DEVELOPING THE TERMITE PLUG-IN

Abstracting operations to link 5-axis CNC routers with parametric CAAD tools

NICHOLAS WILLIAMS and DHARMAN GERSCH
RMIT University, Melbourne, Australia
{nicholas.williams, dharman.gersch}@rmit.edu.au

Abstract. Since the turn of the millennium, architects and designers have used greater access to Computer Aided Manufacturing (CAM) machines to explore links between design and fabrication. This trend is recently manifested in plug-ins for CAD software packages, which enable designers to program industrial robots and additive manufacturing machines. However, amongst the array of contemporary tools, few connect CAD packages to commercial 5-axis routers and, as a result, designers are forced to use complex CAM software to operate these machines with limited exploration of the interface with design. This paper reports on the development of a CAD plug-in for driving such routers and targeted at designers. It discusses key aspects in the conception of the software libraries for an alpha release of the tool, a plug-in for McNeel Grasshopper named Termite. Primary considerations for the development team include the areas and extent of flexibility offered in order to enable non-expert users of such machines to use them to in an effective and efficient manner. Key elements of the tools are discussed, including the definition of machining tools, the creation of generic toolpaths and the subsequent writing machine-code files. A set of example pieces are presented to demonstrate the proposed approach for flank-milling, patterning and connecting timber components at a furniture scale. These are compared to plug-ins for industrial robot with similar technical knowledge and experience amongst the target audience.

Keywords. Digital fabrication; parametric design; architectural prototypes; digital material.
1. Introduction

Gaps between design and fabrication are being shrunk as technology for CNC fabrication becomes increasingly cheap and ubiquitous (The Economist, 2012). Amongst the hype around this technological expansion are a number of assumptions about the skill required to operate machines in a manner that is effective, efficient and safe. At an extreme, it is assumed that Computer Numeric Control (CNC) machines remove all requirements for skilled operators, a conception which has been historically shown to be naïve and costly (Moe, 2010). The reality is that specific user groups have specific demands in flexibility of operation and accuracy of finish. This paper presents the conception and alpha version of Termite, a plug-in targeted at designers for the operation of panel routing machines. The development of this software demands the careful consideration of the target audience, their skill levels and the functionality they will require. Designers are not commonly experts in details of CNC machine operation and are not engaged to finesse details or optimise process for high volume efficiencies. It is therefore important to provide a flexible but simple interface between design and machining. Where designers are producing material artefacts as part of an iterative design process, these are often to relatively low dimensional tolerances. For example in the construction industry +/- 1mm is more accurate than conventional building trades and presents emerging challenges and opportunities (Sheil, 2013).

2. Background

2.1. TARGET AUDIENCE AND CHALLENGES WITH CAM PROGRAMS

Rather than being closely aligned with the tasks of contemporary designer, CAM has conventionally been a separate task, undertaken using software packages operated by specialist users. This reflects a deep set of knowledge required for using CNC and other fabrication machines. However, contemporary interest of the so-called maker movement and businesses promoting amateur users are challenging this divide.

Some tasks such as the 3-axis operations of machines like laser-cutters are generally quickly learnt. However, for machines with more degrees of freedom, the complexity of programming and operation increases steeply. To drive machines with 5 and more axes, typically a standalone CAM program is used, a common example in industry being DelCam PowerMill. This program exposes the user to over 17 dialogue windows for every toolpath, with numerous parameters and settings within each, allowing accurate and opti-
mised programming for a broad number of machines. The complexity of this software and the detailed inputs required is such that regular use is necessary to efficiently operate a machine.

2.2. CAM PLUG-INS FOR DESIGNERS USING ROBOTS

Recently, software plug-ins to CAD platforms have become available which offer the programming of industrial robot arms. These have claimed attention by providing a relatively simple interface between design and fabrication. The simple interface provides access to basic functionality of a machine and enables rapid feedback iterations between digital design versions and material prototypes, without confusing the process with too many details related to machining. Amongst a number of tools, two plug-ins for parametric design platform Grasshopper for McNeel Rhinoceros - KUKA|prc and HAL – are exemplars. Users with relatively little experience in machine programming, can generate files for Kuka, ABB and Universal robots.

Industrial robot arms have been described as “universal fabrication machines” (Bonwetsch, 2013) for their flexibility of tool and geometry. The plug-ins mentioned here provide universal functionality – placing a tool point in a specified position. Much of industry, however, utilises Computer Numeric Controlled (CNC) machines which are designed for specific tasks. These machines are more efficient than universal robots, offering higher accuracy, as well as more consistent and faster cutting speeds. CAD plug-ins similar to those for robots, however, have not been developed for these machines, and so novice users have little opportunity to use them.

3. Key aspects of Termite

Termite, currently built as a plug-in for the parametric modelling platform Grasshopper within McNeel Rhinoceros seeks to be an easy extension to a design workflow. Panel routing machines have a very broad set of input parameters. As a tool for non-expert users, the plug-in schema presented here provides only limited options to the user in order to minimise problems, increase speed and ease of use.

Termite development has initially focussed on aspects of panel machining that cannot be achieved with 3-axis machines such as flank milling, compound saw cuts and angled drilling. The low computational requirements of these operations also make them well suited to near real-time manipulation in a parametric design environment, a critical aspect for the plug-in’s functionality. ‘Surfacing’ which is more computationally intensive presents specific challenges and is covered briefly by parallel finishing and spiral finishing components.
The process to create a machine code output file is broken into five steps – tool definition, geometry definition, cutting operation definition, machine definition and file generation (Figure 1). Outside of these is a ‘parser’ component, which generates the machine preview directly from the output code. This last component allows for existing machine files can be read into grasshopper as geometry, which provides a convenient way of checking files, without needing navigate someone else’s parametric definition. These key components can be modified and deconstructed in greater detail with additional components.

Figure 1. A basic Termite definition consisting of: Tool, Toolpath, Machine, Core and Parser. Note that not all options require user input. Default toolpath feeds and speeds are extracted from tool properties, stock size is defined from input geometry bounding box, and options are set to sensible default values for each toolpath type.

3.1. ‘CORE’ COMPONENT

The core component: Allows key folders to be referenced; ensures the correct document units system; post-processes toolpaths into machine code, and contains all the code classes, which are centralised for ease of maintenance.

The source code to Termite is exposed within each component, making it possible for others to extend and modify functionality. Users can create their own scripted components using termite datatypes if a specific function doesn't exist. All Termite datatypes can also be deconstructed into generic and native Rhinoceros types including lines, points, vectors, numbers, strings and booleans. This approach contrasts with the other packages mentioned above.

3.2. ‘TOOL’ CLASS AND COMPONENTS

Tools can be defined by a user or alternatively generated directly from tool data file stored on the machine. This latter approach enables easy mainte-
nance of keeping tool library in Termite, updated by simply locating a copy of the tool data.asc file from a machine backup.

This machine-managed tool definition contains limits for the tool feeds, speeds, depths, useable length, whether it can cut with it's end, side or both. For example, if a saw blade manufacturer requires that the blade rpm not exceed 4500 and if a termite tool path spindle speed exceeds this limit, it will throw an error. A similar error message is raised if a user tries to use a drill bit rather than a milling bit for a milling operation.

This difference between tool types is a common mistake among novice machine users. Restrictions in the software are essential where the novice programmer is very unlikely to be aware of any restrictions on the tools or machine. This enables those with design skill, but not necessarily CAM knowledge to integrate CAM into their workflows.

3.3. ‘TOOLPATH’ CLASS AND COMPONENTS

Toolpaths in Termite are defined as lists of a generic point and vector format (x,y,z,i,j,k), accompanied by a feedrate. This definition is not machine specific and can readily be fed to different machine post processors, as well as some simple machines and CAM softwares. For example Termite toolpaths could be used to create robot toolpaths in HAL or Kuka|prc, allows our plugin to work alongside other CAM plugins.

Many of the possible attributes of a toolpath are not exposed on the toolpath components. A toolpath settings component is available to provide more detailed controls where necessary, however the default values are intended to cover most scenarios.

Lead-ins, the process of a tool entering a material stock, are an essential part of toolpaths. These are always created as a default of defining a toolpath. There are three main strategies: plunge – directly along a tool axis; ramp – in an oblique direction towards the first point; and spiral – movement in a circular, ramping motion. Lead-ins typically occur at a lower speed. Further detail of a toolpath is needed to accurately and safely define a strategy for cutting ruled surfaces (Figure 2). Many common methods used to generate these surfaces are unsuitable for milling as a tool is designed to cut in a specific direction, typically either parallel to a trajectory or tangential to the direction of cut.
3.4. ‘MACHINE’ COMPONENTS

The machine component defines the capabilities of the machine, such as its bed size, spindle limits, and additional command code formats if available such as chip blower, extractor hood position and clamping/vacuum controls. These ancillary functions can be added as inputs to an extensible component. The machine component also contains an instance of the ‘post’ class, which defines the format specification of machine instructions, as well as headers, footers and tool change routines all of this information is needed to translate the geometry into machine code. This is the point at which close coordination of machine specific syntax is required to interact with more generic toolpath definitions.

3.5. ‘PARSER’ COMPONENTS

The toolpath preview is generated directly from the post-processed output file by means of a parser component. This component uses the format specification of the writer component to read the toolpath back into Grasshopper. This helps ensure that the writing component has written in the correct format. Essentially this allows students to create machine files on their own using termite, then staff to check the output by using only parser and preview components. The parser also retrieves the tool from the tool database.

4. Example case studies using alpha software tool

The alpha release of Termite plugin includes one fully developed machine library for a Biesse Rover 5-axis machine. A series of common tasks are presented as examples for the functionality of the library. The software is cur-
DEVELOPING THE TERMITÉ PLUG-IN

rently being used by workshop staff who operate this machine. This is an audience with range of experience on the machine, some having used it previously and others new to CNC operations. The examples below are common tasks which these staff must undertake.

4.1. FLANK MILLING

A common prototyping challenge is fabricating flat panels with many uniquely mitred edges, either as planar faces or ruled surfaces (Figure 3). Termite can streamline this process by connecting data from the design model to machine code. The existing on-machine solution proved to be laborious, requiring detailed operator input and unwieldy conversions between file formats. This made the prospect of mass customisation both labour intensive and prone to human error. This ultimately stifled the design process by forcing iterations to be locked down early, increasing cost of change (Davis, 2011).

To use the flank-milling component, polylines representing both the top and bottom outlines (tool tip and tool guide) of the panels are selected. Each discretized tool-tip point is paired with a relative guide segment. These segments are then extended and adjusted to ensure the tool cuts perpendicular to the direction of travel.

Lead-ins and rapid paths linking cutting operations are also added. The core converts the tool positions to the machine NC format, writing the file and previewing the tool position at each line of code. This can be scrubbed back and forth through the use of a slider, enabling a user to make a visual check for errors, before loading the file onto the machine. Flat panel collision checking allows us to visualize if the tool shank or tool holder intersect with the stock (Figure 4). In the curved ‘spoon’ example a preview mesh of the form is generated from toolpaths. This representation allows the designer...
to know if a design change causes a collision or a tool limit to be breached in-situ.

![Figure 4. Tool preview and collisions with panel](image)

### 4.2. DRILLING AND COMPOUND CUTS

More simple examples are the linear motions which are required for drilling operations and compound cutting (Figure 5). These can be defined by simple, linear set-out geometry. Common problems amongst operators have been attempts to create drilling operations from non-linear curves and compound cuts from non-planar curves. These problems are avoided by the components ignoring incompatible geometry, and notifying the user which geometry indices have been skipped and for which reason.

![Figure 5. Angled drilling and compound saw cut examples](image)

### 4.3. SURFACE MILLING

A given mesh or B-Rep form can treated as a form for milling through a series of basic strategies. For such an operation, the toolpath component can be specified to generate either a spiral, parallel or algorithmically generated.
Accurate surfacing presents challenges through managing tooltip compensation with reasonable calculation time.

5. Alpha release as part of the Rhino/Grasshopper ecosystem

With such a focus to this plugin, a number of key contrasts in functionality can be seen between Termite and other CAM plug-ins for Rhinoceros (Table 1). KUKA\textit{\slash}prc and HAL are general purpose platforms for the control of robot arms. As these are highly generic machines the software does not impose limitations on toolpath strategies allowing the rotation and travel of a head in a freer set of movements. As discussed previously, this freedom is not desirable to panel routers and so toolpath movement becomes part of a network of constraints in a parametric design schema. The time and effort required for straightforward tasks can be further improved by the development of more, common default settings to components.

Table 1. Comparison of functionality and features of other CAM plugins for Rhinoceros.

<table>
<thead>
<tr>
<th></th>
<th>RhinoCam</th>
<th>KUKA\textit{\slash}prc</th>
<th>HAL</th>
<th>Woodpecker</th>
<th>Termite</th>
</tr>
</thead>
<tbody>
<tr>
<td>5-Axis</td>
<td>●</td>
<td>●</td>
<td>●</td>
<td>●</td>
<td>●</td>
</tr>
<tr>
<td>Parametric</td>
<td>●</td>
<td>●</td>
<td>●</td>
<td>●</td>
<td>●</td>
</tr>
<tr>
<td>Fank milling</td>
<td>●</td>
<td></td>
<td>●</td>
<td>●</td>
<td>●</td>
</tr>
<tr>
<td>Drilling</td>
<td>●</td>
<td></td>
<td>●</td>
<td>●</td>
<td>●</td>
</tr>
<tr>
<td>Saw cuts</td>
<td>●</td>
<td></td>
<td>●</td>
<td>●</td>
<td>●</td>
</tr>
<tr>
<td>Machines</td>
<td>&gt;300</td>
<td>&lt;20</td>
<td>&lt;100</td>
<td>external</td>
<td>1</td>
</tr>
<tr>
<td>Tool limits</td>
<td>●</td>
<td></td>
<td></td>
<td>●</td>
<td>●</td>
</tr>
<tr>
<td>Machine code</td>
<td>●</td>
<td>●</td>
<td>●</td>
<td>●</td>
<td>●</td>
</tr>
</tbody>
</table>

As a contrasting package, RhinoCam provides extensive functionality typical to a CAM software package. This allows an experienced user to specify machining details, however, at the cost of more complex user interface, poorly suited to designers. Furthermore, as a standalone CAM plugin, it is disconnected from the parametric design schema and therefore less easily influence design decisions. Early feedback from users is that Termite has great benefits in ease of processing common tasks.

6. Conclusion and next steps

This paper has reported on the development of a plugin to connect parametric design with panel routers. This plugin is aimed to fill a gap between CAM packages currently used for programming these machines, and the parametric schemas developed for design. A closer coupling of these two el-
lements provides the potential for the fabrication constraints to inform conceptual design work.

Further development is necessary for a broader release of the plugin. This includes the development of libraries for other machines, firstly 3-axis panel routers to complement the existing 5-axis capabilities. The toolpath definition will include redundant information of a vertical tool angle, however, this is considered a small expense when considered in the broader context of the plugin’s flexibility.

Beyond this, ease of programming is a priority and workflows for generating toolpaths will be improved through substantiated feedback from users. Assumptions regarding common mistakes and preferences for default settings will be reviewed. Beyond this, of immediate interest is improvement to surface milling functionality. The toolpaths here can require a large amount of detail and slow, iterative adjustment by a user in order to achieve desirable results.

Acknowledgements

The authors are indebted to the staff in the prototyping workshops in the School of Architecture and Design at RMIT University for ready access to facilities.

References


Sheil, B. (ed.): 2012, Manufacturing the bespoke: making and prototyping architecture. Chichester, U.K.: John Wiley and Sons Ltd.
