2025 IEEE 29th International Conference on Intelligent Engineering Systems (INES) | 979-8-3315-9771-9/25/$31.00 ©2025 IEEE | DOI: 10.1109/INES67149.2025.11078231

Lithostratygraphic Layer Generation based on LiDAR and Borehole Data

Máté Anderko*, Csaba Németh†, Sándor Szénási‡
John von Neumann Faculty of Informatics, Obuda University, Budapest, Hungary
*mate.anderko@stud.uni-obuda.hu, †infopeek@stud.uni-obuda.hu, ‡szenasi.sandor@nik.uni-obuda.hu
Faculty of Economics and Informatics, J. Selye University, Komárno, Slovakia
‡szenasis@ujs.sk

***Abstract*—This paper introduces the development of an innovative subsurface stratigraphy generator model, which integrates LiDAR and borehole data for the effective visualization of underground geological layers. The proposed software is designed to offer a straightforward, efficient, and user-friendly approach to generating 3D representations of subsurface stratifications, supporting applications in education, geological research, and geotechnical design. The model employs advanced computational techniques, including Delaunay triangulation and Kriging interpolation, to produce accurate and realistic visualizations. Specifically, the Kriging method demonstrates exceptional suitability for lithostratigraphic data interpolation, while Delaunay triangulation ensures robust triangulation of geological structures. The proposed tool represents a significant advancement in the understanding and visualization of subsurface stratigraphy, offering valuable applications in research, education, and industry-specific planning processes.**

***Index Terms*—stratigraphic modeling, LiDAR, Kriging interpolation, geological visualization**

I. Introduction

Traditional methods, such as manual analysis of stratigraphic maps or simple drilling data, are lengthy and give only an abstract idea of the nature of the strata. In contrast, modeling software can give the user an easily digestible visual representation. This research aims to build an underground stratification generator that can model an area using LiDAR and various borehole point data. An open-source implementation of the model is also presented.

II. Related Work

The field of stratigraphic modeling has seen significant advancements through various proprietary- and open-source tools. Software such as RockWare RockWorks [1] and Leapfrog Geo [2] provide robust modeling and visualization capabilities but often come with high costs, steep learning curves, and licensing constraints. The main targets of these highly specialized tools are advanced users with substantial technical expertise, which limits their accessibility for educational purposes.

In contrast, approaches such as GeoStudio and GOCAD offer targeted solutions, focusing on geotechnical analysis and reservoir modeling. However, their rigid implementation and dependence on specific data formats often make them unsuitable for exploratory or diverse geological studies. Recent research [3] employing ArcGIS and 3DSMax highlights simplified workflows but remains limited to environments requiring these specific commercial tools.

Kriging interpolation [4] is an advanced geostatistical method used to estimate values at unsampled locations based on known data points. It is particularly effective in spatial analysis, as it accounts for the spatial correlation between data points. Unlike simpler interpolation methods, Kriging not only predicts values but also provides a measure of the uncertainty or accuracy of these predictions. The process involves creating a mathematical model of spatial relationships and using this model to assign weights to nearby data points. These weights are then used to calculate the estimated value at a specific location. Kriging is widely applied in fields like geology [5], environmental science [6], simulation [7], and mining, where understanding spatial variability is crucial.

Delaunay triangulation is a geometric technique used in computational geometry to divide a set of points into triangles. It ensures that no point lies inside the circumcircle (the circle passing through all three vertices) of any triangle in the triangulation. This property maximizes the minimum angle of all the triangles, avoiding thin or sliver-like triangles and creating a more uniform mesh. This method is widely used in fields like computer graphics [8], geographic information systems (GIS) [9], and finite element analysis, as it provides an efficient way to model surfaces and structures.

Our study takes inspiration from these methodologies while aiming to create a cost-effective, open-source alternative. By utilizing widely accessible technologies such as LiDAR and borehole data, combined with efficient algorithms like Kriging interpolation and Delaunay triangulation, our work bridges the gap between academic research and practical application.

III. Methodology

The foundation of the software lies in its ability to integrate LiDAR-based Digital Elevation Models (DEM) with borehole data. These datasets provide a comprehensive understanding of the terrain and subterranean structures.

The software is developed in C++, utilizing the CMake build system for seamless integration and management. The database backbone, based on PostgreSQL with the PostGIS extension, ensures optimal, industry-standard spatial data handling. A lightweight Node.js API facilitates communication between the frontend and the backend server, providing users with a responsive and intuitive experience.

A. Data Management and Access

The data utilized is served from a centralized server instance, encapsulated within a Docker container to ensure streamlined deployment and portability. Data ingestion and preprocessing are automated via a PowerShell script that executes the following key steps:

1) Data Loading: All locally stored GeoTIFF raster files and drilling point data (in JSON format) are collected for further processing.

2) Raster Data Processing: Using the *raster2pgsql* command, raster image files are transformed into SQL tables. These tables follow a naming convention of ¡FILENAME¿-raster, enabling clear identification and referencing within the database.

3) Drilling Point Data Handling: To simplify data management, drilling point objects from multiple JSON files are merged into a single CSV file, easing subsequent database imports.

4) Database Import: The resulting CSV file is imported into the database using the *copy* command, allowing for efficient and high-performance data loading.

5) Geometric Transformation: Coordinates defined by EovX and EovY values are converted into PostGIS geometries. This enables spatial operations such as defining the intersection of given areas with selected sets of drilling points, leveraging LiDAR-based metadata. In response to a request at the */get-data/:name* endpoint, the server returns a ZIP archive containing the corresponding TIFF file and a JSON file with the filtered drilling points. This packaged dataset can then be processed on the client side.

B. Kriging Interpolation

Kriging is a geostatistical interpolation method that leverages spatial correlations among data points to estimate unknown values [10]. Its main steps are:

1) Variogram Construction: It computes the distances between each pair of points and the half-squared difference of their values. Then, it sorts the resulting pairs by distance to derive an empirical variogram, which shows how spatial correlation changes with distance [11].

2) Theoretical Function Fitting: The algorithm then fits a Gaussian theoretical model to the empirical variogram. This provides a continuous function that describes the spatial relationship and allows estimation at any location [11].

3) Covariance Matrix Construction: Using the fitted model parameters (nugget, sill, range), it creates a covariance matrix that quantifies how each known point correlates with every other point [11].

4) Matrix Factorization and Precomputation: To mitigate the estimation error due to the sparsity of data, the algorithm applies ridge regression. After this step, the covariance matrix is factorized via LU decomposition, and the matrix inverse is precomputed for efficient computation of interpolation weights.

5) Value Estimation: The algorithm then computes the covariance of each point with all known points and solves the resulting linear system to obtain weights for these points, and then computes the weighted sum of their values to estimate the unknown value. Using the sill, the variance vector, and the weights vector, it is possible to compute an uncertainty value, which the program uses in a later step.

This procedure results in an interpolation matrix and an uncertainty matrix that account for both the spatial distribution of known points and their measured values.

C. Reducing noise on the DEM raster

When modeling subsurface layers, unwanted surface irregularities (such as those caused by erosion) can distort the interpolation results. To address this, filtering algorithms are applied to reduce high-frequency details in the input data, ensuring smoother lower layers. Two common methods are:

1) Gaussian Filtering: Gaussian blurring is a classic image processing technique used to smooth noise and details. It is based on the Gaussian distribution and involves convolving the input data with a Gaussian kernel. Each output value is a weighted average of neighboring points, where the weights decrease with distance from the center point. Adjusting the kernel size and the standard deviation (σ) of the Gaussian allows for fine-tuning the smoothing effect [12].

2) Median Filtering: The median filter replaces the value of each point with the median of its neighborhood values. This approach effectively reduces noise while preserving edges under certain conditions. However, large median filter windows may introduce artificially round boundaries. A combination of median and Gaussian filters can yield improved results, especially for smoothing large valleys while maintaining a more natural appearance [13].

D. Shifting and finalizing layer data

Using the interpolated data and a blurred representation of the LiDAR map, all point positions are finalized for each layer; this is done by subtracting the interpolation value for the given position from the corresponding value of the raster. It is also necessary to ensure that the layers do not cross each other. This step also involves the uncertainty matrix. Based on the matrix values, the model tries to apply correction by factoring in the normalized values of the layer above.

E. Texture creation

1) Mesh Generation: The *MeshGenerator* is responsible for creating 3D meshes for layered structures by leveraging a *LayerBuilder* instance. This process is designed to be systematic and incremental, focusing on generating the necessary geometry layer by layer.

2) Triangulation: The first step involves triangulating the 3D points of the layer. This requires extracting the (x, y) coordinates from the 3D points for a 2D triangulation process while also preserving the z coordinates (elevations) for later use. These elevations are stored in a mapping structure for easy access. Using a computational geometry library such as *CGAL*, the system constructs a constrained Delaunay triangulation (CDT2). This step ensures that the edges and faces of the triangulation accurately represent the surface geometry of the layer.

3) 3D Surface Generation: Based on the triangulation, a 3D surface is constructed by mapping the z coordinates back to the vertices. The triangulated faces are converted into 3D triangles by incorporating the stored elevation data. The resulting 3D surface mesh encapsulates the geometry of the layer, ready for further processing.

4) Extrusion: To give the surface mesh a volumetric representation, it is extruded along the z-axis. The extrusion is performed downward, reaching a plane that lies below the lowest point of the bottommost layer. This step ensures that the volume of each layer is fully captured, forming a complete representation of its thickness and spatial extent.

5) Difference Calculation: The difference calculation is performed to separate adjacent layers. The extruded mesh of the current layer is subtracted from the extruded mesh of the layer immediately below it. This subtraction ensures that only the unique portion of the current layer's volume is retained. After the operation, the resulting mesh is validated for consistency, and the updated mesh becomes the final representation of the current layer. This process is repeated for each layer, starting from the bottom layer and proceeding upward.

F. Visualization

Visualization is a key aspect of the workflow, enabling users to inspect the generated meshes and verify their accuracy. The software employs the Visualization Toolkit (VTK) to render the 3D layers interactively.

1) Conversion: To prepare the generated meshes for rendering, they are converted from the CGAL *SurfaceMesh* format to *vtkPolyData*. During the conversion, vertex coordinates are extracted and transferred to a VTK-compatible format. Triangular faces are identified and mapped into VTK cells. This ensures that the mesh geometry is preserved while making it compatible with the visualization framework.

2) Rendering: The converted meshes, stored as *vtkPolyData* objects, are registered with the rendering engine for visualization (Fig. 1). The rendering engine then displays the 3D models in an interactive window, allowing users to explore and analyze the structure from multiple angles. This capability ensures that the system is not only accurate in its computations but also user-friendly and visually intuitive.

G. User Interface

The graphical user interface developed for this project is designed with a strong focus on accessibility, ensuring that even users with minimal experience in the field can easily navigate and use the application.

IV. Evaluation

A. Accuracy Assessment

1) Cross-Validation: The "leave-one-out" cross-validation method systematically omits each measured data point and then uses the remaining dataset to produce an estimate for the omitted point. By comparing the estimated value with the known measurement, it is possible to assess how well the interpolation technique performs. Specifically, for each iteration, the point under examination is temporarily removed, and this omitted data point serves as a reference. The variogram is then calculated on the reduced dataset, and the covariance matrix is derived. Using these results, Kriging is applied to estimate the value at the excluded point. The absolute difference between the estimated and actual values provides the estimation error for that location.

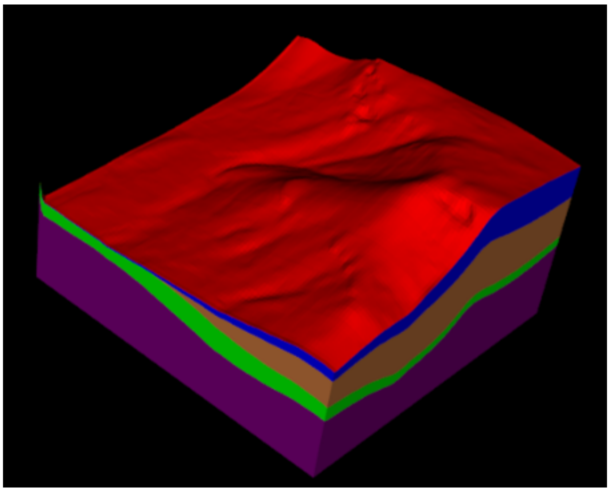

Fig. 1. Renderer output

TABLE I
Cross-validation results for the investigated layers

Layer	MAE	RMSE
Soil	3.43 m	4.43 m
Jakabhegy Sandstone Formation	45.10 m	69.73 m
Cserkút and Tótvár Members Combined	17.52 m	25.59 m
Kővágótöttös Sandstone Member	18.14 m	24.03 m

Repeating this process for all data points generates a series of differences that capture the overall quality of the interpolation. Statistical measures, such as the Mean Absolute Error (MAE) and the Root Mean Squared Error (RMSE), are then employed to summarize these errors. Table I presents the cross-validation results for the investigated layers from the studied area:

The results show that the Jakabhegy Sandstone Formation exhibits a notably higher average deviation than the other layers. Fewer known points are available compared to other layers, and the data originated from a single cluster. Consequently, the significant discrepancies can be attributed to the lower density and less diverse distribution of input data points. Predictions with large deviations more heavily influence the average error, and the outer boundary points tend to show more significant discrepancies due to these data limitations.

2) Validation Using Synthetic Data: One critical limitation of cross-validation using real-world data is the lack of ground

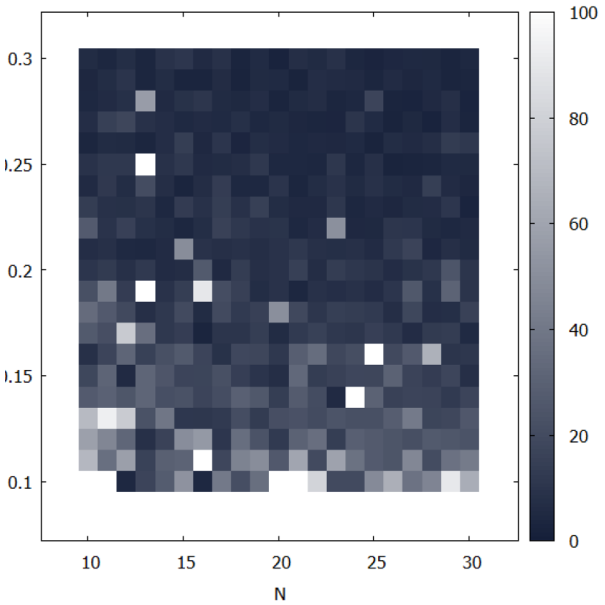

Fig. 2. Average deviation for synthetic layers

truth for each interpolated point. While real sampling locations and measurements are limited, this prevents a direct comparison between the estimated values and their actual reference counterparts. Synthetic datasets can be employed to address this issue, as they provide complete reference values across the entire domain. By generating a synthetic surface analogous to the subsurface layers of interest, we can perform an exhaustive evaluation of interpolation quality.

In this work, Perlin noise-based approach is applied to create a 100×100 synthetic elevation matrix, which simulates a continuous surface. Once this synthetic surface is generated, it is sampled to produce input data for the Kriging interpolation. To simulate varying borehole patterns and their inherent uncertainty, we select the sampling points using a normal distribution. The key aspects to consider are:

1) Number of Data Points (N): Determining the appropriate number of samples used for Kriging interpolation is crucial. In the experiments, N varies from 10 to 30 to assess how the density of input points influences interpolation accuracy.
2) Data Distribution: Predefining the spatial distribution of the data points ensures that our validation framework captures the randomness of drilling locations. We leverage normal distributions with different standard deviations (σ) ranging from 0.1 to 0.3 to control the spatial clustering or spread of sampling points.

By manipulating N and σ, we construct a series of synthetic layers (21×21 in size) and investigate how these parameters affect interpolation performance. The following figure shows the average deviation for each generated layer, along with the layer generation parameters.

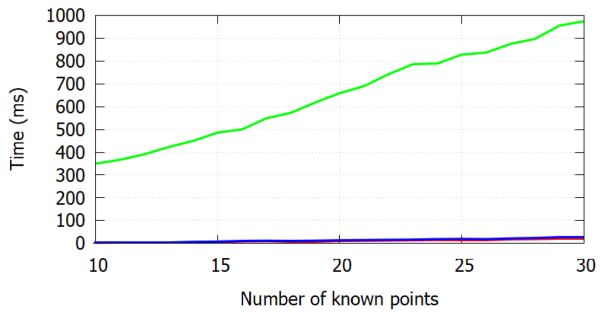

Fig. 3. Interpolation process for 10,000 points (red line: variogram calculation, blue line: covariance matrix calculation, green line: Kriging for all points)

B. Computational Performance

The computational efficiency of the model was tested by varying dataset sizes and complexity. Results showed that the system scales effectively, maintaining low computation times even for dense datasets. Optimizations such as multi-threaded processing and efficient memory management contributed to these outcomes.

1) Interpolation: The runtime of the interpolation process varies significantly based on the number of observed points. Testing reveals that as the number of known points increases, the relative distribution of time spent on different sub-processes within the interpolation also changes. For a test layer with a resolution of 10,000 data points, the results in Fig. 3 highlight how these proportions evolve.

2) Mesh Generation: Mesh generation is identified as one of the most time-consuming stages in the entire process, with an average runtime of 7.529 seconds over 100 runs. Reducing the number of points used, such as considering only every 10th point, significantly decreases the runtime to 3.489 seconds, less than half of the original. However, this optimization comes with trade-offs, notably a reduction in the output polygon resolution and the creation of more straightforward, flatter surfaces. Balancing runtime efficiency and output quality is a key consideration during this stage.

3) Rendering: Rendering the generated meshes is relatively fast, with an average runtime of 0.728 seconds across 100 runs. This efficiency ensures that the visualization component does not become a bottleneck in the workflow, allowing users to interact with and verify the results promptly.

V. Conclusions

This research presents the development of a subsurface stratigraphy generator model that uses LiDAR and borehole data to visualize underground layers effectively. The software aims to provide a simple, fast, and user-friendly way to create 3D models of underground stratifications, thereby assisting teaching, geological research, and geotechnical design.

The novel model is based on technologies such as Delaunay triangulation and Kriging interpolation, enabling the creation of accurate and realistic models. The Kriging method is particularly suitable for interpolating lithostratigraphic data, while Delaunay triangulation provides excellent results in triangulating geological structures.

Overall, the developed model can significantly contribute to the understanding and visualization of subsurface stratifications, thereby facilitating research and planning processes across various industries.

Future developments may include the integration of additional data types, more parallel data processing, and improved interpolation strategies.

Acknowledgment

The authors would like to thank the High-Performance Computing Research Group of Óbuda University for its valuable support. The authors would like to thank NVIDIA Corporation for providing graphics hardware for the experiments.

References

[1] K. MacCormack, "3-d modelling of quaternary sediments within the dundas valley, hamilton, ontario using rockworks 2002," 2005.

[2] T. Živec and M. Žibert, "The 3d geological model of the karavanke tunnel, using leapfrog geo," in *Proc: ITA-AITES World Tunnel Congress*, vol. 2016, 2016, p. 10.

[3] T. Fischer, D. Naumov, S. Sattler, O. Kolditz, and M. Walther, "Go2ogs 1.0: a versatile workflow to integrate complex geological information with fault data into numerical simulation models," *Geoscientific Model Development*, vol. 8, no. 11, pp. 3681–3694, 2015.

[4] G. Matheron, "Principles of geostatistics," *Economic geology*, vol. 58, no. 8, pp. 1246–1266, 1963.

[5] M. A. Oliver and R. Webster, "Kriging: a method of interpolation for geographical information systems," *International Journal of Geographical Information System*, vol. 4, no. 3, pp. 313–332, 1990.

[6] M. R. Holdaway, "Spatial modeling and interpolation of monthly temperature using kriging," *Climate research*, vol. 6, no. 3, pp. 215–225, 1996.

[7] W. C. Van Beers and J. P. Kleijnen, "Kriging interpolation in simulation: a survey," in *Proceedings of the 2004 Winter Simulation Conference, 2004.*, vol. 1. IEEE, 2004.

[8] S. Dinas and J. M. Banon, "A review on delaunay triangulation with application on computer vision," *Int. J. Comput. Sci. Eng*, vol. 3, no. 2, 2014.

[9] V. J. Tsai, "Delaunay triangulations in tin creation: an overview and a linear-time algorithm," *International Journal of Geographical Information Science*, vol. 7, no. 6, pp. 501–524, 1993.

[10] P. V. Arun, "A comparative analysis of different dem interpolation methods," *The Egyptian journal of remote sensing and space science*, vol. 16, no. 2, pp. 133–139, 2013.

[11] K. Johnston, J. M. Ver Hoef, K. Krivoruchko, and N. Lucas, *Using ArcGIS geostatistical analyst*. Esri Redlands, 2001, vol. 380.

[12] F. Bozkurt, M. Yaganoglu, and F. B. Günay, "Effective gaussian blurring process on graphics processing unit with cuda," *International Journal of Machine Learning and Computing*, vol. 5, no. 1, p. 57, 2015.

[13] R. Chandel and G. Gupta, "Image filtering algorithms and techniques: A review," *International Journal of Advanced Research in Computer Science and Software Engineering*, vol. 3, no. 10, 2013.