5th Australasian Congress on Applied Mechanics, ACAM 2007
10-12 December 2007, Brisbane, Australia

An Intelligent System for Rule Based Automated Routing

Christian Van der Velden[1], Cees Bil[1], Xinghuo Yu[2], Adrian Smith[3]

[1]School of Aerospace Mechanical and Manufacturing Engineering, RMIT University, Melbourne, Australia
[2]School of Electrical and Computer Engineering, RMIT University, Melbourne, Australia
[3]451°CONSULTING, Australia

Abstract: The automation of engineering processes through Knowledge-Based Engineering methods and technologies can provide significant savings in project scheduling and cost, increasing competitiveness in a changing aerospace market. In this paper we present outcomes of a research project aimed at improving engineering automation capability through development of a system for automatic rule based path-finding for the complex engineering task of aircraft electrical harness and pipe routing.

Keywords: Computer Aided Design (CAD), Design Automation, Intelligent Systems, Knowledge-Based Engineering (KBE), Routing.

1 Introduction

The automation of engineering processes through Knowledge-Based Engineering (KBE) methods and technologies can provide significant savings in project scheduling and cost. In this paper we present outcomes of a research project aimed at improving engineering automation capability through development of a system for automatic rule based path-finding for aircraft electrical harness and pipe routing.

The current aerospace engineering environment is very different from that of two and three decades ago which saw high activity in the military sector in the years surrounding the cold war, and in the civil sector with high levels of tourism. A significant reduction in the number of major civil and military aerospace programs has led to highly cyclic work patterns of high then low levels of activity, often in response to the current economical and political climate. Accordingly, for engineering companies to remain competitive in this dynamic environment, it is vital to exploit opportunities, and respond to customer needs rapidly to gain a share of engineering workload resulting from major projects.

The level of knowledge in engineering companies is built up over time through development of in-house best practices and handbooks, and experience of personnel. This engineering knowledge is a valuable asset which is critical in successfully bidding for work, and ensuring project success. The ability to harness this knowledge effectively through deployment of rapidly developed software applications for automating engineering processes can provide solutions for problems faced on the engineering floor, and provide a competitive edge in the global market. Such applications can reduce the number of man-hours required for specific tasks which are often the cause of bottlenecks in the product development process. In general, applications developed for automating processes are tailored to specific problems and accordingly, are limited in scope. Typical processes which lend themselves well to automation generally exhibit one or more of the following characteristics:

- Low level, repetitive, and/or highly manual tasks
- Integration of tools and datasets (e.g. CAD / CAE / CAM)
- Automated documenting and report generation
- Simplification and/or standardisation of more complex processes

It is important to recognise that not all processes are suitable for engineering automation. Some tasks, especially those requiring tacit human judgement, will always require some user input, and the effort required to automate such processes often does not outweigh the benefit gained by automated capability. Another consideration is that automating unsuitable processes can reduce the enjoyable parts of an engineer's job, instead focusing their work on mundane button pressing.

This paper details the process of automating the layout design task for electrical aircraft electrical harnesses and hydraulic pipes, and presents results of a test case demonstrating system effectiveness.

2 Problem Background

The increasing complexity of aircraft electrical systems has led to an increase in the number and size of electrical harnesses required to connect subsystems and equipment throughout the airframe. Electrical harness routing is a complex and largely manual task with numerous governing rules and best practices. Wiring looms typically comprise hundreds of harnesses all of which are manually routed, with engineers using personal knowledge and experience of the problem domain.

Also, adding to the problem size and complexity, subsystem design (including the wiring system) is often conducted in parallel with principle structural design in large scale projects. Therefore changes in structure and subsystem layout occurring over the development phase can impact wiring looms, requiring time-consuming and expensive rework. This can lead to lengthy delays in the aircraft development, as has been seen recently with the Airbus A380 wiring systems [1].

Major aerospace companies often have proprietary standards and practices for harness routing, which can vary between aircraft development programs depending on requirements. The generic process for harness routing involves manually creating a set of points in the CAD structural model from which the harness will be clamped to the main structure. Following this, the spine of the harness is passed through these points; ensuring sufficient clearance is given from structure, particular subsystems, certain types of harnesses, moving parts, and areas of high heat. The process can be largely trial and error, and often the only way to determine whether sufficient clearance has been allowed, is to make manual measurements in the CAD model which can be time consuming. These characteristics make the routing task a prime candidate for process automation.

3 Approach

The routing problem is encountered in numerous fields including electronics (including microprocessors and printed circuit boards), navigation systems, and Artificial Intelligence (AI). Many algorithms have been developed in these fields, but few, if any, have been tailored and applied to the aerospace domain.

Algorithms for microprocessor design employ powerful path-finding algorithms including maze routers, channel and switchbox routers, and line routers [2]. These algorithms are driven by technology improvements in component manufacture including the ever-reducing size of wires and number of two dimensional layers available for routing layout. Commercial software packages for multilayered printed circuit board design are also available, employing similar techniques.

Improving AI in computer games is another area which continues to drive intelligent path-finding development. The way in which non-player characters move and react in the game environment largely determines the realism of the gaming experience. To this end, numerous algorithms have been developed, incorporating various behavioural techniques including shortest path, stealthy movement for avoiding detection, cautious movement using cover to avoid damage, etc.

Developments in these fields have provided an excellent base of knowledge which can be utilised for engineering automation of aircraft electrical harnesses and pipe layout. The system developed incorporates path-finding techniques from microprocessor routing and game AI domains, together with knowledge modelling techniques for capturing design rules, enabling the model to be constrained in such a way as to produce a path which accurately satisfies requirements, minimising the amount of manual work required. The system layout is shown below in Figure 1.

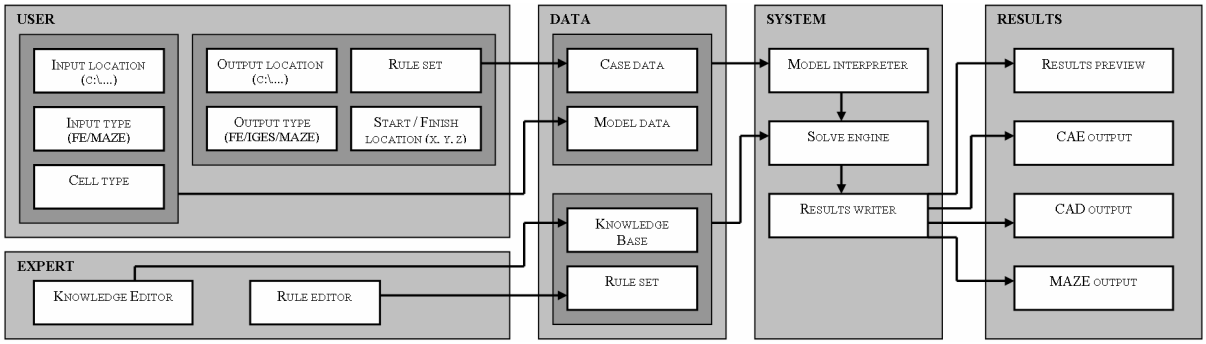

Figure 1: Automated router system structure

4 Test Model

System functionality will be described with reference to a test case consisting of a simplified model of an internal weapon storage area, typical of new generation fighter aircraft such as F-22 Raptor and F-35 Lightning II. Weapon bays are complicated and densely populated assemblies consisting of complex metallic structure, payload envelope, and various subsystems and equipment with hundreds of interconnecting electrical harnesses and pipes. Figure 2 shows one of the weapon bays of the F-35 Lightning II [3]. The photo shows a complex weave of harnesses and pipes throughout the length of the bay. As discussed above, the route for each harness is determined manually on CAD workstations. It does not take a lot of imagination to see that design of such a complicated loom is a very time consuming and resource intensive task.

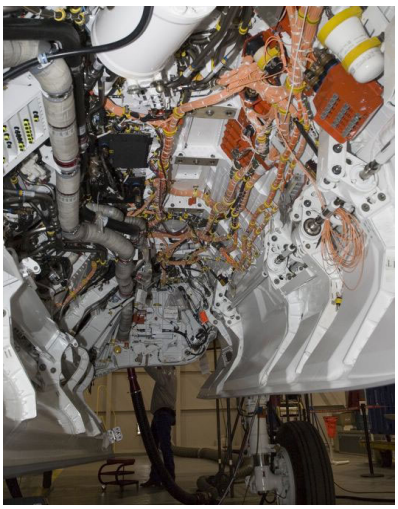

Figure 2: Weapons bay of F-35 Lightning II. [3].

The test case for this software is a simplified version of the assembly shown in Figure 2, based on observations of the CAD model of the F-35 weapon bay structure and subsystems. The test geometry consisting of principle structure (excluding bay doors), arbitrary subsystems, and sample payload was modelled using CAD software and is shown in Figure 3.

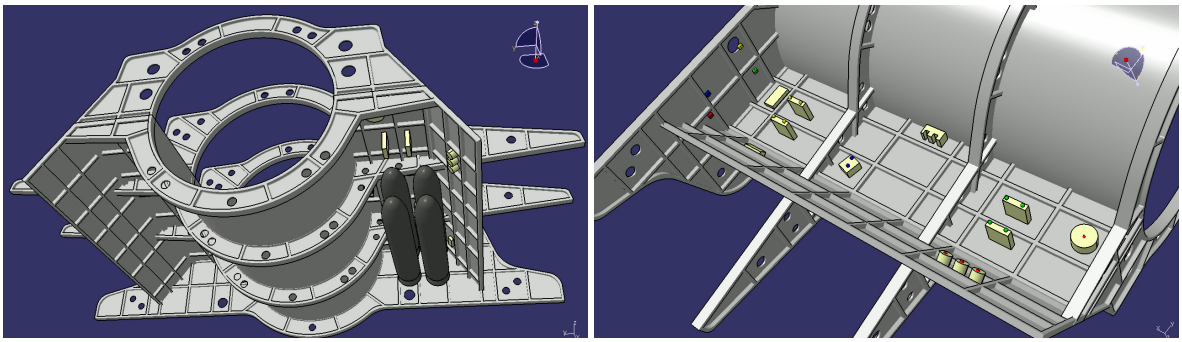

Figure 3: Test case geometry

Geometry is represented internally as a discrete, grid-based maze object with each grid node characterised by an integer address (in Cartesian coordinates) and node type (e.g. principle structure, subsystem, payload, pipe/cable type, searchable space, etc.). Categorisation of nodes within the solver is significant as it allows various obstacle types to be treated independently by rules implemented in the path-finding algorithm, such as following certain types and avoiding others.

5 Knowledge Modelling

The system supports routing of path types from numerous domains including electrical wiring harnesses, hydraulic and pneumatic pipes, and fuel lines. Engineering knowledge of the particular routing domain considered must be represented accurately in the system for results to be valid. Effective knowledge capture processes are therefore important to obtain an accurate and complete coverage of all rules applicable.

In cases where all rules are explicitly documented, modelling knowledge can be a relatively straight forward matter of quantifying and categorising rules. Engineering processes requiring tacit knowledge stored within experienced engineers can present difficulties in knowledge capture, and remains one of the main challenges in KBE.

The automated routing system includes a rule editor for modelling domain rules. The editor consists of a simple form containing a set of controls for defining rule types, conditions for validity, area of influence, and action to be taken. Families of rules for different routing domains and path types are stored in separate libraries in Comma Separated Variable (CSV) format.

Rules supported by the system currently include: bend radius, path profile, attract and repel rules causing paths to favour certain obstacle types and avoid others, turn penalty rules restricting amount of bends in the path, restrictions on movement (single, two and three dimensional movement), and clamping rules. Implementation of these rules within the solver is discussed below in section 6.

Solver settings, import and export options and hard-coded rules are defined within a database termed the "knowledge base" which can be customised for different routing applications. These are accessed through a knowledge editor.

6 Path Discovery

The algorithm implemented in the automated routing system extends the popular A* search algorithm used in shortest path problems in game AI [4]. In the basic A* algorithm, path movement is determined by evaluating a cost function for each node interrogated (Equation 1). Node cost, ***f(n)***, is equal to the sum of the shortest distance from the source node, ***g(n)***, and the estimated cost to the goal using a heuristic function, ***h(n)***. At each iteration of the search, the lowest cost node is selected as the new node to expand. Provided the heuristic which calculates the remaining distance does not overestimate the distance to the target, the algorithm is both optimal and complete [4]. Example heuristic functions are discussed in [4].

$$f(n) = g(n) + h(n) \tag{1}$$

Tailoring the algorithm to the complex rule-based domain of electrical harness design, the cost function is extended to include additional terms representing rules in domain libraries (Equation 2). The ***g(n)*** and ***h(n)*** terms are determined in the same way as normal A*. The ***i(n)*** term modifies total node cost, depending on proximity to given obstacle types, causing the path to favour obstacles of a particular type and avoid others. The magnitude of the ***i(n)*** term is determined by performing a local search within a given radius for each node interrogated in the main search. A separate ***i(n)*** term is added for the number of proximity-type rules in the library, ***k***.

$$f(n) = g(n) + h(n) + \sum_{N=1}^{k} i_N(n) + T(n) + ... \tag{2}$$

For example, a rule may specify that harnesses must be routed close to principle structure for clamping purposes. Such a rule would reduce the cost for nodes falling close to structural obstacles, resulting in a lower cost solution closer to structure. Other rules may state that harnesses must have a given clearance from heat zones, moving parts and cables with particular electromagnetic sensitivities. Such rules increase node cost for nodes near these obstacle types, resulting in a lower cost solution away from these areas.

A turn penalty term, ***T(n)***, is also introduced for models with complex geometry, encouraging the path to minimise unnecessary deflections. The contribution of the term is determined by the turn deflection angle. Mild turns are penalised less than harsh or backwards turns.

For the test case, 16 paths were routed, separated into four categories with four harnesses in each. The rule library implemented included a rule for harnesses to follow structure as closely as possible, minimising the length of unsupported runs for clamping purposes. Rules specifying each harness category to follow previously routed paths of the same type were also implemented.

7 Results Output

The system delivers paths resulting from the automated path-finding process in a number of ways. Firstly, a simple three view path diagram referenced against input geometry is given in the software application itself. This gives the user a quick indication of the quality of the routing job in terms of

whether paths were placed in the desired region of the solution space. For example, if a model features a number of cut-outs and lightening holes which are not properly constrained, the path may follow the outside of the structure rather than the inside. The simple preview can reduce analysis time for results which clearly require refinement. Several statistics relating to path quality are also displayed, including path length, number of turns, solution time, and number of nodes searched.

Secondly resultant path spines are output as CAD-readable wireframe IGES models consisting of straight segments connected with a user defined bend radius. Path models can be imported and integrated into the existing CAD geometry assembly, and detail added as necessary for a complete digital representation of the routed part using CAD software tools (e.g. thickness, detail to terminals, etc.). The output for the test case described above is given below in Figure 4. The results clearly indicate successful implementation of the rule set including the rule for following structure and previously routed cable types.

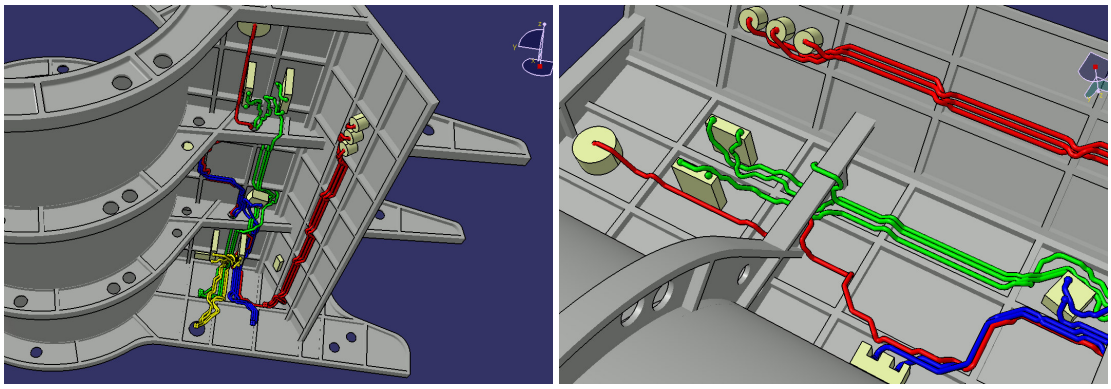

Figure 4: CAD output

The third output method is a Finite Element (FE) model comprising several component layers including input obstacles, routed paths and knowledge components describing the automated routing process. The purpose of this output method is to assist users in understanding results and implementation of rules by the solver. Obstacles encountered in the search space are included for referencing rule implementation against input geometry. They are represented by hexahedral elements colour coded according to category (e.g. primary structure, subsystems, cable category, etc.). Routed path spines are represented in wireframe using bar elements. Additional layers represent rules and search characteristics used in determining path placement, communicating design justification to the user. These layers include 3D maps showing locations where individual rules were implemented in the search space, allowing users to determine regions where particular rules are significant for any manual touching up required. A map of all nodes interrogated by the algorithm is also given, allowing users to determine whether the algorithm is considering the correct area in the search space, or whether additional constraints are necessary. The FE output for the test case is given in Figure 5.

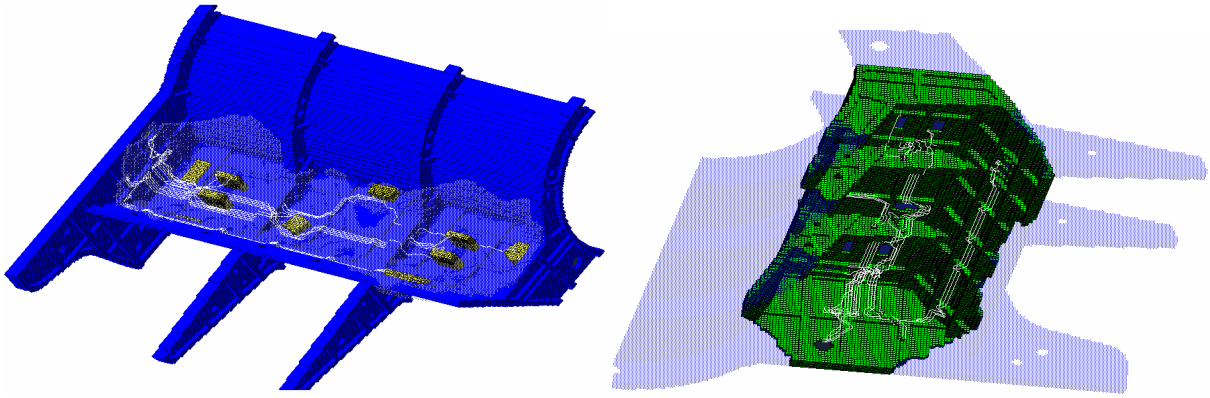

Figure 5: FE output

Various components in the solution can be activated independently to view interaction of various rules using visualisation options available in the third party FE pre-processor used for viewing results. The

two images above highlight different characteristics of the solution. The left image shows all nodes interrogated by the solver, referenced against input geometry. The image on the right shows all points on the reference geometry where the rule specifying paths to follow structure was implemented. As expected, the majority of nodes searched were in the vicinity of points where this rule was implemented, as it was a lower cost solution for points close to primary structure. Similar outputs are available for points where the various path-attract rules were implemented.

After a number of test runs of the system with various rule combinations were conducted, it was found that quality of resultant paths is closely coupled with the weight factor applied to individual rules. Thus a process for tuning the rule library for individual models is required to produce optimal results, which can be a time consuming process. Despite this, the system works very well as a proof of concept application. The solution time is sufficiently small that a number of solutions can be generated for various combinations of rules and weightings in a relatively short time, compared to manual routing practices, allowing a large number of results to be assessed for suitability relatively quickly.

Further improvements to the system would focus on implementation of an optimisation process for tuning rule weights, leading to higher quality results without the manual trial and error process for applying various rule combinations. Such work would also require development of more effective methods for quantitatively assessing path quality.

8 Conclusion

The intelligent routing environment outlined in this paper successfully implements Knowledge-Based methods and techniques together with path-finding principles from other routing domains, to produce routed paths satisfying user defined design rules and constraints. Knowledge and rule editors simplify the knowledge capture process, extending capability to domain experts to implement new routing methods and rules for application to new domains (e.g. water pipes, air conditioning ducts, etc.). The benefits of using such a tool in industry are clear with solution times of a few minutes per path for detailed, high resolution models (depending on the number of rules applied, and the degree to which the model is constrained), in comparison to the manual process, which can take two to three days.

Two main implementations of automated engineering solutions are commonly used in industry. The first involves a formally identified task with well defined requirements, built by a team of developers. The business case for such solutions must be clear, i.e. prove a positive return on investment (number of hours developing the solution and completing the task automatically, versus the number hours required to perform the task completely manually).

The second implementation is generally undertaken on an individual basis by engineers, not necessarily proficient in programming, to automate simple tasks. Examples include spreadsheets, macros, and small applets that reduce the amount of manual, repetitive work required for performing calculations, document generation and manipulation of large amounts of data between analysis and design tools. These types of applications can often benefit many people, but mechanisms for sharing applications, knowledge and skills are often lacking in industry.

Significant productivity improvements can be made by establishing a culture of implementing Knowledge-Based principles in engineering design and analysis. This can be facilitated by introducing engineers to a structured process for automating engineering tasks, providing appropriate levels of training in development tools, and providing awareness of existing infrastructure such as generic code libraries for performing common tasks. Ultimately, project success requires rapid mobilisation of resources to respond quickly to problems faced by both customers, and internally on the engineering floor. Implementation of Knowledge-Based methods and technologies can provide this capability, delivering time and cost savings, and improve competitiveness in the global engineering market.

References

[1] Airbus Website. [Online October 2006] URL: http://www.airbus.com/en/presscentre/press releases/pressreleases_items/06_10_03_a380_delays_company_restructuring_plan.html

[2] Groeneveld, P. “Electronic Design Automation, Part 1”. Technische Universiteit Eindhoven, The Netherlands, 2005.

[3] F-35 Joint Strike Fighter Official homepage. [Online October 2006]. URL: http://www.jsf.mil

[4] Russel, R., Norvig, P. “Artificial Intelligence A Modern Approach”. Second Edition. Prentice Hall, United States, 2003.