (Application Programming Interface) for web-based render 2D and 3D graphics. Thanks to HTML5, it uses the Canvas element to render interactive models, and it has access to the DOM tree. Moreover, WebGL is based on OpenGL ES 2.0, an open-source API to generate graphics for many platforms. OpenGL has more than two hundred functions to create complicated 3-dimensional scenes from basic geometrical figures. Graphics rendering is way faster than other approaches because commands are being realized by GPU (Graphical Processing Unit).
WebGL has much functionality. Thanks to this API, one can create 2D or 3D models with different colors or add lighting to them. To models can be added texture, which can also be animated, like a whole model. It supports many popular 3D model formats, i.e., JSON, OBJ, FBX, PLY, and GLTF.
Every commonly used web browser implements this API, i.e., Mozilla Firefox, Google Chrome, Safari, Opera, or Microsoft Edge. So, the API has also spread out because various JavaScript libraries are implementing it to handle and manipulate 3D models (Three.JS, SceneJS, or CesiumJS).

4.2. CesiumJS

CesiumJS is an open-source JavaScript library for creating a 3D globe and interactive maps. Cesium provides accuracy, precision, ease of use, functions, features, high performance, and rich documentation. From geodesy to smart cities to drones, developers across industries use CesiumJS to create, share, and present geospatial data in 2D and 3D. The library was developed in 2011 by a team of developers at an aerospace software company to deal with visualizing objects in space. As an open-source platform, it was released in 2012. It offers a mix of open source and commercial software to build 3D geospatial dynamic applications. (Gede M., 2017).
Cesium provides lots of features, i.e.:

- Stream in 3D tiles and other standard formats
- Visualize and analyze on WGS84 globe
- Share with users on web desktop apps or mobile apps
- Visualize 3D models using glTF
- Load KML, GeoJSON, and CZML to draw a variety of objects and geometry
- Support time-dynamic simulation and 4D visualization

- GPU-accelerated 3D analysis tools

CesiumJS ensures interoperability and scaling for big datasets. It supports popular 3D model formats, for example: .obj, .fbx, .dae, .gltf, .glb. To properly upload any models, it is required to use local coordinates (the geometry must be centered around the origin). Files may be zipped, and units are assumed to be in meters. CesiumJS makes importing models possible by using Assets Server or hosting them as glTF in the local server.

4.3. Cesium 3D Tiles

The best solution is to implement a 3D church model using 3D Tiles. They are an open specification for streaming massive 3D geospatial datasets. 3D Tiles are commonly used to stream 3D content: buildings, point clouds, and any vector data. To upload geospatial datasets, CesiumJS render them using 3D Tiles. Datasets are converted to glTF by 3D Engines, the WebGL runtime asset format developed by Khronos. (Cozzi P., 2015)
3D Tiles are:

- Optimized for streaming and rendering
- Interactive
- Styleable
- Adaptable
- Flexible
- Heterogeneous
- Precise
- Temporal

4.4. Implementation

The first step to implementing created 3D church model is getting an authorization token by creating an account on the Cesium platform. When Cesium Authorization Server has made the token, we have access to the library's full functionality. The implementation starts with creating an HTML file. Then it is required to append scripts importing library with appropriate CSS styles. Next, in the body element, the <div> element has to be entered to store the entire content of the application. The next step is the initialization 3D globe with Cesium.Viewer() method and passing to its parameters: terrainProvider and imageryProvider. The first of them contains a built-in CesiumJS library WorldTerrain. The second one allows for adjusting the selected background of the visualized scene. In this application, OpenStreetMap was selected. It is a suitable background for the general appearance of the space, containing the main topographic elements (roads, geographic names, POIs, or buildings). Cesium OSM Buildings - a global 3D buildings layer, was added to diversify the visualized scene. Thanks to the OSM buildings layer style attribute, it is possible to use the show function, which visualizes only selected objects. In this example, the church OSM building is not shown. Figure 8 illustrates the basic implementation.

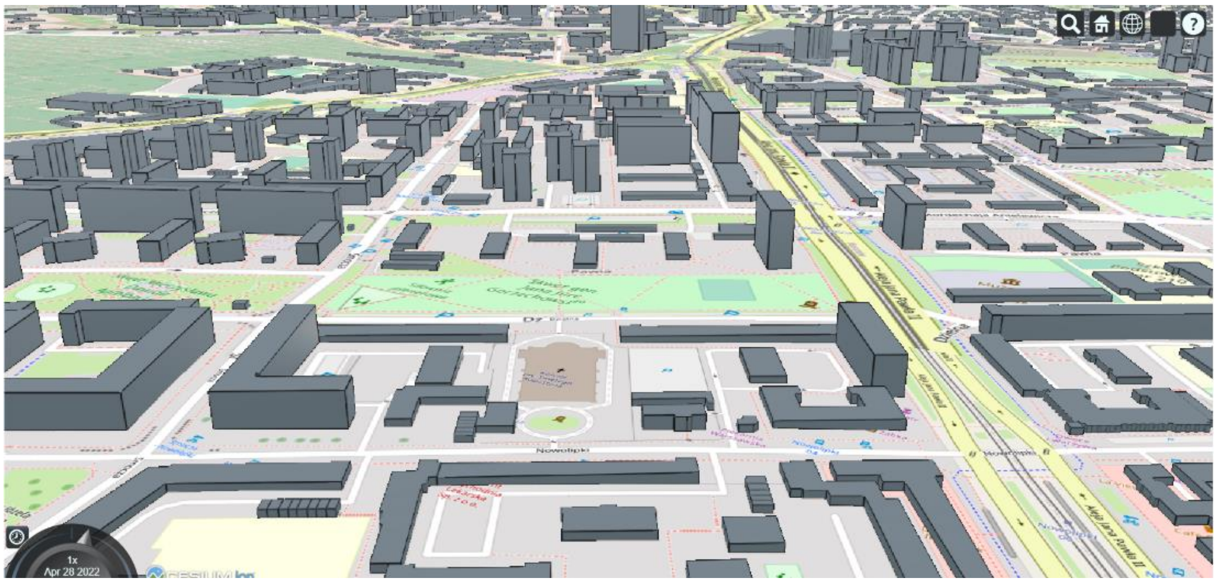


Figure 8: Basic implementation in Cesium

The last step is to import the 3D model of the church as a 3DTileset. The 3D model was added from the local repository using an asynchronous procedure. The CesiumJS library allows to implementation of a scene camera setup. Using the eastNorthUpToFixedFrame() method, we navigate to the appropriate place on the globe, while the lookAtTransform() method allows the camera position setup. Figure 9 illustrates the final implementation of the 3D church model using CesiumJS.

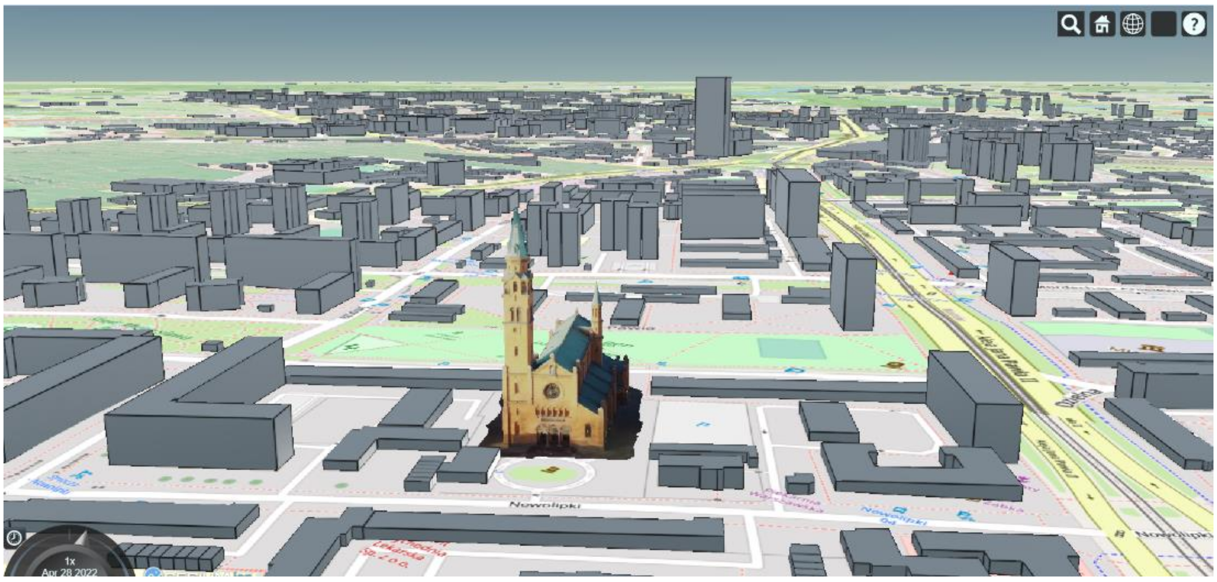


Figure 9: Final implementation of the 3D church model

5. SUMMARY

Undoubtedly, photogrammetry using UAVs provides an excellent tool for acquiring 3D data intended to create realistic building models for interactive visualization. Modern drones and flight planning software allow quick photogrammetric missions and data acquisition of very good radiometric and accuracy. For objects such as a church, oblique photos served an essential function in addition to vertical photos.
Terrestrial laser scanning provides a very dense and accurate point cloud. This quality is redundant for visualization purposes at this requirement level and implies the need for data sampling. However, TLS data is necessary to correctly represent the lower parts of the building and those parts that are invisible to UAVs.
With consistent georeferencing, data integration was fast and straightforward. An important step was the texturing of the 3D model, which required a 3D model for the immediate building environment as well, allowing for proper texturing.
The CesiumJS library makes it easy to share and visualize the created 3D models of topographic objects. The 3D Tileset data format used by Cesium enables high-performance rendering. Despite its many functions, the CesiumJS library still has some imperfections regarding the difference between the ellipsoidal height and the normal height. When exporting a model from Agisoft, the offset should be corrected using the CesiumJS library methods.

REFERENCES

Biason A., Moerwald T., Walser B., Walsh G.; 2019; A new approach to the Terrestrial Laser Scanner workflow: the RTC360 solution; FIG Working Week 2019

Cozzi P.; 2015; Introducing 3D Tiles; www.cesium.com

Gede M.; 2017; Using Cesium for 3D Thematic Visualisations on the Web; Proceedings of the International Cartographic Association

Wiedemann A., Moré J.; 2012; Orientation strategies for aerial oblique images. International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences, 39, 185-189

BIOGRAPHICAL NOTES

Jakub Gryzio is a bachelor's degree Geoinformatics student in Geodesy and Cartography faculty at Warsaw University of Technology. His interest in programming, web development, and database management. He is involved in the activities of the KNGiK research group and works in the Hexagon company.
Pawel Czernic is a master's degree student in Geodesy and Cartography at Warsaw University of Technology. He is an active member of the KNGiK research group and works for WUT. His interest in LiDAR technology, underwater photogrammetry, and technical innovations.
Łukasz Wilk is a master's degree student in Geodesy and Cartography at Warsaw University of Technology and a graduate bachelor of Geoinformatics. His everyday work revolves around photogrammetry, computer vision, spatial data machine learning, and big data processing. He has been a member of the KNGiK research group since 2018.

CONTACTS

Jakub Gryzio
Warsaw University of Technology, Faculty of Geodesy and Cartography
Plac Politechniki 1, 00-661 Warsaw
POLAND
+48 501 671 746
jakub.gryzio.stud@pw.edu.pl

BEng. Paweł Czernic
Warsaw University of Technology, Faculty of Geodesy and Cartography
Plac Politechniki 1, 00-661 Warsaw
POLAND
+48 790 717 835
pawel.czernic.stud@pw.edu.pl

BEng. Łukasz Wilk
Warsaw University of Technology, Faculty of Geodesy and Cartography
Plac Politechniki 1, 00-661 Warsaw
POLAND
+48 694 651 149
lukasz.wilk3.stud@pw.edu.pl