52nd AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics and Materials Conference<BR> 19th
4 - 7 April 2011, Denver, Colorado

AIAA 2011-2148

On the development of a heuristic routing application for the automatic wire harness design in the aircraft

Zaoxu Zhu[1], M.J.L. van Tooren[2] and S.W.G. Van der Elst[3]
Faculty of Aerospace Engineering, Delft University of Technology, Delft, 2629 HS, the Netherlands

Abstract Based on the use of dedicated software tools, Knowledge Based Engineering (KBE) techniques are able to capture and reuse engineering product and process knowledge to reduce time and cost during product development. In this paper a conceptual design of an automatic wire harness routing application is proposed, based on KBE. As the core of this application, the wire harness routing algorithm is discussed in detail. Through comparison with related methods and existing algorithms, a suitable algorithm, named A* is selected. In order to comply with the complex aircraft specific routing environments, the selected algorithm is enhanced with so-called wall-attraction capabilities. The resulting algorithm is called the wall-attraction A* (W*) algorithm, and features a settable parameter that controls the balance between performance and admissibility of the solution. By conducting a set of experiments the influencing parameter is optimized. A case study is finally presented, featuring a conceptual aircraft environment that validates the above theories. Applying the W* algorithm, the wire harness routing application can connect any source and destination nodes satisfying according the relevant constraints meanwhile the valuable time and resources are saved.

I. Introduction

NOWADAYS with increasing complexity of aircraft systems, the design process also become increasingly complicated. Hence, a method which can provide a satisfactory design conforming to the customer requirements eagerly need to be proposed. This method should not only provide business opportunities with decrease the consumed time, cost and error for design, but also support engineers to increase their productivity and creativity and release the workload.

Facing this new challenge, the Knowledge Based Engineering (KBE) system is adopted. KBE, defined by La Rocca, is a technology that bases on the use of dedicated software tools which are able to capture and reuse product and process engineering knowledge. The main objective of KBE is reducing time and cost of product development by means of the following:

- Automation of repetitive and non-creative design tasks
- Support of multidisciplinary integration from the conceptual phase of the design process

Traditionally, KBE has some features similar as geometric modeling in CAD environments, allowing geometry to be rapidly created and modified using rules. Gradually KBE systems allow the instantiation of each single object of the series by using different parameter values, as well as different parameters, since each object in the series can be the instantiation of a different class. The whole KBE model is dynamic by nature.

Wire harness design is a highly complex task in the aircraft design area and mainly finished by engineers manually. This is typically behind the nowadays requirement of the customer and market such as rapidly putting into the market, saving valuable time and money. This represents a huge challenge that taking advantage of KBE to automate the wire harness design process in order to save the money and time of aircraft design process. Base on KBE an automatic routing system for electric wire harness is presented in this paper. Firstly, the background of automatic wire harness design in aeronautics field is given. Secondly, the methodology and existed routing algorithms are introduced and analyzed, and then a suitable algorithm named A* for rule based routing method is

[1] Phd candidate, Delft University of Technology, Kluyverweg 1, 2629 HS Delft, the Netherlands.
[2] Professor, Delft University of Technology, Kluyverweg 1, 2629 HS Delft, the Netherlands, AIAA MDO TC Member.
[3] Phd candidate, Delft University of Technology, Kluyverweg 1, 2629 HS Delft, the Netherlands.

chosen. Thirdly, in order to comply with the complex aircraft specific routing environments, the selected algorithm is enhanced with so-called wall-attraction capabilities. So a modified algorithm, wall-attraction A* algorithm (W*), is introduced. Furthermore, through experiments, the influence factor in evaluation function of the W* algorithm is discussed and the parameter of evaluation function is optimized. After that, a case study presents the usage of above theory in a conceptual routing environment. Then a conclusion is given.

II. Problem background

It is generally believed that as a part of a whole aircraft design, wire harness design is a time-consuming job and has kept lots of manual processes. As one of the most complicated system in the whole aircraft, electric wiring harnesses are comprised of hundreds of cables and thousands of wires. It is necessary to provide electrical connectivity between all the mission and vehicle systems ensuring sufficient redundancy and reliability. The manual way for designing wire harnesses typically involves waiting for a hardware prototype of a whole or parts of an aircraft, manually measuring wire paths and processing numerous engineering change orders. The process consumes a lot of time and materials, and is often costly to expedite. Indeed design wire harnesses traditionally is high-stake and need to deal with huge data including weight of the wiring, the space requirements, electromagnetic interference, corrosion, resistance to environmental hazards and maintainability. It is generally know that the partial reason of delivery delay for Airbus A380 is the problem of wire harness. According to a variety of media, it's tangled in a bunch of electrical wire harnesses -- 530km of cables, 100,000 wires and 40,300 connectors, to be exact. So it is undoubted that wire harness design is worthy for automation and its complicated, repetitive, time consuming and rule-based nature make it a well-suited field to develop KBE design applications. The gains from automation should outweigh the costs of any new hardware and software which are used to develop the novel automation system. Moreover the case that this system can enable quality products to get to market faster will be accepted without question.

On the purpose of automatic designing the wire harness, all the multidisciplinary data which are involved in the current manual design process should be understood. Mechanical, electrical and manufacturing inputs must be considered. Accurate and detailed manufacturing outputs must then be generated automatically to prevent human error. This merger of data can be considered as a knowledge base, which understands all wiring elements and their relation to each other, as well as the design and manufacturing processes. Some of this knowledge comes from design handbooks published by different institutions. Also the formalized experience drawing from practiced engineers is vital for this knowledge base.

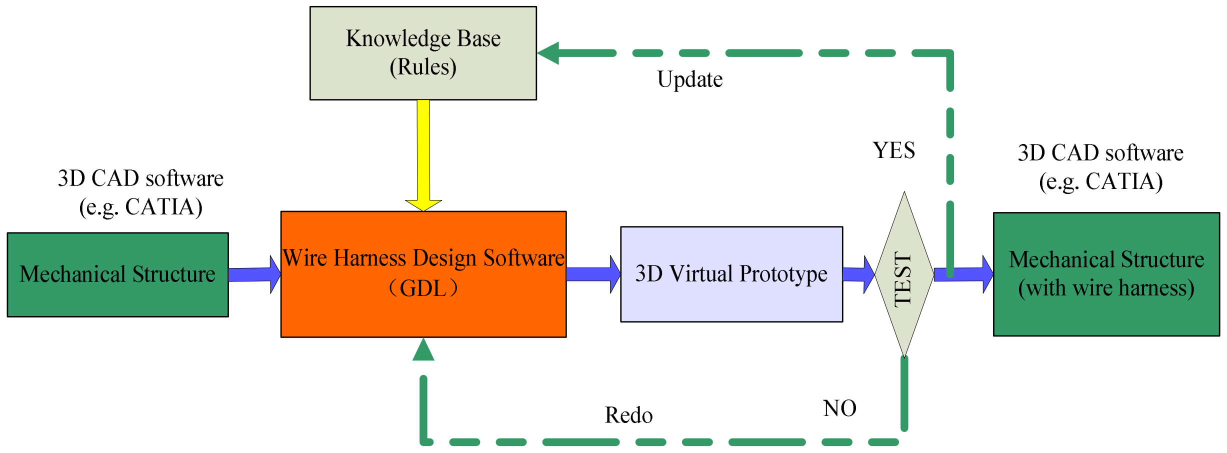


Figure 1. Flow chart of automatic wire harness design system.

Fig.1 shows a conceptual design of an automatic wire harness design system. Different from the traditional wire harness design, automatic wire harness design gets the mechanical model from 3D CAD software. The rules which are stored in the Knowledge Base and include the knowledge from handbooks and experienced engineers will be used to generate the wire harness automatically in GDL, which is an acronym for the General-purpose Declarative Language and will be explained in next section, programming environment. A virtual prototype of wire harness will be finished far in advance of any hardware prototype. Combining with rules, this virtual prototype can be used to determine wire bundles, paths and lengths, and to eliminate the process of manual measurements. The wire harness need redesign if it does not according with the requirement. Otherwise the information regarding to it will be uploaded to the Knowledge Base which will be updated correspondingly, and the 3D model of wire harness are going to transform to mechanical model in 3D CAD software for next procedure. This paper concentrates with the development of the automatic wire harness design software. The other parts of the whole system are going to be discussed on the next research stage.

III. The routing toolkit and algorithms

In order to fulfill the automatic wire harness design software, the automatic routing should be achieved firstly. As the core of the automatic wire harness design process, automatic routing in the aircraft environment is a time consuming and repetitive job. In order to automate this procedure, we should take advantage of computer and design excellent software to take over this arduous job from engineer’s hand. To complete this, we have twofold task, first choose a programming language, and second pick out a suitable algorithm.

A. Routing toolkit

As a KBE based problem, routing should be solved according KBE outline. GDL, a programming language derived from Common Lisp, represents a “next generation” KBE toolkit and provides all the benefits that matter from the legacy “pioneer” systems. It is a superset of ANSI Common Lisp, and consists mainly of automatic code expanding extensions to Common Lisp implemented in the form of macros. Also GDL is object-oriented, and it has all the features you would normally expect from an object-oriented language, such as

• Separation between the definition of an object and an instance of an object;

• High levels of data abstraction;

• The ability for one object to “inherit” from others;

• The ability to “use” an object without concern for its “under-the-hood” implementation.

As successful bidder software for automation wire harness design, GDL has the following specific property:

• Organizing and interrelating large amounts of information in ways not possible or not practical using conventional languages or conventional relational database technology alone;

• Evaluating many designs or engineering alternatives and performing various kinds of optimizations within specified design spaces;

• Capturing the procedures and rules used to solve repetitive tasks in engineering and other fields;

• Applying rules to achieve intermediate and final outputs, which may include virtual models of wireframe, surface, and solid geometric objects.

B. Discussion of path-finding

1. Inspiration

Finding a suitable path is common encountered in a lot of industry field, such as 2D/3D computer games, very large scale integration (VLSI) circuits, global positioning system (GPS) navigation, printed circuit boards (PCBs), and so on.

The highlight of path finding in video games is in-time, especially in real-time strategy games. Path finding in the context of video games concerns the way in which a moving entity finds a path around an obstacle according different rules. Path finding has grown in importance as games and their environments have become more complex. The engineers in this domain take lots of inspirations from Artificial Intelligence (AI). One of the greatest challenges in the design of realistic AI in computer games is agent movement. Path finding strategies are usually employed as the core of any AI movement system. Path finding strategies have the responsibility of finding a path from any coordinate in the game world to another. Systems such as this take in a starting point and a destination; they then find a series of points that together comprise a path to the destination. The game AI pathfinder usually employs some sort of pre-computed data structure to guide the movement. However, we all agree that path finding of video games is not a real three dimensional problem. Traditionally, detailed routing was considered as a 2-dimensional problem and the engineers always solve the routing problem through projection mapping. And the situation is similar in the filed of VLSI circuits design, for the number of layers was very small compared to the length and the width of the board. In spite of this, as the mature industry fields with development of decades, this also provides us a good origin to route the wire harness and electrical loom in 3D aircraft routing environment. Many relevant algorithms can be used directly or with improvements to fulfill the aerospace vehicle routing task of both global and detailed.

2. Routing algorithm analysis

There are lots of algorithms for path finding such as breadth-first, depth-first, best-first, A*, Hill climbing, Dijkstra's algorithm and so on. All of them are belong to graph searching category and can be classified roughly by non-information search and heuristic search. Starting from conveniences of programming and analyze, the breadth-first algorithm is chosen firstly.

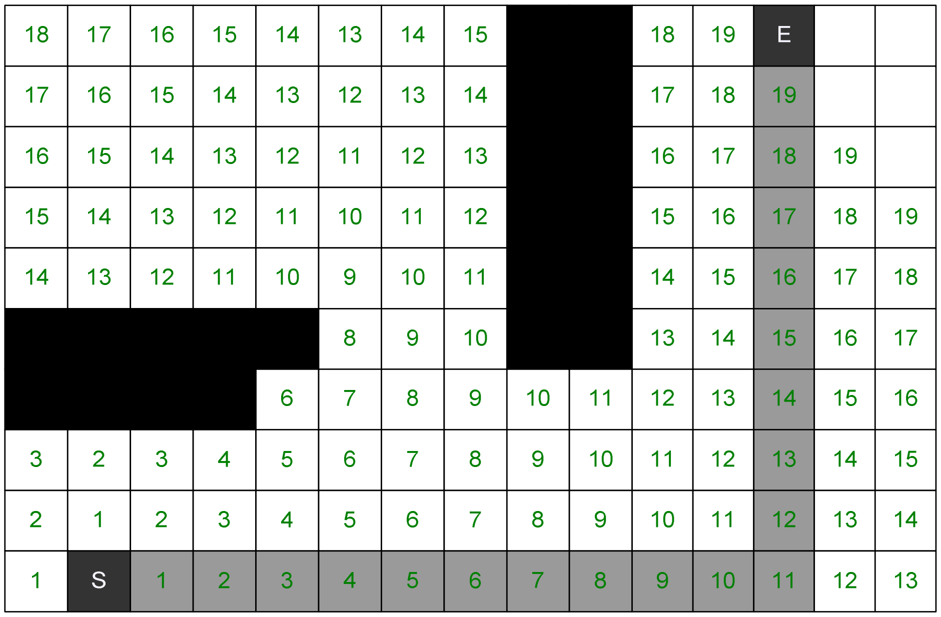


Figure 2. Breadth-first routing algorithm

Breadth-first search algorithm is a form of exhaustive search in which each of the nodes at a particular level is systematically explored before going to the next level, until the goal state is attained. It also can be considered as a flooding algorithm for distributing to every part of a connected network. For the routing environment is meshed and treated as combination of small mazes, the Breadth-first algorithm is implemented by 2D maze to simulate the routing procedure in aircraft surrounding. Fig 2 shows a simple 2D maze. During this process, the root i.e. the start node is expanded first, all the nodes which are bred by the root are expanded next, and then their successor will repeat this, and so on. Below shows the bread-first algotithm.

```
;;;******************************************
;;;Pseudocode of Breadth-first searching algorithm
;;;******************************************
Begin
  G:=G0, (G0=s), Open:=(s), Closed:=();
  loop: if Open= () then exit(fail);
          n:=first(Open);
          if Goal(n) then exit(success);
          remove(n) from Open, add (n) to Closed;
          expand(n) $\rightarrow$ {$m_i$}, add $m_i$ to G;
          add $m_j$ to Open;
   go loop;
End
```

Here ‘G’ means graph search, and‘s’ means start nodes. Open and Closed represent ‘open list’ and ‘close list’ separately. Firstly, the open list contains start node and only start node, and the closed list is empty. ‘n’ is the current node. The set { m_i} is child nodes of node n. m_j is the nodes which are not stored in open list and end list and it will be added to the open list. This process implements iteratively until reaching the end nodes or the open list become hollow. During this, the obstacle should be ignored, and treated as unreachable nodes. After that, the algorithm moves from the end node to start node in a backward direction. And the path which will be found if it is existed, lookes like the gray nodes in fig. 2. The weight of the maze is concerned with the distance from the current maze to the root. In this paper the weight is set as ‘1’. This means the weight of all the nodes at depth n in the maze are $n\times1$, and nodes in next depth are $(n+1)\times1$, and so on. With the breadth-first routing algorithm, the path between start node and end node is found easily. Intuitively, it is one of the shortest paths between them. Actually, breadth-first search always finds the shortest path to the end nodes for it always reach all the nodes at level n before searching on level n+1. And this is generally accepted in the Artificial Intelligence field.

However, as a non-information search algorithm, it needs flooding over nearly all the nodes in the searching space. This makes us have to take care of the time consumption and memory space requirement. The algorithm can be improved by different ways, for example, choosing the node farther from the center of the searching space as the start node, start searching from both start node and end node as shown in Fig. 3, and so on. These can save the consumption time and memory in a certain extent. But the inner property of this algorithm makes it still be an inefficient algorithm especially in a lager scale 3D routing environment.

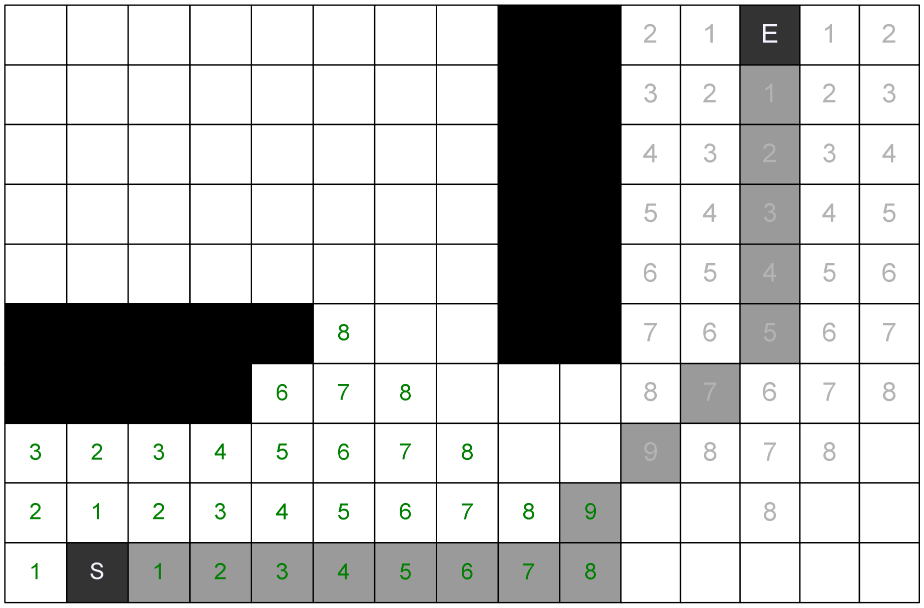


Figure 3. Improved breadth-first routing algorithm

In order to overcome the disadvantage of the non-information search algorithm, the heuristic searching algorithm is adopted. Heuristic searching makes use of the heuristic information which is contained by the problem itself to guide the searching procedures to reduce the searching scope and decrease the complexity of the searching process. As an outstanding one in heuristic searching method, A* is discussed here. A* described by Peter Hart, Nils Nilsson, and Bertram Raphael in 1968 derived from best-first search and can find the lowest cost path between the start and end nodes. Using A* we can easily find the balance point between finding the optimum solution through exhaustive searching and finding a satisfactory solution with lowest searching cost. Next, a detailed explanation will be given.

3. Routing algorithm in aircraft wire harness design

Using A* algorithm, because of taking advantage of the heuristic information, the ineffective ergodics processes are obviously decreased. And its properties, such as good performance of searching in maze, the method for calculation value of g, h and f, and the evolvability which means the new rules can be easily added by modifying the weight function, make it becomes the suitable algorithm for solving the wire harness routing problem in aircraft design.

In A* the evaluation function is $f(n)=g(n)+h(n)$. Here, n is the current node, and g(n) means the movement cost of moving from the source to the current node on the grid. The h(n) is a estimated movement cost of moving from current node on the grid to the destination. In order to improve the calculation efficiency, generally the g(n) and h(n) are round to integer. Below is peseudocode of A* searching algorithm.

```
;;;*********************************
;;;Pseudocode of A* searching algorithm
;;;*********************************
1 Begin
2   Give start, goal;
2   Openlist =(), Closedlist =();
3   g(start) =0, h(start) =heuristic-diatance(start, goal); f(start) = g(start) + h(start);
4   Move start to Openlist;
5   While(Openlist != nil)
6     {
7        Get node n which is first node in Openlist;
8        if(n == goal)
9           {break;}
10       Remove(n) from Openlist;
11       Add (n) to Closedlist;
12       foreach y in neighbor-nodes(n)
13          if ((y in closedlist) or (y is unwalkable))
14             continue;
15          eles if (y not in Openlist)
16            Add(y) to openlist;
17            Set father-node(y) =n;
18            g(y) = g(n) + dist_between(n,y);
19            h(y) =heuristic-diatance(y goal);
20            f(y) = g(y) + h(y);
21          eles            ;;;(y in Openlist)
22             if((g(n) + dist_between(n,y)) < g(y))
23                Set father-node(y) =n;
24                g(y) = g(n) + dist_between(n,y);
25                f(y) = g(y) + h(y);
26          sort Openlist from small to big according f value;
27    }
28  ;;;ergodic father-node to get optimized path
29  if(optimized-path-existed)
30     {
31      n = goal, Pathlist = (goal);
32      while(n != start)
33         {
34          n= Get-father-node(n);
35          Add(n) to Pathlist;
36         }
37     }
38 End
```

Fig. 4 shows the evaluation of f(n), g(n) and h(n). In this algorithm, f(n) is used to keep the searching on the rails and g(n) makes sure that the final path is optimum solution and h(n) is charge for finding the satisfactory solution

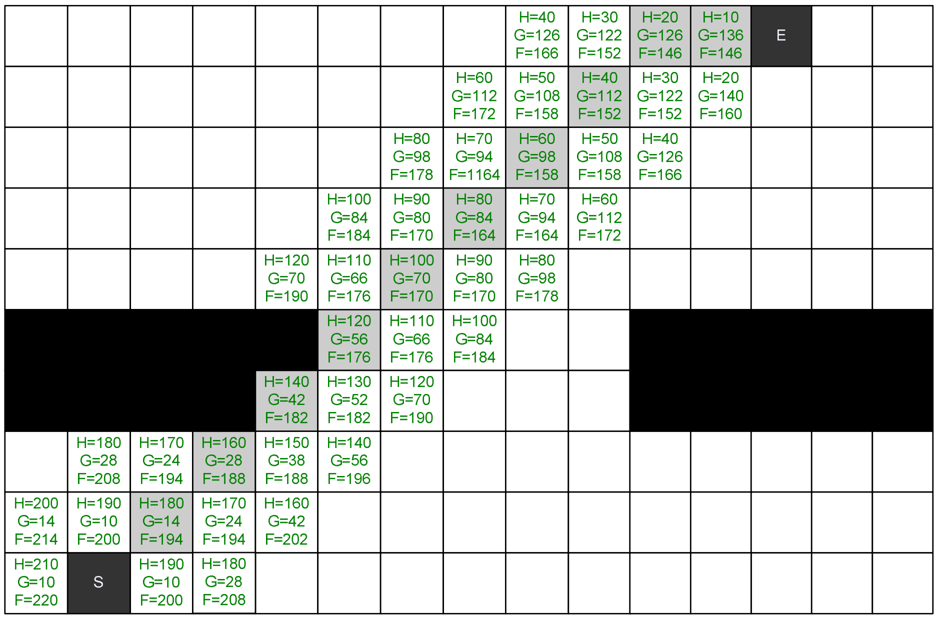


Figure 4. A* algorithm (with diagonal searching)

with minimal search effort.

Acturally, A* algorithm is already widely used in the field computer games, VLSI circuits, GPS. However as mentioned before, A* is only popular in 2 dimensional problem. Considering the future usage, it is undoubted that the A* algorithm should be extended to 3D application in order to comply the requirement of the automatic aircraft wire harness design. For the algorithm it is convenient to using the evaluation function to transform it from 2D to 3D. However when applying the A* algorithm in 3D, the complexity of analyze and calculation will increase explosively especially for the very large scale 3D routing environment with some obstacles. Because of limitation of the time and computer memory, some optimized methods will be proposed in next section to overcome this shackles of 3D automatic routing.

IV. Improvement and optimization

A. Wall-attraction A* algorithm

In the real aircraft, the wire harnesses is routing along the planes and surfaces such as deck and fuselage skin. This is reasonable for some of the room inside the aircraft be left for luggage and passages. Also this is convenient for the cables to be fixed and maintained in the future. Here this kind of surfaces in the planes such as deck, ribs and spars are named "wall". In order to make the wire harness be routed along the wall to according the requirement of the aircraft design, a so-called wall-attraction A* algorithm is proposed, i.e. W*. Using of W* algorithm can guarantee the wire harness be routed along the wall with the advantage of A* algorithm. At the same time, because the room far away from the wall is very rare for the wire harness to go through, W* algorithm can concentrate the nodes near the wall. This will obviously decrease the searching time with finding the optimized solution satisfactorily. Fig. 5 shows an applied case for W*. Intuitively, the routed cable in this case corresponds with the requirements i.e. following the wall and becoming the shortest path. When the requirements are modified e.g. some segment of the wire harness should be routed far away from the wall to avoid be damaged, as a rule-based algorithm W* can also modify its rules slightly to according the changed requirements.

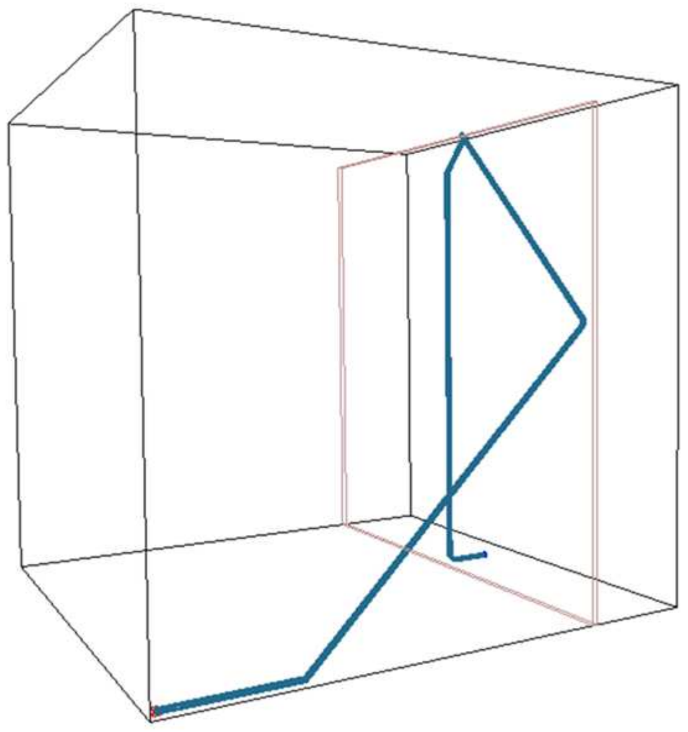
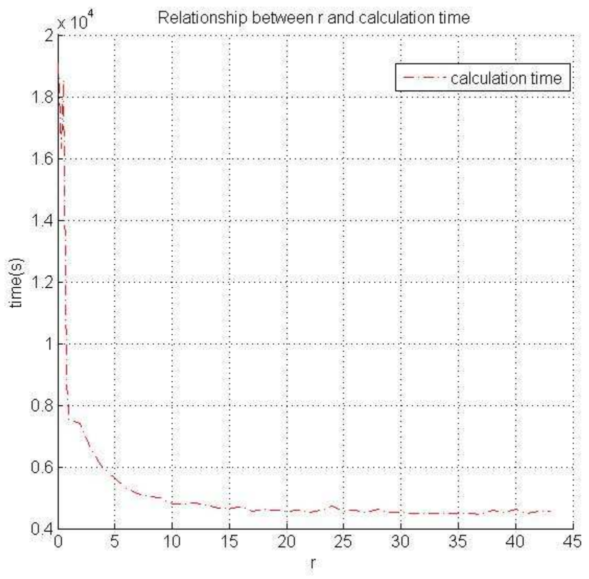


Figure 5. Applied case for W*

B. Optimization

As discussed in part B of charpter III, the evaluation function f(n)=g(n)+h(n) is the core of A* and W* algorithm. g(n) is the cost from source to current state and related to optimum solution and h(n) is charge for finding the satisfactory solution with minimal search effort. If we weight the value of g(n) i.e. emphasizing the cost information, this algorithm is going to find a optimum solution but takes a lot of time. On the contrary if the value of h(n) is weighted i.e. emphasizing the heuristic information, the searching speed will be increased, however the optimal solution can not be insured. In order to analyze the relationship between optimum solution and saving run time and try to find the most feasible solution in this specified situation, a improved evaluation function $f'(n)=u \cdot g(n)+w \cdot h(n)$ is proposed. w is the factor of h(n). If it becomes big, the weight of heuristic becomes big correspondingly. g(n) is similar to this. Here if the u and w equal to 1, the $f'(n)$ becomes $f(n)$. Using $f'(n)$ the property of A* and W* algorithm is reserved. Meanwhile the weight of cost and heuristic can be changed easily.

Herein the ratio of w and u i.e. r=w/u is used to evaluate the weight of different parts. If the heuristic information takes more share on the whole evaluation the value of r will increase, and vice verse. If r equal 0 the A* and W* algorithm will become the Dijkstra's algorithm. At the other extreme, if r is very high relative to g(n), then only heuristic information plays a role, and A* turns into Best-First-Search. Here for the convenience of data acquisition and analysis, u is identically equal 1. So r equals to w.



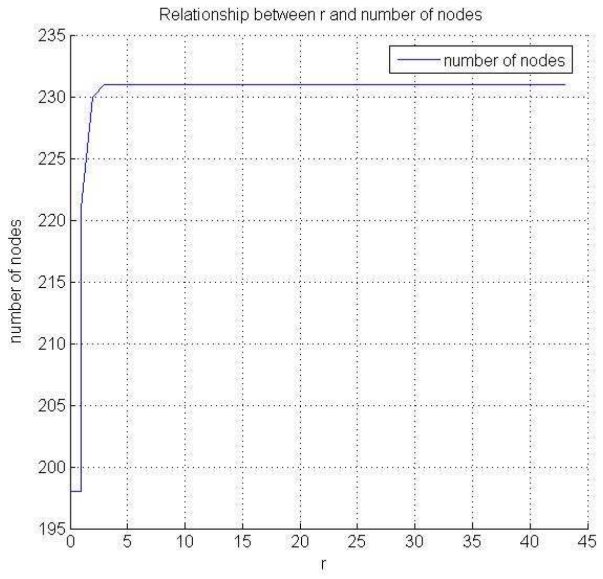


Figure 6. Relationship between r and calculation time **Figure 7. Relationship between r and number of nodes**

In order to find a suitable value of w for aircraft wire harness design, some experiments are implemented and analyzed. Based on this an automatic wire harness routing system will be built. Fig. 6 and Fig. 7 show the curve of experiment outcome. In Fig. 6, the x axis is r and the y axis is the run time of implementing the routing software. Here, the run time dramatically drop down until r equal 1. When r is bigger than 15 the run time is not decrease any more. It is meaningless to increase the r any more. In Fig. 7, the axis is r and y is the number of nodes. The number of the nodes means the summation of all the nodes between the source and destination through the final path wire harness. Using the number of the nodes, we can recognize whether or not the solution is optimal. During the harness design process, different requerments will be refered by our customers such as a feasible solution or a optimal solution. Through the two charts, it is very convenient to find the suitable r to accord the requirements between the admissibility of the solution and performance of searching process.

V. Results

In this section a case study of automatic path-finding in a conceptual routing environment is presented. In order to apply this 3D wire harness routing method, a concept geometry model of a wing is adopted as the routing environment. For using grided-based W* algorithm, the model is meshed into a set of small cubes. Fig. 8 gives the outline of the routing environment and the meshed cubes. These cubes contains not only the geometry information but also properties such as coming from ribs, spars, hydraulic pipe, and so on. These poperties will be used in the future combining with rules such as harness turing radius, Min/Max clearance distinguish, movement restriction and so on.

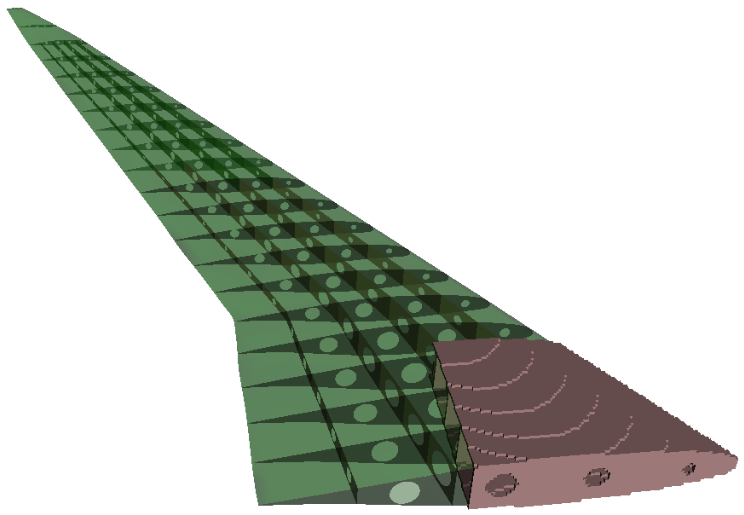


Figure 8. Geometry model and meshed cubes

At this momentment, only the geometry information of cubes is used for routing. Here the automatic wire haness routing software which is developed in GDL is used. Accroding the information of two given harness connectors, which are be translated to start and end point of W* algorithm, the program can automatically find the optimial path between two connectors. Fig.9 is a piece of harness routed in the wing. It comes out directly from GDL. This geometry model also be stored into neutral file such as STEP and IGES with property. This method facilitates commercial 3D modal software to inspect and modify them.

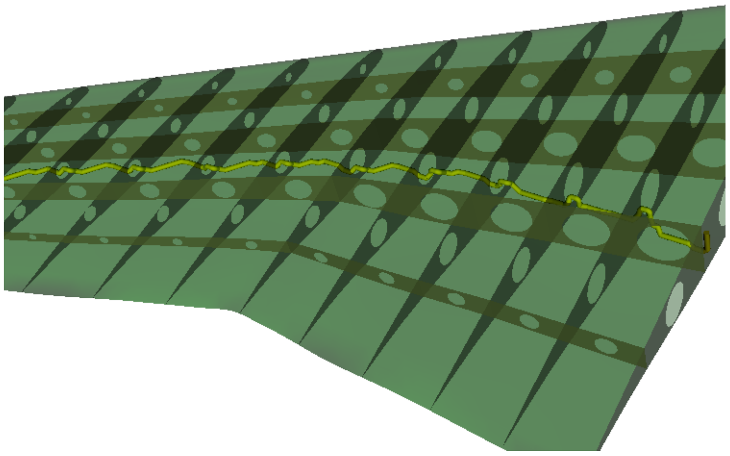


Figure 9. A piece of wire harness

VI. Conclusion

This paper has discussed the application of automatic wire harness design. Based on KBE, a automatic wire harness design system is proposed. As the core of this system, the automatic routing software is developed. The methods and algorithms which are able to be used in routing application are analyzed separately. Through this, the A* algorithm is selected for path-finding. When the A* is used to implement the routing application in very large scale 3D routing environment, the time consuming and memory limitation compel us to improve this reputed algorithm. In addition, according the requirements of the real case i.e. most of the wire harness should be routed along the wall in the aircraft environment. Based on these two situations, an improved A* algorithm (W*) is proposed. W* is the most suitable for automatic routing in the very large 3D routing environment.

Based on W*, some experiments are conducted. The results of the experiments represent the admissibility of W* algorithm. Also the evaluation function is be analysed graphically through the results of the experiments. And the concrete parameters of the evaluation function are represented for the specific routing problem in the aircraft design. A case study is finally presented, featuring a conceptual aircraft environment that validates the above theories. The wire harness routing application can connect any source and destination nodes satisfying according the relevant constraints meanwhile saves the valuable time and resources.

References

[1]S.W.G. van der Elst, M.J.L. van Tooren. DOMAIN SPECIFIC MODELING LANGUAGES TO SUPPORT MODEL-DRIVEN ENGINEERING OF AIRCRAFT SYSTEMS, The 26th Congress of International Council of the Aeronautical Sciences, Alaska, 2008

[2]C.Y. Lee. An Algorithm for Path Connections and Its Applications. IRE Transactions on Electronic Computers. September 1961.

[3]Dave Cooper, Gianfranco LaRocca. Knowledge-based Techniques for Developing Engineering Applications in the 21st Century. 7th AIAA Aviation Technology, Integration and Operations Conference. 18 - 20 September 2007, Belfast, Northern Ireland

[4]N.P.Padhy. Artificial intelligence and intelligent systems. India: Oxford University Press, 2005: 97.

[5]Jorge Nocedal, Stephen J. Wright. Numerical Optimization. Springer. Series in Operations Research. 1999

[6]David Szeszler. Combinatorial Algorithms in VLSI Routing. PhD Dissertation, 2005.

[7]R.Graham, H.McCabe, S.Sheridan. Neural Pathways for Real-Time Dynamic Computer Games. Sixth Irish Workshop on Computer Graphics (2005) Eurographics Irish Chapter

[8]Nicola Clark. The Airbus saga: Crossed wires and a multibillion-euro delay - Business - International Herald Tribune
URL: http://www.nytimes.com/2006/12/11/business/worldbusiness/11iht-airbus.3860198.html

[9]Patrick Lester. A* Pathfinding for Beginners http://www.gamedev.net/reference/articles/article2003.asp