Innovations Syst Softw Eng (2007) 3:117–128
DOI 10.1007/s11334-007-0021-4

ORIGINAL PAPER

An intelligent system for automatic layout routing in aerospace design

Christian Van der Velden · Cees Bil · Xinghuo Yu · Adrian Smith

Received: 30 May 2006 / Accepted: 1 February 2007 / Published online: 8 May 2007

Abstract This paper discusses the development of an intelligent routing system for automating design of electrical wiring harnesses and pipes in aircraft. The system employs knowledge based engineering (KBE) methods and technologies for capturing and implementing rules and engineering knowledge relating to the routing process. The system reads a mesh of three dimensional structure and obstacles falling within a given search space and connects source and target terminals satisfying a knowledge base of design rules and best practices. Routed paths are output as computer aided design (CAD) readable geometry, and a finite element (FE) mesh consisting of geometry, routed paths and a knowledge layer providing detail of the rules and knowledge implemented in the process. Use of this intelligent routing system provides structure to the routing design process and has potential to deliver significant savings in time and cost.

C. Van der Velden (✉) · C. Bil
School of Aerospace Mechanical and Manufacturing Engineering, RMIT University, Melbourne, VIC, Australia
e-mail: S2108846@student.rmit.edu.au
URL: http://www.rmit.edu.au

C. Bil
e-mail: cees.bil@rmit.edu.au
URL: http://www.rmit.edu.au

X. Yu
School of Electrical and Computer Engineering, RMIT University, Melbourne, VIC, Australia
e-mail: x.yu@rmit.edu.au
URL: http://www.eng.rmit.edu.au/∼xinghuo

A. Smith
GKN Aerospace Engineering Services Pty Ltd, Melbourne, VIC, Australia
e-mail: adrian.smith@au.gknaerospace.com
URL: http://www.ukes.aerospace.gknplc.com/aesinternet/australia.html

Keywords Intelligent systems · Knowledge based engineering · Routing · Computer aided design (CAD) · Design automation

1 Introduction

The aerospace engineering environment is becoming increasingly connected with new technologies enabling concurrent engineering to be common practice within engineering companies around the world. Complex programs are often conducted 24 h a day, spanning multiple sites and time zones with models and data passed on several times a day. Design, analysis and manufacturing technologies including CAD software for generating geometry, computer aided engineering (CAE) software for analysis, and computer aided manufacturing (CAM) software for automated manufacturing are evolving with advances in computer hardware and software. These technological improvements have shifted the distribution of workload between design and analysis engineers in the aerospace engineering environment from about 4:1 in the 1980s, to about 1:2 in today [26]. However, despite the technologies enabling these practices, limited communication between teams working in different locations can lead to difficulties in understanding the rationale behind particular design decisions made by engineers working offsite. New challenges are also faced in effectively managing ever increasing volumes of data and engineering knowledge.

Knowledge based engineering (KBE) is a branch of engineering concerned with capturing rules and knowledge relating to processes in the engineering of a product or system, and implementing the knowledge in software systems called knowledge based systems (KBSs) that emulate human decision making and automate processes. Use of KBE methodologies and technologies provides a structured approach to

design [4]. Output applications from KBE processes facilitate design automation, verification, and integration.

Traditionally, KBE has been closely coupled with geometric modelling in CAD environments, allowing geometry to be rapidly created using sets of rules describing steps in the process. However, competitive advantage can be gained by extending KBS capability to reflect the current environment by linking and integrating design and analysis processes by building links between the tools used to carry out the work.

This paper outlines the current progress in a research project involving development of an intelligent routing system which will be used for automating design of electrical wiring harnesses and hydraulic and pneumatic piping among other routing applications. The software outputs a CAD model and finite element (FE) mesh of a path connecting source and target points through three dimensional structure and other obstacles, while satisfying design rules and best practices. The routing system integrates two common engineering software tools: CAD for drawing three dimensional models and CAE for analysis, together with a computer aided problem solver constructed for the routing problem.

2 Problem background

The placement of electrical wiring harnesses in aircraft is governed by numerous regulatory and functional design rules which must be satisfied for certification. Electrical wiring looms can be comprised of thousands of cables and are generally manually routed by engineers using personal knowledge and experience to determine path placement. CAD software tools are used to assist the engineer in adding detail to a path, but the actual route taken by a harness is determined manually. This process is highly repetitive and is difficult to find optimum solutions. In addition, electrical wiring design often proceeds in parallel with principle structural design and is subject to changes in structure that occur with subsequent design iterations, requiring time consuming rework for any harnesses affected. In a similar way, hydraulic and pneumatic pipes in aircraft are manually routed and are governed by different set of design rules. The repetitive, rule-governed nature of the routing process makes it a prime candidate for application to a KBS.

Figure 1 shows an example of a complicated loom consisting of electrical harnesses and hydraulic and pneumatic piping in an internal payload storage area of the F-35 Joint Strike Fighter (JSF).

3 Related work

This section provides a brief background of relevant work in the areas of intelligent system development and routing technologies and methodologies.

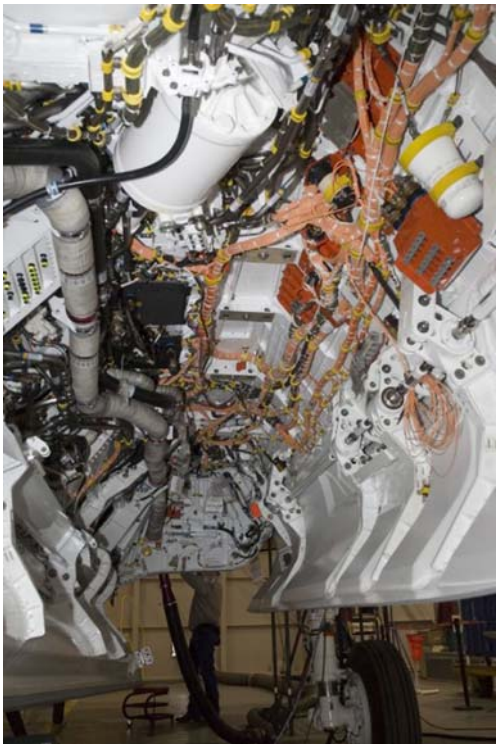


Fig. 1 Example electrical wiring and pipe loom in weapons bay of F-35 JSF [5]

3.1 Intelligent systems

Artificially intelligent systems such as knowledge based, expert and fuzzy systems are used across a wide range of problem domains and industries to deliver design automation and validation, control of industrial processes and numerous other tasks. These systems vary in complexity from simple procedural systems that automate well defined engineering tasks, to higher level systems which use reasoning and semantics to emulate human thought and problem solving processes. The major activities of the KBS development cycle include the following:

3.1.1 Knowledge acquisition (KA)

In order for outputs of a KBS to be relevant and useful, it must take into account all applicable knowledge involved in performing the task manually. This requires collecting problem domain knowledge and remains one of the significant challenges in the successful implementation of KBSs. Domain knowledge may be explicit or tacit in nature. Explicit knowledge is often procedural and is relatively straightforward to collect, interpret and represent in a software system. The main challenge lies in collecting tacit knowledge which is generally not well defined and most often lies in the minds of experienced engineers. A large amount of literature has

been published on the subject of KA. A quick search on the internet will yield hundreds of papers, a small selection of which is given in the references section [7–9,20,30].

3.1.2 Knowledge modelling (KM)

Following the KA phase, collected knowledge must be modelled and represented in the system. KM is a discipline itself with numerous methodologies described including CommonKADS [26] and object-modelling technique (OMT) (Rambaugh et al., 1991). These methods use Universal Modelling Language (UML) and object-oriented (OO) techniques including class, activity and state diagrams, and principles of inheritance, association and abstraction. In these methodologies, knowledge is treated as a set of objects sorted into categories, each with various properties and interrelationships describing the problem domain. Rules are then formulated based on this knowledge and implemented as "IF {condition}, THEN {statement}" rules. Resources and courses in KM can be found freely available on the internet.

3.1.3 KBS development

The desired scope of a KBS must be clearly defined at the beginning of application development. The detail and complexity delivered by KBSs can vary widely and can be divided into four categories [3]:

- *Automation of narrow tasks*. Includes automating rudimentary tasks such as drawing tools used for building digital models (e.g., lines, circles, etc.).
- *Automation of model and data abstraction*. Provides higher level operations which can be performed on the former which add detail, or knowledge, to the product [e.g., geometry operations such as mid-surface extraction and defeaturing tools, and programming tools such as application programming interfaces (API's)].
- *Automation of a documented design process*. Automates a complete engineering task consisting of a number of lower level tasks covered by either of the first two levels into a single process where the user specifies critical parameters.
- *Discovering solutions to unique problems*. Applies reasoning, or semantics, from a library of multidisciplinary knowledge and experience of varying types to solve new problems situations occurring within the domain of knowledge.

3.1.4 KBS integration, test and evaluation and ongoing support

Following the development of KBS, the application must be integrated into the organisational working environment and thoroughly tested. Software systems also need to be maintained in terms of knowledge content and compatibility with various operating system platforms.

The intelligent-CAD (ICAD) system, developed by Knowledge Technologies International (KTI) (now owned by Dassault Systemes), was a popular software environment for building KBS applications from the late 1980s to early 2000s. The ICAD system was based on the LISP programming language and enabled development of systems for automation of design and manufacturing data. Numerous engineering companies including BAe Systems, Boeing, Airbus, and Jaguar, among many others, have utilized the system for rapid prototyping, concept evaluation, and component design and manufacture [4]. Applications developed using this technology typically aim to minimise design decisions, often through the specification of handful of input parameters necessary to generate a complete design. Over recent years, support for the ICAD system by the parent company has gradually been withdrawn and its use in industry has diminished.

MOKA is another framework for developing knowledge based systems. It consists of a suite of tools and methods which provide structure to the development process, covering all stages of the KBS development process [2]. The MOKA Website gives good information on this system and has links to other frameworks and projects in the area of KBE (http://www.kbe.coventry.ac.uk/moka/).

Other technologies such as Dassault Systemes' Knowledgeware, which is built into the CATIA design system, assist designers in capturing and reapplying knowledge for automating design of similar products, and linking product knowledge such that changes in one component are reflected in affected parts (Dassault Systemes Website, 2006).

3.2 Routing

The routing problem is commonly encountered in numerous fields ranging from electronics, navigation systems, artificial intelligence (AI), and data flow in computer networks. Examples include design of printed circuit boards (PCBs), very large scale integrated (VLSI) circuits, global positioning system (GPS) navigation, computer games, and robot AI.

Methods and tools used for VLSI routing automation provide a good starting point for addressing electrical loom and pipe design problems in aerospace vehicles. Computer processors consist of millions of logic components interconnected using very fine wires within a very small space. Early algorithms for circuit design were based on a multi-layered two dimensional approach with one of the objectives to minimise the number of layers due to limitations in component manufacturing. Improvements in technologies and manufacturing process for electrical components have increased the number of layers that can be used, leading to a reduction in

chip size. However, VLSI routing automation is still not a completely three dimensional problem.

In general, once physical component layout is defined and routing requirements given, usually in the form of a netlist of pins to be connected, the routing process consists of four main steps [10]:

- Region definition: problem is divided into smaller routing problems.
- Global routing: planning phase which assesses and prioritises nets to maximize completion rate (proportion of solvable nets), and minimize total path length, especially for critical nets.
- Region ordering: determines order in which regions are routed to avoid congestion.
- Detailed routing: determines the exact path taken by wires including layers and connecting contacts.

Many algorithms have been developed for the VLSI routing task for both global and detailed routing. One of the earliest works in this area is Lee's breadth-first maze algorithm [18]. This algorithm uses a grid based representation of the search area with walls, paths, and source and target terminals. The search proceeds by propagating a wave from source and/or target terminals, and assigns values to each node depending on distance from the source or target. A backtracking phase then determines the shortest path between the two terminals. Fig. 2 shows a simple two dimensional maze with a source and target terminal (denoted S and T, respectively) the numbers in each node of the maze represent the shortest rectangular distance to the target. This algorithm is popular due to its simplicity and ability to guarantee the shortest path for a single net. However, several limitations make it unsuitable for use in real world problems, including its low efficiency ($O(d^2)$ for two dimensions and $O(d^3)$ for three dimensions), and sensitivity to net ordering, making optimal solutions very difficult or impossible to find. Also, due to the breadth-first search technique employed, the search proceeds equally in all directions until the target is found, leading to high memory requirements and long run times. These limitations and the complexity of the VLSI routing problem make Lee's maze algorithm unsuitable for most realistic routing problems. Instead, powerful heuristics are employed which can find near optimal solutions for problems with a large number of nets.

Numerous algorithms have been developed which extend Lee's basic principles employing intelligent searching techniques including depth-first, best first and greedy searching. Examples include Soukup's algorithm which attempts to route multiple connections at once [28], and Hadlock's algorithm with adds penalties for deviations away from the ideal (no obstacle) path from source to target [10]. Other examples include channel and switchbox routers which place horizontal segments on one layer and vertical segments on an adjacent layer connected by contacts called vias [12,36], line routers such as Hightower's algorithm [13], as well as gridless techniques [6]. Much literature has been published on these subjects and can be found in the references.

The A* (A Star) algorithm is an example of a grid based path finder which extends Lee's maze algorithm. It is used extensively in computer game navigation and is of interest when considering a rule based routing system. A* employs best-first search techniques, leading to a more direct search. In A* the following cost function is evaluated for each node searched:

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

where $f(n)$ is the estimated node cost, $g(n)$ is distance from the source and $h(n)$ is the estimated cost to the goal using a heuristic. The algorithm favours nodes with a lower $f(n)$ score. The algorithm is optimal and complete provided that the function which calculates $h(n)$ is an "admissible" heuristic, meaning that it does not over-estimate the distance to the target [24]. Fig. 3 shows the F, G, and H terms evaluated at each node of simple maze, and a shaded path connecting source and target terminals S and T.

Intelligent system approaches to the routing problem have also been developed. Previous works in the area of intelligent path finding have been predominantly in the area of

2	1	2	3	4	5		
1	**S**	1	2	3	4	12	
				4	5	11	12
				5	6	7	8	9	10	11
10	9	8	7	6	7	8	9	10	11	12
11	10	9	8	7	8	9	10	11	12	
12	11	10	9	8	9	10			**T**	
	12	11	10	9	10	11				

Fig. 2 Maze routing algorithm

G=4 H=8 F=12	G=3 H=7 F=10	G=2 H=6 F=8	G=3 H=5 F=8		G=9 H=3 F=12	G=10 H=2 F=12	G=11 H=3 F=14
G=3 H=7 F=10	G=2 H=6 F=8	G=1 H=5 F=6	G=2 H=4 F=6		G=8 H=2 F=10	G=9 H=1 F=10	G=10 H=2 F=12
G=2 H=6 F=8	G=1 H=5 F=6	**S**	G=1 H=3 F=4		G=7 H=1 F=8	**T**	G=9 H=1 F=10
G=3 H=7 F=10	G=2 H=6 F=8	G=1 H=5 F=6	G=2 H=4 F=6		G=6 H=2 F=8	G=7 H=1 F=8	G=8 H=2 F=10
G=4 H=8 F=12	G= 3 H=7 F=10	G=2 H=6 F=8	G=3 H=5 F=8	G=4 H=4 F=8	G=5 H=3 F=8	G=6 H=2 F=8	G=7 H=3 F=10
G=5 H=9 F=14	G=4 H=8 F=12	G=3 H=7 F=10	G=4 H=6 F=10	G=5 H=5 F=10	G=6 H=4 F=10	G=7 H=3 F=10	G=10 H=4 F=14

Fig. 3 A* algorithm

automated electronic component design including integrated circuits and printed circuit boards. One example is an expert system for the VLSI channel routing problem described above [32]. This system identifies several metrics for measuring routing performance and has an "expert" module containing the applicable knowledge and rules for addressing each, located around a central problem space termed a "blackboard". As the solution progresses, the various experts are consulted. A second example is a knowledge-based routing system for VLSI channel and switchbox problems called WEAVER [14], which has a similar system architecture to the previous example. Because the number of nets to be routed in a VLSI problem is so large, any manual design work is very time consuming. Thus one of the key metrics for evaluating results is the completion rate, or proportion of nets solved.

An expert system for automation of pipe routing design in ships has also been described [17]. Main aims of the system are to minimise user decisions, provide a user friendly environment, and provide simple methods for editing the knowledge base. The system consists of a standard expert system shell with knowledge modelled using the OMT method mentioned above, a CAD software package, and a solving application, which interfaces using API functions built into to the software components.

4 Project details

This project is a collaborative research and development effort between RMIT University and GKN Aerospace Engineering Services Pty. Ltd., both based in Melbourne, Australia. Funding for a three year PhD scholarship was provided by the Australian Research Council. The project began in Early 2005 and will conclude at the end of 2007.

The main objective of the project was to build up a capability for knowledge based system development by creating an intelligent system for routing automation. More specifically, the project aimed to develop engineering tools and methods that partially or fully automate the design of system runs through existing structure. Ideally the output would be CAD geometry describing the system path. Wherever possible the system was to implement well established technology existing in other industries including intelligent algorithms from VLSI and computer game path finding applications, CAD interfacing techniques, and knowledge based and expert system architectures.

5 System structure

This section describes the current configuration of a KBS for three dimensional routing in aerospace vehicles. Applications of the system include electrical wiring and hydraulic/pneumatic piping design. The system reads model data together with source and target terminal locations and solves the routing problem, satisfying constraints. Wire/pipe geometry and other information required to describe the system path is output as CAD geometry and an FE model. The routing system is comprised of five layers, which are classified as either human interface or system layers (Fig. 4).

The interface layers include an input and editor layer. The input layer interfaces with users of the system and is used for defining specific cases of a routing problem, including terminals to be connected and a set of rules to be followed (e.g., electrical, hydraulic, pneumatic, etc.). The user passes a CAD model of existing structure and obstacles such as equipment and subsystems to the system, which in turn is meshed using FE pre-processor software (e.g., HyperMesh). The editor layer interfaces with the domain expert and knowledge engineer who interpret design rules and best practices in the form of "IF {condition}, THEN {statement}" rules. These are stored in external libraries and are accessed by system layers.

The system layers consist of a data layer, problem solving layer, and results layer. The data layer stores both case specific data (models and requirements), and rules sets divided into libraries for different routing applications. This layer is separate from the problem solving component of the system to maximize flexibility and reduce rework in adding or changing design rules. The problem solving layer consists of an applet which interprets model and requirements data, a solving algorithm, and an inference engine that accesses the rule base as necessary to satisfy applicable design constraints. The output layer writes results in two formats, one read directly into CAD software which can be used as a spine for drawing tools to add detail, and the other is an FE model which consists of several components describing geometry and rules accessed and intent taken at each point along the routed path. Each of the five layers will be described in detail in the following sections.

Steps in the routing system design process are summarised in the following flowchart in Fig. 5. Firstly, physical structure is designed and modelled by structural engineers using CAD software. Electrical or piping requirements are defined in terms of start and finish locations and other relevant characteristics such as diameter, category of load, etc. The three-dimensional CAD model is exported using a neutral file format. The CAD model is converted to a discrete format suitable for applying a grid-based search algorithm. The discrete data set is extracted and input into the maze algorithm which determines paths for each set of source and target terminals, adhering to constraints stored in the knowledge library. The library module is interchangeable for different routing applications, not necessarily limited to aircraft. After completion of the path finding process, the output path is converted to wireframe geometry, which is imported into a

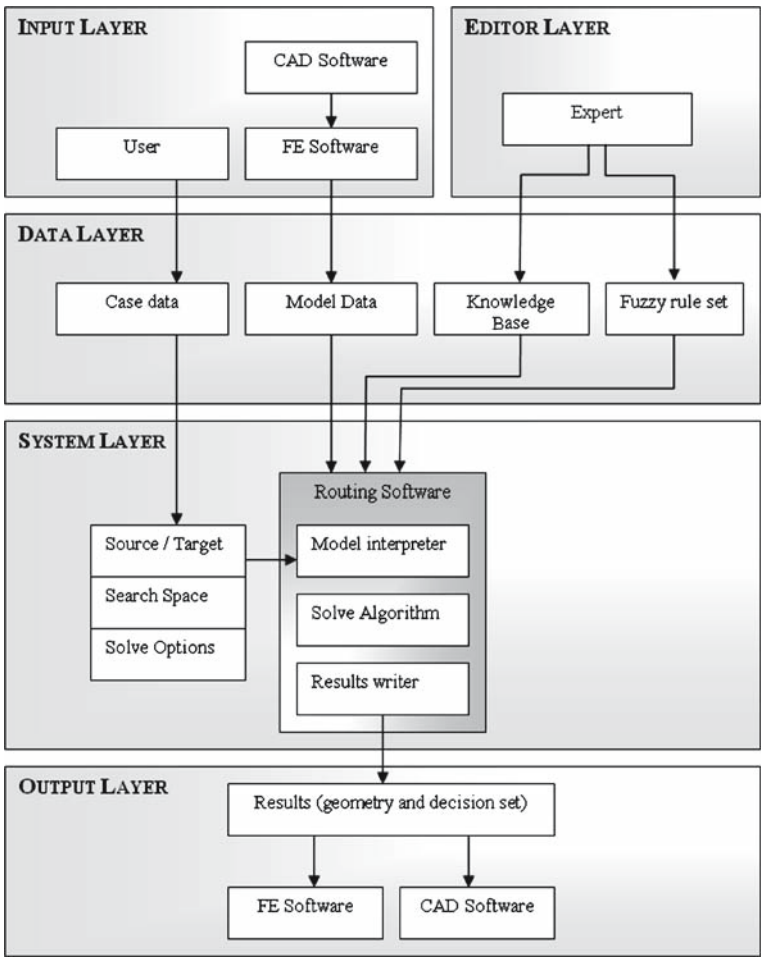


Fig. 4 Knowledge based router system structure

CAD package and detail added according to the knowledge base consulted in the process.

6 Editor layer

The editor layer is the main interface for domain experts to impart knowledge to the system. As mentioned previously, design of electrical wiring systems for aircraft is a complex task with hundreds of rules and best practices which must be satisfied. Currently there is no section of the Federal Airworthiness Requirements for transport category aircraft (FAR-25) dedicated to wiring design practices. Instead, a number of sections briefly touch on the subject including 25.1301/1309, 25.1529, 25.1353, 25.869, AC 43.13-1b, AC 25-16, AC 25-10, and policy memos [25]. Engineers must then search though a large amount of data to single out rules applicable to wiring and monitor amendments made by governing bodies. A different set of rules applies for military aircraft, contained in MIL-W-5088L: Military Specification—Wiring, Aerospace Vehicle. The following, obtained from civil and military airworthiness requirements (FAR-25 and MIL-W-5088L), lists just some areas of consideration when designing electrical wiring systems:

- Electrical loads
- Bend radii
- Breaker/wire sizing
- Clamping
- Separations and terminations
- Passing through lightening holes
- Grounding and bonding
- Conduits and insulation
- Connectors
- Unused wiring/slack
- Riding on structure/other wires
- Documentation

The system contains some simple predefined rules, as well as a rule editor which consists of a Windows form with simple controls for entering rule conditions and actions. Families of rules can be exported in separate libraries. Rules currently supported by the system include

- Path profile
- Bend radii
- Clamping rules

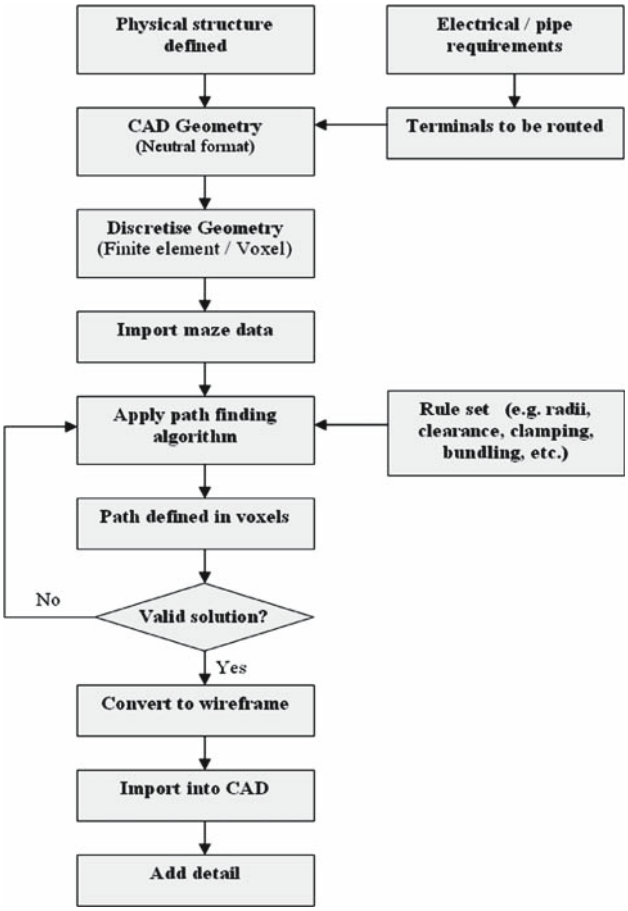


Fig. 5 Routing procedure

- Attract and repel rules (discussed in section on routing algorithm below). Includes EMS rules, conforming to walls and other types of structure, conforming to routed paths.

7 Input layer

The input layer is the main interface for users of the routing system. When the routing software is executed, the user defines the required search space and import options such as mesh size and element type, and imports the model for solving. Source and target locations can be defined in the model itself and interpreted by the solver, or by typing in the coordinates of these points in x, y, z form. Applicable rules are selected (e.g., electrical design, hydraulic pipe design, fuel line design, etc.). Several export options are available relating to output visualisation.

Properties for individual routing problems can be saved, allowing these settings to be retained in the event that geometry is altered. Multiple jobs can be built up into sessions, allowing many runs through the structure to be made without user input.

8 Data layer

The data layer stores both case data and rule libraries. Case data consists of inputs from the user, including the model and set of nodes to be connected, as well as a list of rules to be applied. This data are stored in simple XML tree files and can be saved/read by the system. Knowledge is represented in the system using rule libraries and fuzzy rule sets. Whereas many KBSs have rules hard-coded into the software, the rule bases for the intelligent routing system are maintained separately from the system core, allowing knowledge to be added and amended without rebuilding the entire application. Rules are implemented as "IF. . ., THEN. . ." statements which use local search functions to determine structure and other obstacles with which to interact.

Rule libraries are stored in a comma separated format and are described by several attributes including: cell type which is the subject of the rule, condition for rule to be applied and action to be taken. The rule library can contain multiple rules and the solver loops through these where applicable.

9 Problem solving layer

9.1 Reading model data

A grid-based algorithm was selected to perform the path finding task, requiring the continuous three dimensional model of surrounding structure and obstacles to be converted to a discrete form. A FE modelling approach is used for this process which is common practice in the engineering process for analysing the response of structures to loading. This method takes advantage of existing knowledge and tools. A solid mesh of structure and obstacles is generated using automatic meshing tools in a FE pre-processor software package. The meshes of various components which fall within the search space can either be exported as a single model or as separate components, allowing rules to be applied to each component independently. Mesh coarseness can be varied depending on accuracy required.

The routing software reads and arranges the FE mesh into a three dimensional maze object consisting of a regular grid with each cell described by an x, y, z integer address and cell type (e.g., empty space, wall, start location, finish location, routed path, excluded zone, etc.). The format of the maze object makes it easily navigated by the solving algorithm described below. The main drawback of this method is the high mesh fineness required to get a good representation of the geometry without missing data.

Given that aircraft typically undergo numerous design upgrades throughout their service life, the system can recognise previously routed wire harness or pipe paths and represent them accordingly in the maze. This allows additional cables or pipes to interact with existing routed systems.

9.2 Routing algorithm

The algorithm used by the routing system extends the A* algorithm described above to three dimensions, whereas most VLSI routing algorithms are based on a multilayered two dimensional approach. The algorithm also extends the score function, adding extra terms for rules implemented from the knowledge base:

$$f(n) = g(n) + h(n) + i(n) + j(n) + \cdots \quad (2)$$

These extra terms act to increase or decrease the total path score for nodes falling within rule areas of influence, thus shifting the path direction. For example, a rule $i(n)$ might specify that if there is a routed path in the search space, to follow that as closely as possible to reduce the amount of clamping required for the new path. Such a rule will typically be described by a weighting to apply to the cost function, a radius of influence, and a decay rate such that nodes closer to path nodes will have higher influence on the $i(n)$ term. The function to calculate $i(n)$ is given in Eq. (3), where W is rule weighting, T is distance of rule target cell, and D is decay rate. A node which is two nodes away from a path node will have a lower $i(n)$ value than one immediately adjacent due to the decay rate:

$$i(n) = W \times (1 - (T - 1) \times D) \quad (3)$$

Thus nodes closer to the previously routed path will, have a lower $f(n)$ score due to the influence of the $i(n)$ term, causing the algorithm to head in that direction. Similar rules could be made for particular nets with electro-magnetic sensitivities (EMS) whereby particular category wires must not lie within a given radius of another wire type. In this case the additional terms in the algorithm would increase the $f(n)$ value, causing the algorithm to search away from the repelling wire.

The algorithm supports routing of multiple nets within same search space, and routing of multi-terminal nets which have more than one target location. In the later case, the system finds the path between two terminals first and connects additional terminals to the original path.

10 Output layer

Resultant paths from the routing process are output using a neutral CAD format called IGES (initial graphics exchange Specification). A wireframe representation of the path including straight lines and curves is written by the router and can be read directly by most modern CAD packages. The path is imported into the existing assembly of structure and subsystems. Actions are then performed on the wireframe geometry, adding detail necessary for a complete digital representation of the routed part (e.g., extruding a profile along the path).

As mentioned previously, difficulties in understanding the rational behind design decisions made by others can occur due to a lack of communication. For example, consider a harness which must be routed past an obstacle which can take one of two paths, an upper path or a lower path—both with equal merit (Fig. 6, left). An engineer using their own knowledge and experience selects the top path due to personal preference. This is acceptable since the two paths have the same cost, however, the reasoning behind this decision is not documented. Suppose an additional component is to be placed on the upper side of the existing obstacle by an engineer who has no knowledge of the routing process. The engineer would face a conflict between the routed harness and placement of the new part. In this case the conflict could be easily resolved by moving the harness to the lower side of the obstacle Fig. 6, right). However this may not be immediately clear to the designer, and would require time to resolve. This and similar conflicts could be avoided through implementing methods of communicating the knowledge used in the process.

It is therefore important for intelligent systems to not only implement knowledge in design automation, but communicate the knowledge used allowing designers to understand

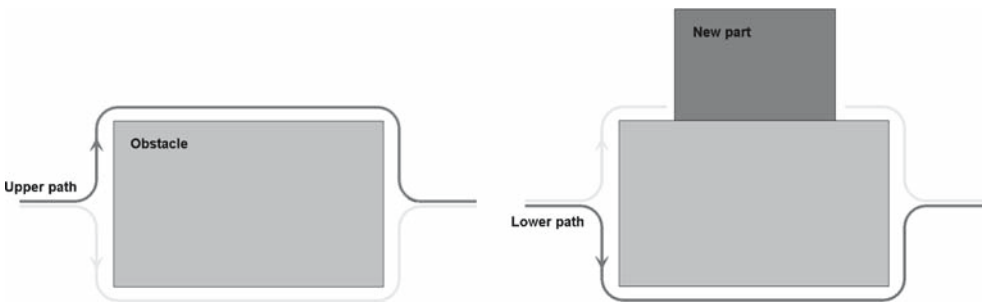


Fig. 6 Example of conflict resolution

limitations of the process and room to alter the design as necessary.

To this end, results from the routing process are also output as an FE mesh containing physical geometry, routed paths and a knowledge layer. Physical geometry includes walls and other entities encountered in the search space and are represented using three dimensional hexahedral elements. Routed paths are displayed in wireframe and are defined by one dimensional bar elements. The knowledge layer provides details of the methods used in the routing process including rules accessed and their influence on the routed path, as well as a three dimensional map of the area searched by the algorithm. Examples from both output methods are given in the following section.

11 Results

This section describes the complete process for the automated design of routed paths using the intelligent routing software. Five key stages in the design process are described with accompanying screen captures of a test case. The test case is an internal payload storage area of a modern fighter aircraft similar to that shown in Fig. 1.

11.1 Step 1: Structural design

At this point the user will have a CAD model of structure to be traversed and any equipment or obstacles that fall within the search space (Fig. 7). Source and target terminal locations are defined using point objects. In most cases the search area consists of an assembly containing multiple parts such as structure, subsystems, etc. Multiple parts can be imported into the solver with each defined uniquely, allowing rules to be applied independently to each. Geometry is exported using a format readable by CAE software in as many sections as necessary to describe the model in terms of applying rules which will result in a valid path. In this example these sections include the following: principle structure, sample payload and subsystems. Three paths are to be routed within the search space with points for source and target terminals defined.

11.2 Step 2: Discretising geometry

This second stage involves converting the continuous model geometry to a discrete form, using FE meshing software. Model sections output from CAD software are imported into the CAE meshing software and are individually meshed using automatic meshing tools (Fig. 8). Nodes are created at the previously defined source and target terminal locations and entity sets are created for each. In this example, principle structure, payload and subsystems are meshed and exported independently.

11.3 Step 3: Path finding

The third stage involves use of the intelligent routing software. The software runs on Windows 2000 or higher platforms with Microsoft .NET framework version 1.1. The application itself uses a Windows form interface and is largely mouse operated. A series of tab pages guide the user in setting up the model for routing (Fig. 9). User inputs include:

- *Search space*—define dimensions of search area (length, width, height), and coordinate system.
- *Import options*—specify FE mesh to import, element size and type, and cell type to fill.
- *Rule sets*—select from set of standard rules, or use rule editor to create custom rules.

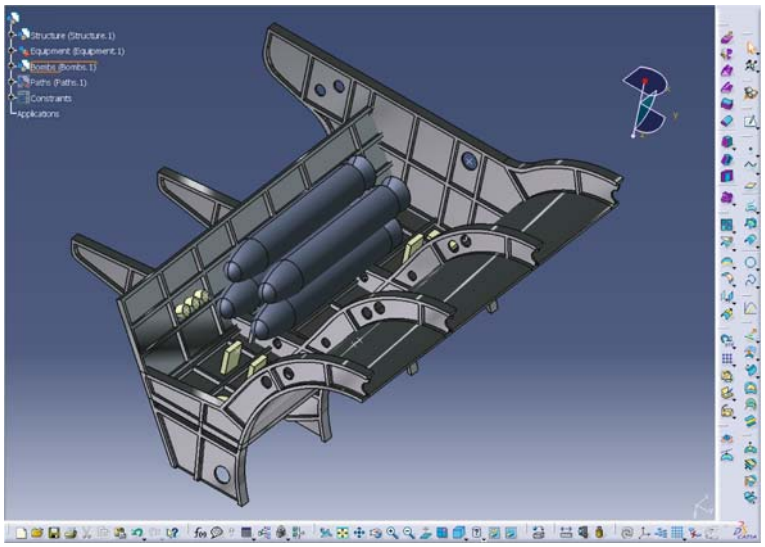


Fig. 7 Step 1: generating geometry

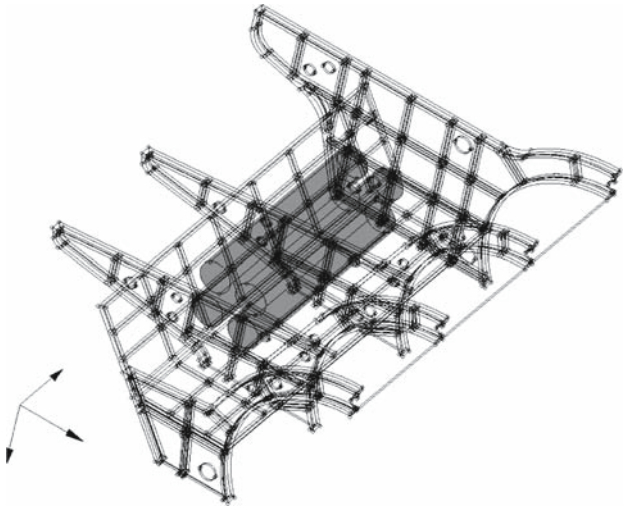


Fig. 8 Step 2: meshing geometry

- *Solver options*—define solver type, directions to search, and multi-net or multi-terminal routing.
- *Export options*—select layers to be displayed in FE output.

Property files containing all user inputs can be written, allowing simulations to be repeated, in the case of changes in geometry or similar routing jobs. Multiple property files can be built into sessions, allowing multiple routing jobs to be run without user input. When the routing job is solved, a simplified preview of the resultant path can be viewed within the routing software, to be used as a quick check for identifying a satisfactory output. Results are exported as an IGES model containing path geometry, and FE model containing geometry, routed paths, and design knowledge.

The meshes of model sections are imported into the maze object, with each defined by a different cell type. Principle structure is defined as walls, sample payload as a no-go zone, and subsystems as a different cell type. Rules can then be applied to the different cell types. In this case a rule for conforming to walls, and a rule conforming to previously routed paths are applied.

11.4 Step 4: Viewing results

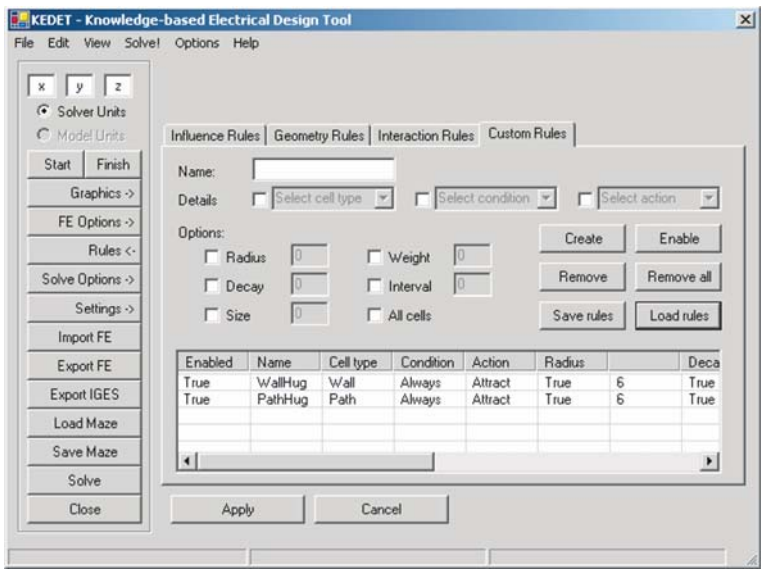
The FE output is imported into the CAE meshing software for viewing results. The output for the test case is shown below (Fig. 10, left). The regular grid used in the maze object can be seen, with three geometry sections (structure, payload and subsystems) represented by cube elements. The three routed paths are represented in wireframe by line elements. The knowledge layer consists of cube elements which follow the routed paths. These elements are colour coded depending on the rules accessed to determine each path step. The output also shows a map of nodes searched and the areas of structure where the wall attract rule was used (Fig. 10, right).

11.5 Step 5: Adding Detail

The second output from the routing software is a wireframe IGES model of the path. This is imported into the CAD model and detail added. In this case a circular profile is extruded along the length of the routed paths and is incorporated into the existing model of structure, payload and subsystems (Fig. 11).

In its current configuration, the system successfully demonstrates knowledge based and path finding concepts. The quality of routed paths for a number of test cases has been examined and was found to be very good, with several rules

Fig. 9 Step 3: Routing software



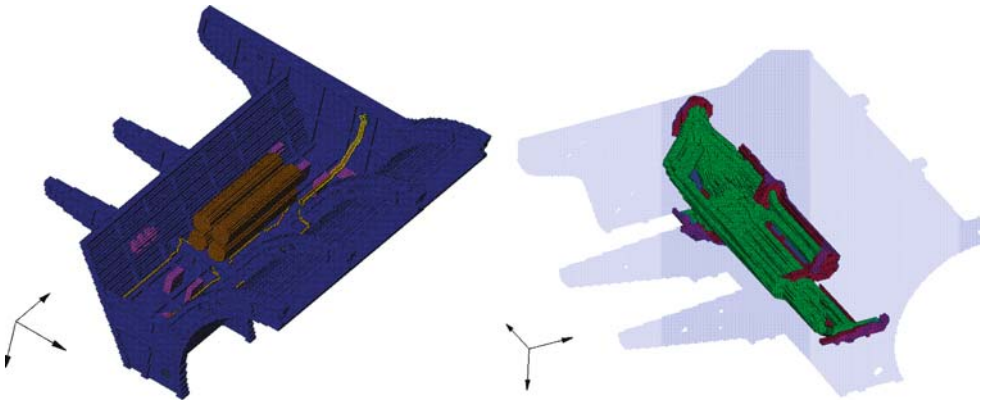
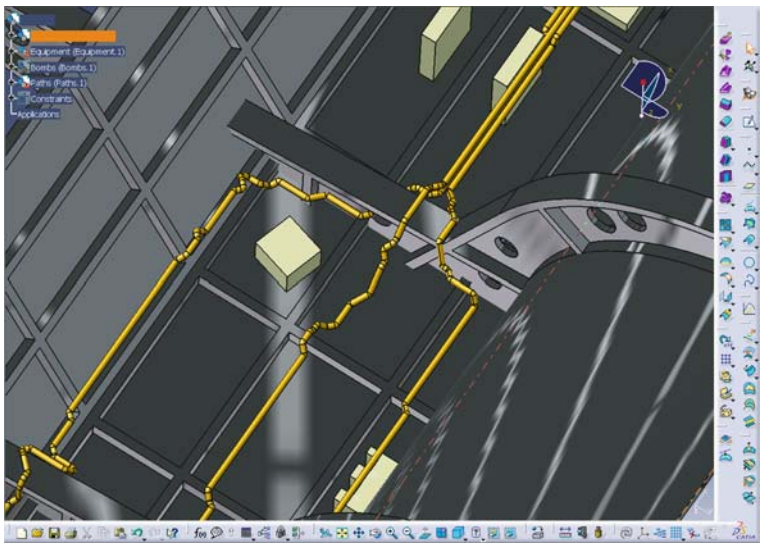


Fig. 10 Step 4: FE output

Fig. 11 Step 4: IGES output



successfully implemented. At this stage in system development, rule parameters are based on estimates rather than domain specific data, making it difficult to compare results directly with manually routed paths. Despite this, the behaviour of the path finding algorithm, demonstrated by the geometry of routed paths, was found to be effective. In the above example, two rules were implemented specifying that walls and routed paths should be followed closely. Both rules were validated, with the path not deviating from the structure by more than the specified rule radius of effect, and following routed paths until separation was necessary to reach the target.

System performance in terms of run time for large models (approx 3 million nodes) was in the order of 10 to 15 minutes, running on a PC with average specifications.

The system will be developed further by improving the knowledge base and rule inferencing process which will enhance the output to a point where resultant paths are to an acceptable standard for certification. The knowledge base will be extended by modelling knowledge contained in guidelines. Implementation of knowledge throughout the solution process will be enhanced by integrating an expert system shell to better manage rule inferencing.

12 Conclusion

Discussed in this paper was development of an intelligent routing system which uses knowledge based techniques for management of engineering knowledge including acquisition, modelling and implementation. The resulting system successfully routes paths through a three dimensional maze constructed from a FE mesh of obstacles within a set solution space. The system architecture was described by its principle components including: input layer for system users, editor layer for domain experts, data layer for knowledge modelling and representation, problem solving layer, and output layer. The system delivers a CAD-readable representation of path geometry and a FE mesh containing geometry and knowledge accessed during the routing process. An easily updated

knowledge base provides flexibility to implement new routing methods and rules, allowing the system to be applied to new problem domains (e.g., air conditioner ducting). The system performed well in routing a number of test cases, satisfying all design rules and constraints. Output quality of routed paths in terms of relevance to particular problem domains will be improved with addition of more detailed domain specific knowledge to the knowledge base, implementation of new types of rules, and improvements to the intelligent rule inferencing process. The ultimate, and achievable, aim of the system is to output routed paths of similar quality to those designed manually by a human expert, in a shorter time span.

References

1. Arnold MH, Scott WS (1988) An interactive maze router with hints. Lawrence Liver-more National Laboratory, University of California
2. Brimble R, Oldham K, Callot M, Murton A (1999) MOKA: a methodology for developing KBE applications. In: Proceedings of the 8th European conference on product data technology, Norway
3. Brown DH, and Associates (2006) Knowledge-based engineering systems: applying discipline and technology for competitive advantage. Gardner Publications, USA. http://www.mmsonline.com/articles/0600sup.html. Online May 2006
4. Cooper S, Fan I, Li G (2001) Achieving competitive advantage through knowledge-based engineering. Department of Enterprise Integration, Cranfield University
5. F-35 Joint Strike Fighter Official homepage. http://www.jsf.mil. Online October 2006
6. Finch AC, Mackenzie KJ, Balsdon GJ, Symonds G (1985) A method for gridless routing of printed circuit boards. In: Proceedings of the 22nd Conference on Design Automation
7. Gil Y, Kim J (1999) Deriving expectations to guide knowledge base creation. In: Proceedings of the sixteenth National Conference on Artificial Intelligence
8. Gil Y, Paris C (1994) Towards method-independent knowledge Acquisition. In: Knowledge acquisition special issue: the integration of machine learning and knowledge acquisition vol 6. pp 163–178
9. Gil Y, Blythe J, Kim J, Ramachandran S (2001) An integrated environment for knowledge acquisition. In: International conference on intelligent user interfaces
10. Groeneveld P (2005) Electronic design automation, Part 1. Technische Universiteit Eindhoven, The Netherlands
11. Guruswamyl M, Wong DF (1991) A General Multi-layer Area Router. Department of Electrical and Computer Engineering, The University of Texas at Austin
12. Hamachi GT, Ousterhout JK (1984) A switchbox router with obstacle avoidance. In: Proceedings of the 21st Conference on Design Automation IEEE Press
13. Hightower DW (1969) A solution to line-routing problems on the continuous Plane. In: Proceedings of the Sixth Annual Design Automation Workshop
14. Joobbani R, Siewiorek DP (1985) WEAVER: A knowledge-based routing expert. Department of Electrical and Computer Engineering, Carnegie-Mellon University
15. Junghanns A, Klein R, (2000) Using search in knowledge-based engineering. In: Proceedings of ECAI-00, Germany
16. Kaas E The NGRAIN technology difference explained a whitepaper for technical evaluators of visualization and simulation technologies. NGRAIN Corporation. Vancouver, Canada
17. Kang S-S, Myung S, Han S-H (1999) A design expert system for auto-routing of ship pipes. J Ship Product
18. Lee CY (1961) An algorithm for path connections and its applications. IRE Trans Electron Comput EC-10(2)
19. Lester P (2006) A* pathfinding for beginners. Almanac of policy issues website. http://www.policyalmanac.org/games/aStarTutorial.htm. Online May 2006
20. Lieu YI (1990) Knowledge acquisition: issues, techniques, and methodology. In: Proceedings of the 1990 ACM SIGBDP Conference on Trends and Directions in Expert Systems
21. Lunow RE (1988) A channelless, multilayer router. Lawrence Livamore National Laboratory, California
22. MIL-W-5088L: Military specification—wiring, aerospace vehicle. Department of Defense, United Stat. 10 May 1991
23. Moosa Z, Edwards D (1995) An investigation of iterative routing algorithms. Department of Computer Science, University of Manchester
24. Russel R, Norvig P (2003) Artificial Intelligence a Modern Approach, 2nd edn. Prentice Hall, Englewood Cliffs
25. Sadeghi M (2003) Electrical wiring practices. Federal Aviation Administration. Aircraft Electronics Association Convention
26. Schreiber G, Wielinga B, de Hoog R, Akkermans H, Van de Velde W (1994) CommonKADS: a comprehensive methodology for KBS development, Expert, IEEE 9(6):28–37
27. Smith AL, Bardell NS (2005) A driving need for design automation within aerospace engineering. In: 11th Australian International Aerospace Congress, Melbourne, Australia
28. Soukup J (1979) Global router. In: Annual ACM IEEE Design Automation Conference, pp 481–484
29. Stevenson A (1996) Voxels and volumetric representation. The University of British Columbia, Vancouver, Canada
30. Studer R, Benjamins VR, Fensel D (1998) Knowledge engineering: principles and methods. Data Knowled Eng 25(1–2):161–197
31. Tehranipoor M (2005) CAD Algorithms—routing. Department of Computer Science and Electrical Engineering, University of Maryland, Baltimore County
32. Vakil D, Zargham MR (1988) An expert system for channel routing. Computer Science Department, Southern Illinois University
33. Van der Velden C, Bil C, Yu X, Smith A (2006) Towards a knowledge based cable router for aerospace vehicles. In: Information and Knowledge Engineering Conference, Las Vegas, United States
34. Van der Velden C, Bil C, Yu X, Smith A (2005) Mathematical techniques applied to knowledge based engineering design systems. In: Engineering Mathematics and Applications Conference, Melbourne, Australia
35. Yap P (2002) Grid-based path-finding. Department of Computing Science, University of Alberta, Edmonton, Canada
36. Yoshimura T, Kuh ES (1982) Efficient algorithms for channel routing. IEEE Trans Comput Aided Desi Integrat Circ Syste