INTEGRATION AND USAGE OF ARTIFICIAL INTELLIGENCE SYSTEMS IN PARAMETRIC GENERATION / AN EXPERIMENT/

Roland Edgar Vasiliu - *"Ion Mincu" University of Architecture and Urban Planning, Bucharest, Romania.*

Abstract. This article presents the development process of a parametric application for design and architecture, designed to generate customizable three-dimensional structures through a modular, spiral, and deformable logic. The software tool enables the creation of complex spatial forms by controlling geometric parameters and applying spatial transformations within an intuitive graphical interface.

The application was developed in Python [1], using essential open-source libraries: PyQt5 for the graphical interface, Trimesh for geometry generation, and Open3d for real-time interactive visualization. The project began with the procedural generation of spiral-stacked cubic modules and gradually evolved to include various geometric primitives, advanced deformation techniques, and scaling in real physical units. Additional features, such as wireframe visualization and .OBJ export, ensure compatibility with external 3D design workflows [2] (e.g., Blender, Rhino).

This experimental project, with an applied research component, demonstrates how parametric thinking and accessible computational tools can be integrated into a unique generative framework, enabling real-time exploration of architectural form and beyond. The resulting application provides both pedagogical value and professional applicability in the field of computational design and digital prototyping, up to a physical development stage including 3D printing.

The experiment illustrates the applicability of artificial intelligence programs and thus becomes a potential starting point for integrating and using these new technologies in future applications that can generate or analyze structures from the conceptualization process of forms.

Introduction. In recent years, procedural generation of three-dimensional forms has gained increased interest in architecture, engineering, product design and the visual arts. Software applications that enable parametric modeling of structures offer high levels of flexibility, efficiency and creativity, being essential for rapid prototyping, form studies, spatial simulations and computer-aided manufacturing. These tools are no longer just a technical support, but become co-authors of the design process, allowing fast exploration of variants and alternative scenarios (Jaechang Ko, John Ajibefun, Wei Yan, 2025, Generative AI-powered parametric modeling and BIM for architectural design and visualization).

The present project aims to develop a parametric 3D application dedicated to generating deformable spiral structures and, in a more advanced stage, volumetric fractal networks that grow along the same trajectory. The user controls the real physical dimensions of the modules (width, height, depth), deformation (twist and noise), global curvature (bend on the X and Z axes), the level of fractal branching and the base geometric form. The application exports models in .OBJ format [3], compatible with external software such as Blender, 3ds Max or Unity, opening possibilities of integration into complex workflows including rendering, simulation or virtual reality.

The following sections describe the methodology, the software architecture, the implemented functionalities and the obtained results, as well as future development directions.

Project Objectives. The main objectives were initially formulated around the idea of parametric generation of a spiral building and were later expanded to include advanced deformations and fractal developments. Thus, the project aims to:

- develop an intuitive graphical interface, based on PyQt5, that allows direct control of the main parameters of the generated 3D form through sliders, numeric fields and dropdown selections;

- generate spiral structures procedurally by interpolating position and rotation parameters for each individual module, with the possibility to control the number of floors and the characteristics of the helical trajectory;

- implement modular deformation using twist and noise effects for obtaining more organic forms, and add a progressive bend on the X and Z axes that gradually curves the structure with height;

- control the physical dimensions of spiral modules via numerical inputs (QDoubleSpinBox) for width, height and depth, expressed in real units (millimetres, centimetres, metres), with interpolation between the base and top dimensions;

- introduce an extended set of geometric primitives (cube, sphere, torus, cylinder, tube, pyramid, frustum pyramid, cone, frustum cone), which can be used as base modules of the building or of the volumetric network;

- implement real-time manual rotation of the generated model through mouse manipulation, including a quick "Reset View" button and zoom via mouse scroll;

- implement a fractal component that allows the generation of volumetric branches at each module, controlled by parameters such as fractal depth, branch scale and number of branches per module;

- provide export of the generated models in .OBJ format through a "Save As" function that offers control over file name and location;

- prepare the application for distribution and use on other computers by generating a standalone executable with PyInstaller, eliminating the need for a manually installed Python environment.

Methodology. *Software architecture.* The application was developed in Python 3.x, using PyQt5 for the graphical interface and PyOpenGL for rendering the 3D forms. The architecture is structured into two main components:

– GLWidget: a component derived from QGLWidget, responsible for initializing the OpenGL context, drawing the 3D forms, applying

deformations (twist, noise, bend), generating the fractal network and exporting the model in .OBJ format;

– MainWindow: the main window component, which organises the interface (sliders, comboboxes, numeric fields, buttons) and manages the connection between user actions and the update of the 3D scene.

Communication between the interface and the graphic engine is handled through the PyQt5 signal/slot system. Each time a parameter changes (for example, when moving a slider or changing the unit of measurement), MainWindow calls the update_params() method of GLWidget, passing a consistent set of values: number of floors, amplitude, frequency, torsion, vertical offset, start and end dimensions, bend parameters, fractal parameters, selected shape, display mode and unit scale. GLWidget updates its internal fields and redraws the scene in real time, ensuring a fluid interactive workflow.

Generating the spiral form and the volumetric network. The basic spiral form is generated by successively placing 3D primitives along a helical trajectory. For each module i, the position is computed parametrically as a function of a parameter t between 0 and 1, related to the total number of floors. The X and Z coordinates are obtained using trigonometric functions (sine and cosine) that incorporate the amplitude and frequency of the spiral, while the Y coordinate is linked to the effective height of the module, resulting in a vertical growth without gaps between forms.

In addition to the main spiral trajectory, the application introduces a volumetric fractal network. For each main module, secondary branches can be generated, positioned radially around the module and scaled by a controllable factor. The fractal depth determines how many levels of branching are generated, while the number of branches per module controls the density of the network. In the implemented version, a balance between expressiveness and performance was sought, so branches are generated at the level of the main module, with the possibility of later extending towards full recursive branching.

Dimension interpolation along height (width, height, depth from base to top) is integrated both in the main modules and in the fractal branches, allowing scenarios in which the structure becomes gradually thinner or more massive vertically.

Form deformation: Twist, Noise and Bend. To overcome the rigidity of a simple modular repetition, the application implements several types of deformation. Twist is applied as a progressive rotation around the vertical axis, so that the angle of each module increases gradually as we move upwards along the spiral. The twist parameter is controlled by a slider, and the effect is also propagated to fractal branches. Noise is introduced as a controlled random variation of vertices, especially in the case of deformable cubes and spheres, in order to obtain more organic forms. Noise intensity is regulated through a numeric parameter, avoiding excessive distortions that would compromise the legibility of the architectural shape.

Bend is implemented as a gradual curvature of the growth trajectory, applied on the X and Z axes. Two independent sliders control bend on each axis, and the deformation is computed as a function of the height of the module, such that the structure appears to lean progressively, generating arched or almost vegetal forms. This bending system combines coherently with the spiral, twist and fractalisation, resulting in rich and expressive geometries.

Parameter control and graphical interface. The graphical interface was designed so that the user can control a large number of parameters without getting lost in complexity. The number of floors, amplitude, frequency, torsion, vertical position (offset), twist intensity and noise intensity are controlled via sliders. The physical dimensions of the base and top modules (width, height, depth) are entered through QDoubleSpinBox widgets, expressed in real units, and a dedicated combobox allows choosing between millimetres, centimetres and metres.

The geometric form is selected from a dropdown menu that includes all implemented primitives (cube, sphere, torus, cylinder, tube, pyramid, frustum pyramid, cone, frustum cone), enabling fast exploration of the impact of each primitive on the global volumetry. The display mode can be switched between wireframe and solid, to emphasise either structure or solid volume.

The fractal component is controlled through three main parameters: fractal depth (number of levels of branching), branch scale (expressed as a percentage) and number of branches per module. Together, these can transform the structure from a simple spiral into a complex volumetric network, which can be interpreted as a metaphor for growth, densification or urban aggregation. Zoom via mouse scroll and free rotation with the mouse complete the direct interaction with the model.

Real-time rotation and interaction. Model rotation is achieved by capturing mouse drag movements on the horizontal and vertical axes, updating the rotation angles stored in GLWidget. A dedicated button allows quick reset of the view, returning the model to a neutral reference orientation.

The introduction of zoom via scroll significantly improved how users can explore structural details. The zoom parameter controls the camera translation on the Z axis, allowing approach or retreat from the model without altering the real dimensions of the geometry. All these interactions are designed to be fluid and immediate, with no perceptible delay.

3D model export and application distribution. The generated model is exported in Wavefront .OBJ format, preserving the positioning, dimensions and deformations of each module. The export function was improved by introducing a “Save As” dialog that lets the user choose a file name and destination directory. In this way, users can easily organise collections of generated models and integrate them into external rendering or simulation environments.

For distributing the application on other computers without manually installing all Python dependencies, PyInstaller was used to generate a standalone executable. This executable includes the Python interpreter and all required libraries [4], enabling the application to run on compatible Windows systems simply by launching the .exe file.

Results. *Presentation of the developed application.* The main result of the project is a functional parametric 3D application capable of generating deformable spiral structures and volumetric branched networks in real time. The user can adjust a large number of parameters and immediately see their impact on the spatial form, supporting an exploratory, almost playful, way of working. Tests have shown that the application can generate both regular structures with constant dimensions and repetitive forms, and complex structures with continuous variations of dimensions, pronounced bend, high twist and dense fractal networks. Export in .OBJ format was verified by importing the models in external applications (Blender, 3ds Max, Unity), confirming compatibility and geometric correctness.

Implemented and tested functionalities. The implemented and tested functionalities include:

– generation of a parametric spiral of floors with control over real height, amplitude, frequency and torsion;

– interpolation of physical dimensions from base to top for width, height and depth, expressed in selectable real units;

– selection of the base geometric form from an extended list of primitives (cube, sphere, torus, cylinder, tube, pyramid, frustum pyramid, cone, frustum cone);

– application of twist, noise and bend (on the X and Z axes) combined with the spiral trajectory;

– generation of a volumetric fractal network through branches placed radially around the main module, controlled by fractal depth, scale factor and number of branches;

– interactive rotation with the mouse, zoom via scroll and quick reset of the view;

– export to .OBJ format through a "Save As" function;

– preparation of a standalone executable for distribution to other computers.

Conclusions and future perspectives. *Conclusions.* The project has met and even exceeded its initial objectives, evolving from an application for parametric generation of a spiral building into a much more complex tool capable of producing parametrically controlled volumetric fractal networks [5]. The combination of real physical dimension control, advanced deformations and fractal generation turns the application into a digital laboratory for exploring architectural and spatial forms.

The intuitive graphical interface, together with real-time interaction through the mouse and zoom, offers a fluid and accessible working experience both for advanced users and for students or creative explorers. Direct export to .OBJ format and the possibility of running the application as a standalone executable facilitate integration of results in diverse workflows, from rendering to virtual reality and digital fabrication.

Future perspectives. Building on the established foundation, there are numerous directions for continuation and refinement. One is to extend the fractal component towards multi-level recursive branching, with the possibility of defining L-system style rules for growth. Another direction is the integration of textures and materials, potentially through procedural generation, to bring the models closer to photorealistic rendering scenarios [7].

Additional export formats such as .GLB or .FBX would enable direct integration into virtual or augmented reality environments, as well as use in game engines. One can also imagine modules for simplified physical simulation (for example, approximating loads or structural behaviour), as well as advanced interfaces that support saving and loading parameter presets, facilitating comparison between design variants.

In education, the application can be used as a teaching tool to illustrate concepts of geometry, parametric design, fractals and the relationship between numerical representation and spatial morphology.

The most important outcome of this experiment is the complete process of designing, developing and testing a program built from scratch, in which artificial intelligence was used as a tool/assistant for programming, supporting both the writing of the source code and the completion of the final software application. This methodology also contributes to a much deeper understanding of how AI models operate and how they can be guided by humans. In this context, the initial concept and the first ideas remain essential within the creative process. It is crucial for the human actor to define as clearly as possible their expectations, objectives and the intended development direction of the project. Thus, the entire journey – from installing the programming languages and required libraries, to the actual implementation of the software, its iterative testing by the user and, finally, packaging it as a standalone installer – becomes a potential working model that can be reused in other relevant fields, not only in architecture or design.

References.

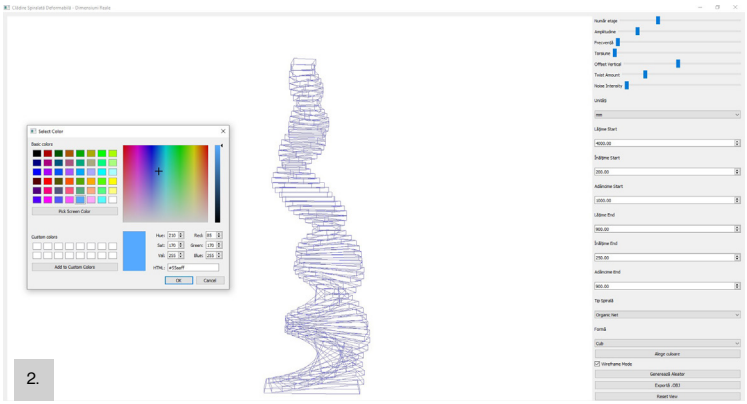
1. Riverbank Computing Limited, PyQt5 Documentation, [Online]. Available at: https://www.riverbankcomputing.com/static/Docs/PyQt5/
2. PyOpenGL Developers, PyOpenGL Documentation, [Online]. Available at: https://pyopengl.sourceforge.net/
3. Wavefront Technologies, The OBJ File Format Specification, [Online]. Available at: https://www.fileformat.info/format/wavefrontobj/egff.htm
4. Python Software Foundation, Python 3.x Documentation, [Online]. Available at: https://docs.python.org/3/
5. https://www.bispublishers.com/ai-assisted-architectural-design.html
6. https://www.cambridge.org/core/journals/proceedings-of-the-design-society/article/generative-aipowered-parametric-modeling-and-bim-for-architectural-design-and-visualization/A76987B083B87FA9E8579B1DCA532A0B
7. https://www.researchgate.net/publication/ 325758928_Machine_learning_for_architectural_design_Practices_and_infrastructure#:~:text=Intersections%20between%20ML%20and%20simulation%20can%20enable,as%20generative.%20design%20tools%20with%20structural%20(Karamba.

Figures.

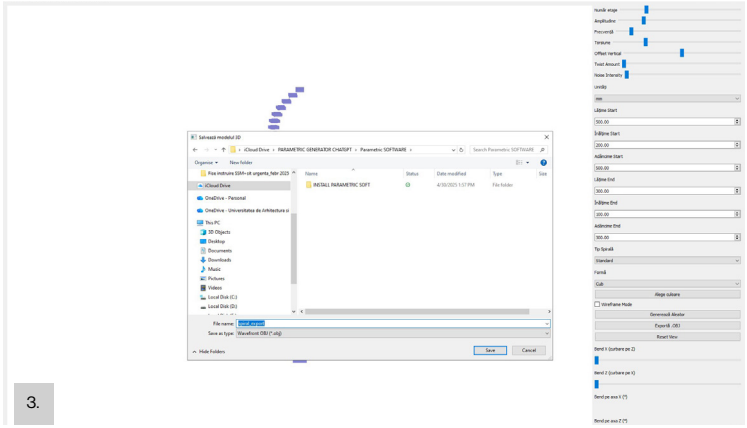
FIGURE 1 – Parametric
FIGURE 2 – Parametric
FIGURE 3 – Parametric



1.



2.



3.

FROM 'ALPINE VOIDS' TO REGENERATIVE DESIGN: THE SILENT FORTRESSES OF THE MOUNTAIN LANDSCAPE/

Anna Kerschbaumer - *Università degli Studi di Bari Aldo Moro - PasaP med, DIRIUM (Dipartimento di Ricerca e Innovazione Umanistica), Str. della Torretta, 70122 Bari, Italy.*
a.kerschbaumer@phd.uniba.it

Abstract. Located in the mountainous heart of the Eastern Alps, the Tyrolean region constitutes a historic borderland marked by an exceptional concentration of fortified architectures. Castles, towers, and defensive structures – predominantly medieval but often layered over earlier foundations – form a stratified system deeply intertwined with the area's complex orography. Their distribution reflects the region's long-standing geopolitical relevance as a strategic corridor between the Mediterranean and continental Europe, where topography historically enabled control of land and river routes. Within the framework of the call for papers Regenerative Design for the Built Environment and Urban Voids, this contribution proposes a rethinking of abandoned fortifications as a distinctive category of 'Alpine voids': mountain spaces stripped of function and progressively disconnected from the territorial and social systems that once sustained them. Once central to community life – structuring settlement patterns, agricultural organisation, and local identities – these sites have been marginalised by military decommissioning, infrastructural transformations, and the limitations imposed by their own topographical contexts. Preservation practices, often guided by rigid conservationist paradigms, have further struggled to re-establish meaningful contemporary relevance. Yet this condition of suspension conceals latent potential. Despite their apparent incompatibility with contemporary sustainability discourse, many fortified structures embody a forgotten form of environmental intelligence, grounded in careful adaptation to climate, topography, and resource constraints. Their spatial logic thus offers valuable insights for regenerative approaches in mountain territories. The following questions arise: Have these ruins truly reached the end of their life cycle? Can fortified heritage be reframed within regenerative design? By envisioning fortifications not as static relics but as adaptable components of a broader landscape system, the paper explores strategies for their reintegration as cultural, ecological, agricultural, or social nodes. Through context-sensitive design, Alpine voids can become catalysts of sustainable and inclusive territorial regeneration, reconnecting heritage, environment, and contemporary community needs.

Introduction. In the collective imagination, the term 'urban void' evokes images of absence. Derelict factories, disused rail yards, empty lots: abandoned fragments of the city where activity has ceased and meaning seems to have eroded. These spaces are the physical evidence of how urban and territorial systems evolve, forget, and renew themselves: each void records a pause in continuity, the trace of an exhausted function or a shifted paradigm. But what truly defines an urban void? Is it a condition bound to the city, or does it rather describe a broader phenomenon, a space marked less by geography than by disuse, disconnection, and loss of social relevance? Can voids exist outside the conventional urban domain, perhaps in territories whose apparent remoteness conceals similar dynamics of abandonment and potential?

This paper addresses these questions by extending the notion of 'the urban' beyond the city's borders. Through a focus on the Alpine context, it introduces the possibility of reading the territory itself as an expanded urban field: a continuum of settlements, infrastructures, and landscapes that interact with one another. Within this perspective, even non-metropolitan environments may reveal conditions analogous to those of the city's voids. For this fate of forgetfulness and disuse is not solely confined to factories or railyards: it may also affect other places that were once integral to the urban realm but have been progressively cut-off from it; sites once essential to human activity and collective identity, now emptied of use and isolated, having lost their role and centrality within contemporary life.

Many such spaces survive in the Alps in the form of disused fortifications: castles, towers, and military structures that once shaped the evolution of the mountain territory but now seem condemned to an inevitable condition of decline. These places will here be referred to as 'Alpine Voids': territorial voids embedded within the mountain landscape, defined by an evident interruption of relationships between form and use, memory and inhabitation, past and present. Despite their formal protection as heritage, such places often linger in a state of suspension: legally preserved yet substantially forgotten, visible yet disconnected from the cultural and environmental organism that once sustained them. Their neglect often stems from a convergence of factors: not only the obsolescence of their original purpose, but also shifting patterns of accessibility, territorial reorganization, or even, at times, the deliberate will to forget histories of conflict and control. Yet their marginality and exclusion need not be read solely as decay. They can also be understood as a state of latency: a pause that conceals potential and invites transformation. Seen through the lens of regenerative design, Alpine voids may be reinterpreted as fertile interstices where regeneration might take root. In addition to this, the accurate study of fortified structures may even uncover a deeper level of value: if observed closely, these voids may reveal a hidden, site-specific form of environmental intelligence from which contemporary designers can still learn. To reinterpret these spaces today means to unveil that latent value and reactivate their capacity to mediate between past and present, memory and reuse, heritage and ecology.

The Tyrolean Region as a Case Study: A Geography of Defence. Building upon the notion of Alpine Voids introduced above, the Trentino - South Tyrol region offers a tangible case in which this condition becomes visible. Located in the mountainous heart of the Eastern Alps, it presents itself as a territory where traces of the past endure in the form of fortified architectures. From prehistoric hilltop enclosures – such as the South Tyrolean Wallburgen, erected to protect early settlements and communication routes – to medieval castles commissioned by prince-bishops and noble dynasties, each epoch has inscribed its own architectural layer upon the landscape. Later centuries introduced aristocratic residences, particu-