

3D Editor for Archeology

Tomas Suchanek
xsucha01@stud.fee.vutbr.cz

Faculty of Electrical Engineering and Computer Science
Technical University of Brno
Brno / Czech republic

Abstract

This contribution explains the project of 3D editor for archaeology. The work included selection of suitable implementation platform, design of specialized functions for archaeology, and implementation of these functions in MicroStation environment.

Keywords: 3D editor, archaeology, MicroStation, MDL.

1. Introduction

This work initially started after discussions with archaeologists from Philosophical Faculty of Masaryk University in Brno. They had an idea to create some tool for reconstruction of original buildings form founded reminders.

The goal of archaeological research is understanding to past forms of people life their habits tools and houses. Traditionally, maps of archaeological fields are made using the reminders of foundations. They as precisely ad possible document artefacts found on the fields. Having the maps, archaeologists try to restore historical villages using their imagination. They usually work with pencil and piece of paper or paper models (see Figure 1). Such work typically takes very long time.

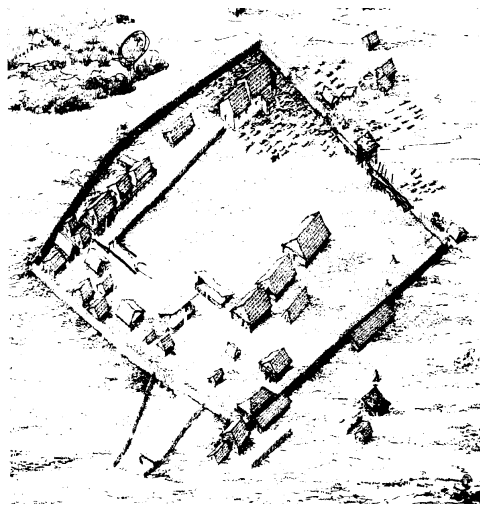


Figure 1: Global view of one locality

Archaeologists at Philosophical Faculty in Brno use computers for creating the maps. By having the maps on the computer, a good chance exists to reconstruct some form of 3D model of the sites

and eventually to visualise it. Unfortunately, the process of creation of the 3D model from the map, cannot be fully automatic due to shortage of the height information; however, computer can be used to make the user-assisted process of creation of 3D model easy by automation of typical operations. Such models could be visualised delivering the results and possibilities that were previously totally unimaginable – starting from "common" realistic display and enabling functions such as computing light parameters from information about placement on the Earth and time. Such functions can be very important in studies of the functionality of the buildings, e.g. for study of various religious ceremonials.

Another product of this work is a tool that makes easy creation of 3D model for archaeologists. The above described editor can also be used for reconstruction of complete villages (groups of buildings). The editor was named ArcheoBuilder.

2. Selection of implementation platform

During creation of the editor was needed to select an appropriate platform for implementation. Two systems were considered. AutoCAD from Autodesk and MicroStation from Bentley Systems and Intergraph.

AutoCAD is multifunctional CAD system expanded typically in general engineering. AutoCAD is implemented only on PC architecture (Windows NT, MS-Windows, MS-DOS). It provides all typical operations for 2D design. For 3D design, it provides possibility for modelling surfaces and solid bodies (see Figure 2) by using predefined objects or by rotation and projection of 2D objects. AutoCAD includes an embedded programming tool called AutoLisp. It is an analogy to traditional functional language Lisp. It is, however, not very comfortable for application development.

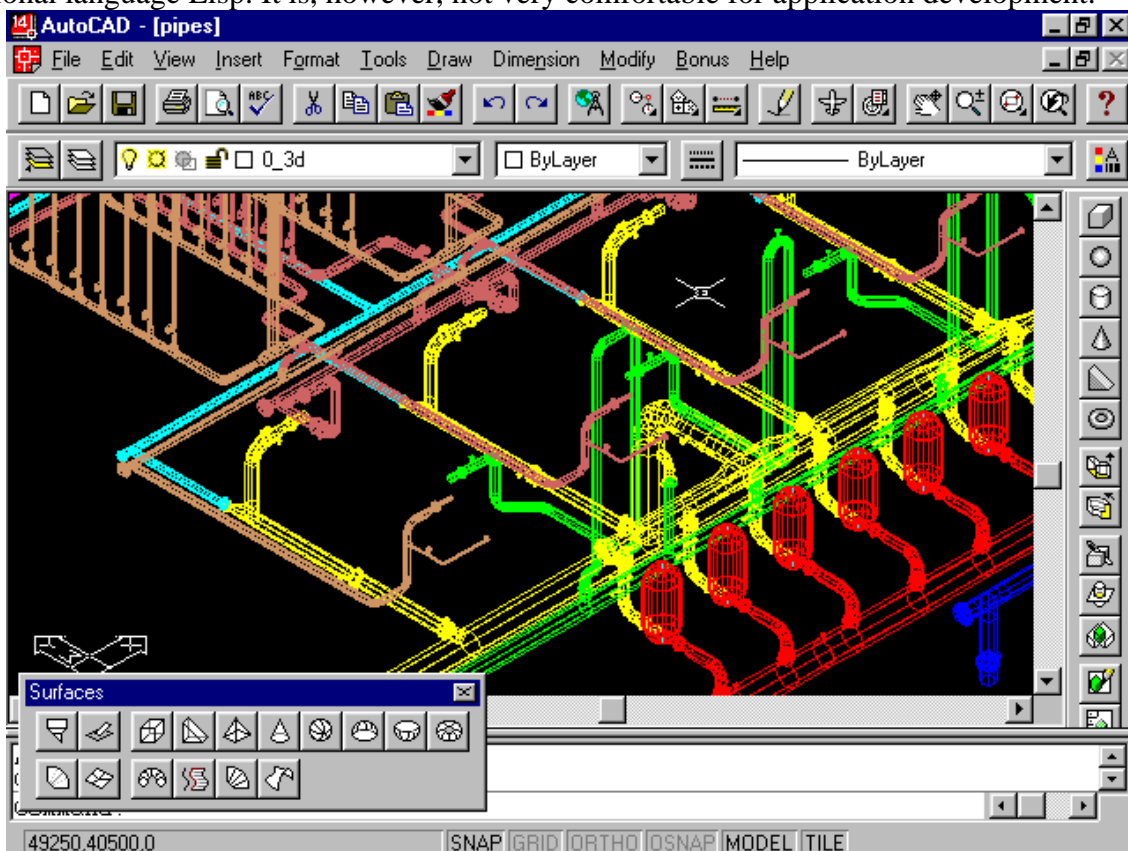


Figure 2: User interface of AutoCAD R14

MicroStation is multifunctional CAD system, too. But it is expanded typically in building industrial blocks, architectural structures and surveys. It is implemented on several computer platforms (Unix, Windows, DOS). For 2D design, it provides all common operations including modeling B-spline curves. MicroStation allows connection drawings to some large database systems. For 3D design MicroStation provides only modeling of surfaces (see bottom left part of Figure 3). Solid objects are represented by their surface. This fact can cause problems in modeling of buildings where it is needed to know solid parameters of modeled bodies for simulation mechanical properties. But commercial supplements exist than professionally solve this problem (e.g. MicroStation Modeler). For developing applications is programming language MDL embedded into MicroStation. MicroStation Development Language (MDL) a complete development environment is available that lets applications take full advantage of the power of the MicroStation CAD engine.

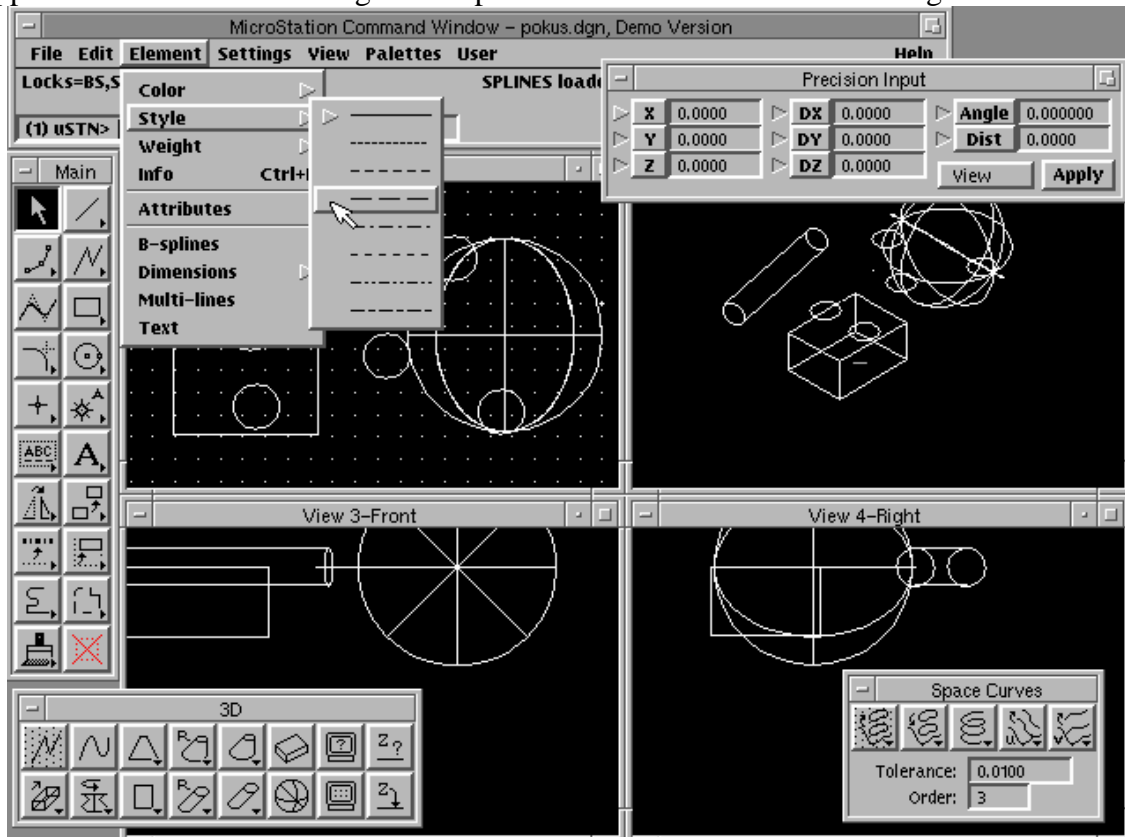


Figure 3: User interface of MicroStation 4.0

Finally, MicroStation was chosen as the implementation platform for the following reasons:

- It is used at Philosophical faculty in Brno for drawing maps,
- It has better programming tools (see chapter 0),
- It is supported at more platforms,
- It is better oriented to architecture

3. MicroStation Development Language - MDL

MicroStation Development Language (MDL) is a complete development environment that lets applications take full advantage of the power of the MicroStation CAD engine. MDL can be used to develop simple utilities, customized commands, or sophisticated commercial applications for vertical markets. With MDL, you can:

- Develop tightly integrated interactive applications that run inside MicroStation and "look and feel" like a part of MicroStation. In fact, many features of MicroStation are implemented as MDL applications.
- Develop applications that can be easily ported between hardware platforms. MDL removes, to the extent possible, the typical concerns with the graphics subsystem, operating system, and CPU. Porting an MDL application typically involves just transferring source files to the new platform and rebuilding.
- Enable user interface-related data to be organized for efficient translation for foreign language markets.
- Have MicroStation serve as the "CAD operating system," providing the same type of support to CAD applications as the operating system does to non-graphic applications. MicroStation loads and executes MDL applications.[1]

3.1 Components of MDL

MDL supplies the following:

- A complete implementation of C. The MDL compiler compiles C source files into pseudocodes understood by MicroStation.
- A pseudocode interpreter inside MicroStation that executes MDL programs.
- A complete development environment, including the following tools:
 - ⇒ C compiler, librarian, and linker
 - ⇒ Resource compiler and librarian
 - ⇒ Source level debugger
 - ⇒ Make program
- A runtime library of over 1750 built-in functions -- functions that are part of the MicroStation executable. Built-in functions run at full compiled speed and are the same functions used internally by MicroStation's own commands. These built-in functions are essential for applications to work seamlessly with MicroStation and other applications. MDL programmers use the same tools as MicroStation developers to create commands.
- Access to built-in variables. Much of MicroStation's global data is available to MDL programs as built-in variables. Header files define the data structures of built-in variables.
- Hooks (pointers to functions) to modify MicroStation's behaviour. There are many places where applications can use MDL to modify MicroStation's behaviour. An application can designate one of its functions as a hook and cause MicroStation to call that function when certain events occur.
- A runtime environment that provides standard operating system functions such as file access, memory management, and memory protection.
- Tools for developing a platform-independent graphical user interface including dialogue boxes, pull-down menus, and palette menus. The user interface toolbox insures that the same dialogue box code will work on all platforms.
- The resource manager lets you separate your application's code and data. The Resource Manager controls data that is not actually included in the C program. The Resource Manager includes a C-like language used for defining data, a Resource Compiler for compiling definitions, and a library of MDL built-in functions used to access data contained in resource files. The type of data managed by the resource manager includes dialogue box definitions, messages, and prompts.

4. Editor functions

The most needed functions for archaeological reconstructions that were selected for implementation are the following:

- Insert a stock – In the place where map contains small circle, it is possible that a stock was originally inserted to the ground. Such places can be placed "randomly" in the space. Archaeologist working with ArcheoBuilder only will select the appropriate circles and enter height of the stocks. Editor automatically inserts the stocks to the drawing.
- Insert a palisade (see Figure 4) or a fence (see Figure 5) – In the place where map contains some slightly oblong oval, a palisade could have existed (a set of stocks). The editor automatically generates the set of stocks and places it to the place selected by archaeologist.

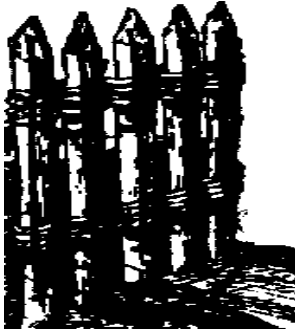


Figure 4: Palisade

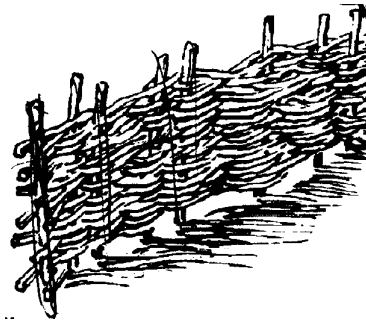


Figure 5: Fence

- Projection of wall – Only the bottom part of old buildings comes down. This occurs in the map as a platform of buildings. It is needed to project circuit wall form the platform.
- Place a roof – Archaeologist working with ArcheoBuilder enters three points, height, and the type of roof. The editor places the roof automatically. Some roof types can be seen in Figure 6 - Figure 11.

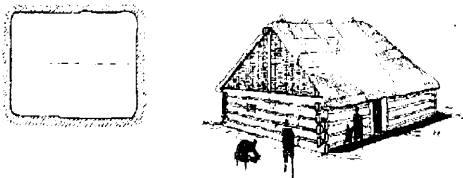


Figure 6: Loghouse



Figure 7: Loghouse

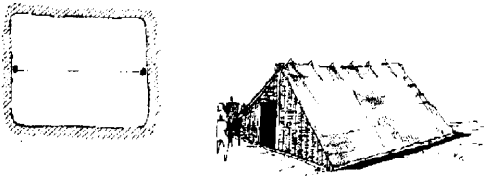


Figure 8: Loghouse



Figure 9: Loghouse

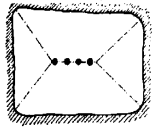


Figure 10: Loghouse



Figure 11: Loghouse

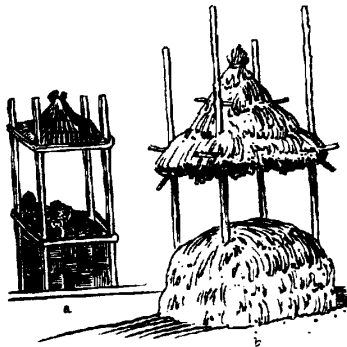
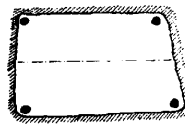


Figure 12: Hay storage

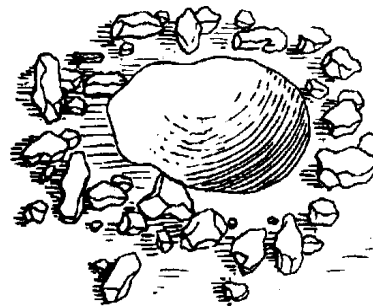


Figure 13: Grate

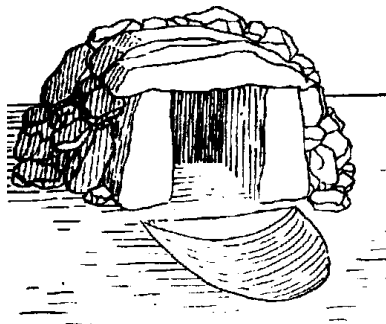


Figure 14: Stone oven

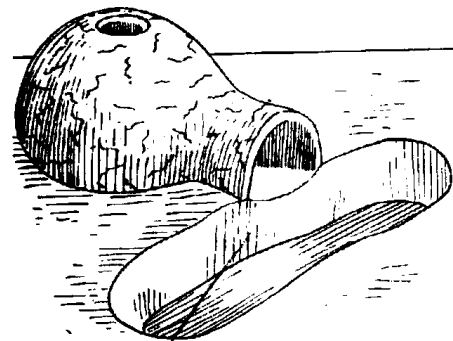


Figure 15: Loam oven

These functions were designed in co-operation with archaeologists at Philosophical faculty of Masaryk university in Brno. They believe that they can process most of the data from their excavations using the above functions. Of course, other changes and general editing of the drawings will be possible through common functions of MicroStation.

4.1 System functions

Other important features of ArcheoBuilder

- 3D drawing detection – This function is important from the view point of MicroStation itself. When 3D features of the system are to be exploited, user should explicitly ask for it at the time of creation of the drawing. Running ArcheoBuilder with 2D drawing is, of course, meaningless and executing 3D commands on 2D drawing causes errors.

- Tool palette – This function is important from the archaeologist’s view point. The palette includes also dialog boxes in which user can assign real dimensions to the objects that are being created. In fact, palette forms the ”face” of the program (see Figure 16).
- Attaching MDL application to the drawing – This feature speeds-up the work with drawings for the user who is working with one model for most of the time. A ”starting element” is included in the drawing and is executed on opening the file. In our case, the application is ArcheoBuilder.
- Finishing the application – The application could be explicitly finished by pressing appropriate button, or it is finished automatically on closing the drawing.

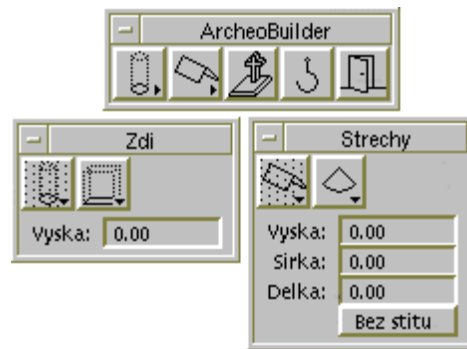


Figure 16: The ArcheoBuilder application

4.2 Example of usage

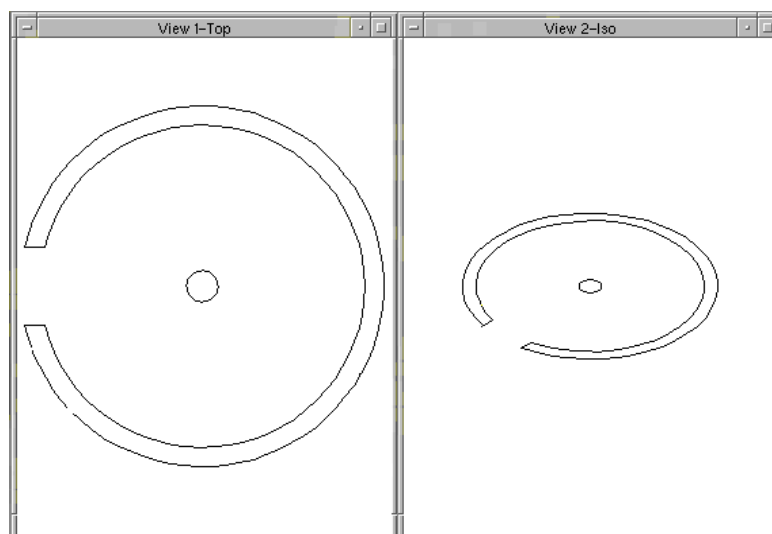


Figure 17: Map of decision

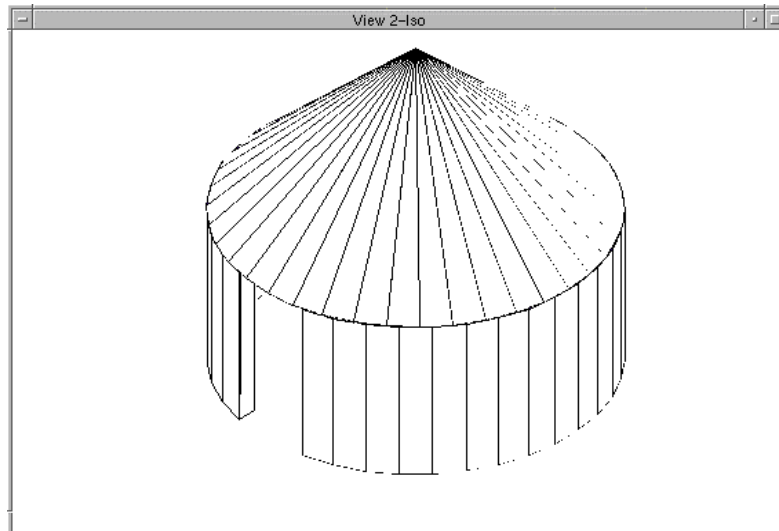


Figure 18: Reconstructed building

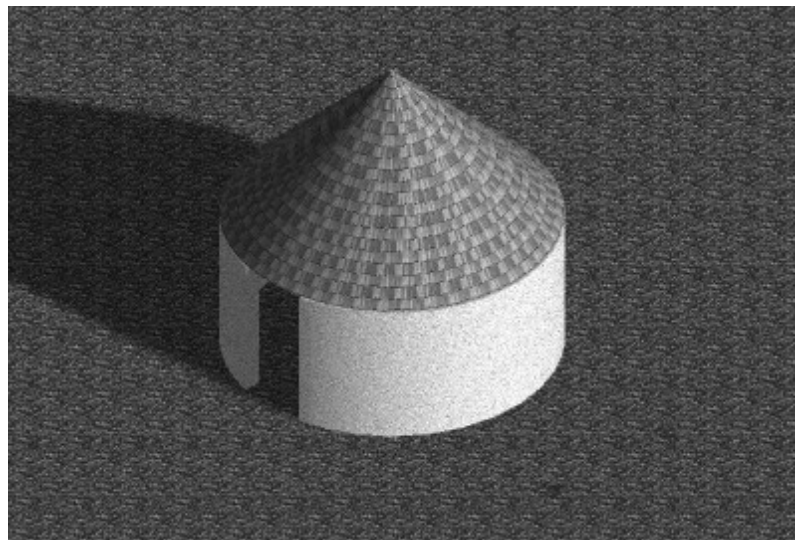


Figure 19: Rendered view

5. Conclusions

The ArcheoBuilder editor was created according to the wishes of archaeologists. Hopefully, it will be helpful for archaeologists in their work on reconstruction of buildings that existed on the Earth many centuries ago.

6. References

- [1] MicroStation PC - MDL Manual, Intergraph Corporation, Huntsville , 1991.