

Automation of Electro-Hydraulic Routing Design using Hybrid Artificially-Intelligent Techniques

OLIVER Q. FAN, JOHN P. SHACKLETON, TATIANA KALGONOVA

School of engineering and design

Brunel University

Uxbridge Campus, Brunel University, Uxbridge, UB8 3PH

UNITED KINGDOM

<http://www.brunel.ac.uk>

Abstract: Traditional 'simple' genetic algorithms are theoretically capable of solving the 3-D spatial problem represented by hydraulic and electrical harness design. However, the size of the 'solution space' to be searched, for even the simplest of problems can represent a computational load sufficient to limit any practical application. This research proposes a 'key-point search' which when used prior to the GA can successfully reduce the size of the computational task. It does this by identifying those points in the physical three-dimensional space which are most likely to be useful in the final solution and producing an initial population of solutions from these points. This is shown to significantly reduce computation times to find valid solutions.

Key-Words: Artificial Intelligence, Automated routing, 3D pipe routing, Genetic Algorithms, Key point search, Industrial Design

1 Introduction

As Cross points out [1], brute force of computation actually can achieve performances that outmatch human performance in a number of significant areas of human cognitive endeavour. Moreover, in addition to doing things that human beings simply cannot do unaided, he notes that humans want machines to do things that are arduous and difficult for human beings to do. So rather than just emulate human abilities, artificially intelligent design support should be able to a) do things that designers cannot do, and b) relieve designers of unrewarding aspects of the design activity.

The problem that forms the basis of this study originated from an industrial partner who had the problem that whenever they make even minor design changes to the mechanical geometry of their products, they potentially have to re-route their hydraulic and electrical lines. This is difficult for them to do, because it is not easy route hundreds of different wires and pipes inside a complex 3D environment where each competes with the other for space. Moreover it is time consuming, and not a particularly rewarding job for designers, and if a method to automate this task could be developed, the potential savings in time and human resource could be immense.

The problem falls into a category of spatial configurations problems, and there have been some attempts to use evolutionary design techniques have been developed to solve other problems in this domain. Bentley has an evolutionary system to design different tables which is a combination of legs and flat surfaces. [2]. In our problem, not only spatial position of parts and their environment should be taken into take account, but ideally other constraint on the design solution must be take into account as well.

2 Tedious problems for designers

The aim is to develop a universal method to route pipes and wires through any given 3D space automatically and within a reasonable time. Additionally, this method should also be capable of expansion to encompass other characteristics such as pipe cost, standard inventories, etc.

A primitive problem, based on a simply defined physical space with a few obstacles, was used to develop and test the new algorithm. Each pipe or hose is pre-defined with their starting point, ending point and size along with other constrains such as the price of the hose, size of the hose etc. However

there is nothing inherent in either programme, or the underlying methods developed, to limit it to such primitive problems. One space and obstacle used in the primitive problem is shown in Figure 1, though to make it more difficult, in other test cases a second obstacle has been placed randomly in the space before programme is running. The primitive routing problem requires that two different hoses are routed through the space, Hose A from Point A to Point B, and Hose B from Point C to Point D. A satisfactory routing design requires that

- The hoses should neither intersect with the obstacles in, nor with each other.
- The Hose has to be close to the surface to get support.
- The cost of total combination will be as low as possible.

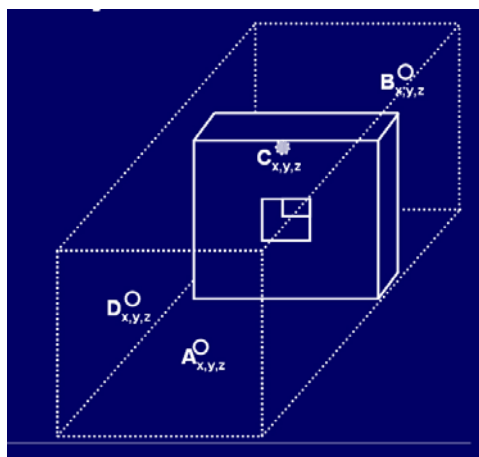


Fig1: Primitive problem space.

The space is 2000mm x 200mm x 160mm in size, the wall is 100mm x 160mm x 150mm and the hole is 20mm in radius. Hose A is 10mm in radius and Hose B is 5mm in radius. Price of hose A is 1.5 per unit, hose B is 2.5 per unit.

3 KPA + GA

3.1 The GA and its drawbacks

On first viewing, the problem seems not too difficult, since the 3D space is complex, and there only 2 hoses, though the issues to be solved around this are common to much more complicated hose combinations and more complex 3D environments, though with increasing space and more hoses, many more potential solutions are possible. As we

know any point 1 to point 2, the direct line is the shortest, therefore the cost of the hose will be lowest. But what if the combination of some hoses are more expensive than one hose and competes for the same space? Should this combination of hoses get priority? Huge numbers of possibilities exist for the solutions, hoses compete for space from each other, and also with the 3D environment. Moreover there can be other constraints to be applied to, such as in the primitive case, the overall cost of the route. The method to solve the problem will have the ability to search very large solution space and come up with good enough solutions. Because the given 3D environment is changeable, a direct formula will be difficult to find. So we are looking at indirect methods like evolutionary systems which do evolve to search large space and check the results until the results are satisfactory. GAs are one of the most successful evolutionary systems and were developed by Holland in an attempt to explain the adaptive process of natural systems and to design artificial systems based upon these natural systems [3]. GA has been widely used for a broad range of optimisation problems for years [4], and have been described as being a ‘search algorithm with some of the innovative flair of human search’ [5]. GAs are today renowned for their ability to tackle a huge variety of optimisation problems and for their consistent ability to provide excellent results, they are robust [6][7].

In this research, a GA system was used to try to solve the problem, and very quickly the disadvantage of GA systems was apparent. The efficiency problem; to evolve a reasonable solution with 2 hoses with only 1 constraint applied which is the length will normally take GA hours to run. Do a mathematical calculation, there are $2000 \times 200 \times 160$ possible turning points of a hose, and in GA we defined maximum turning points of a hose is 4, so there will be

$$(1): (2000 \times 200 \times 160)! / (2000 \times 200 \times 160 - 4)!$$

choices for one hose within GA. This is a huge number even for a modern computer. And that is only for one hose in a not very large 3D space; imagine if there are hundreds of them. GA does provide acceptable results eventually if you give it enough time. But the time GA consumed is not practical at all. Is there any way to speed up the GA? There are two obvious ways. First choice is to find faster computers to deal with the problem, the faster the better of course. More computing power means less calculation time. But this is not a practical way either since there is an always

physical limit for a computer and the problem can be easily big enough close to infinite. The second choice is to narrow down the search space. Properly all the humans are using this to solve a really time problem unconsciously. If a human designer look at the primitive problem, they will spot the area around the obstacle straight way. The trick is human designer will penalize unless space straight away, then for a simply case like the primitive problem, it not much route choice left. In simple GA, the whole space is possible location for the route and each location treated equally. For a more complex 3D space, human designer will take lot longer time to figure out the route(s) because even human designer can penalize useless route location quite easily, it still will have too much possible route location choices left. If there is a way that let the computer to penalize useless space in the given 3D space, the search space of GA will be lot smaller. And optimisation is one of the advantages of GA over human designers. If we can combine these two advantages together, the human's ability to penalize useless 3D space and computer's superb combination calculation, is it will be a new way to route hoses automatically in reasonable time?

3.2 Key Point Search (KPA)

3.2.1 Step of KPA

To improve the efficient of GA, we propose the idea of finding key points of the given 3D space which we called it Key Point Search (KPS).

The first step is to put the whole 3D space into a 3D bounding box and then divide the box into small 3D cubes. [Fig. 2] The 3D bounding box will provide the physical limits of the space; it can be extended if needed.

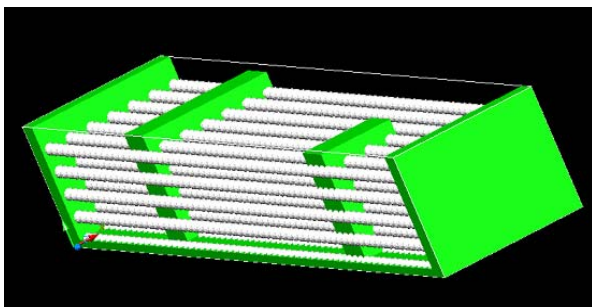


Fig 2. 3D environment in bounding box

The size of the cubes is changeable too; the suitable size will be the size of the biggest hoses or bigger that will guarantee any one of the hose will get through. In my program, the cube size is set to 20

by default. That means, the whole 3D space has been divided into 8000 small cubes.

The second step is to eliminate the cubes that intersect with the walls. The location of those cubes will not be needed since the intersection with the wall. After this process, there are 5956 cubes left. That means almost 25% of the cubes are intersected with the wall and those locations are not good for routing.

The third step is to eliminate the cubes that surrounded in all the directions by other cubes, in another words, those cubes are in the middle of the air with connections to all directions. Why we have to eliminate those locations? It's because if they are surrounded by other cubes, they can accessed by all the possible directions and can be represented by other cubes. For example in X direction, if there're 3 cub's next to each other, n1, n2, n3. So in the direction of X, if we can reach n1 and n3, then we can definitely reach n2 since n1, n2, n3 are connected next to each other. Similarly, if there is a string of points n1, n2, n3 ... nx, nx+1 which are connected next to each other, if we can connect to n1 and nx+1, then we can reach any cubes in between n1 and nx+1 since they are all connected next to each other. After this process, there are 3290 cubes left.

The fourth step is to eliminate the cubes that surrounded by other cubes in one plane, either X, Y, or Z plane. If a cube (red in Fig. 3) is surrounded by other 4 cubes in a plane that means either one of those 4 cubes that can access the middle cube and the middle cube can be represent by surround 4 cubes, so the middle cube can be eliminated.

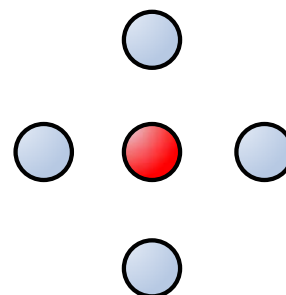


Fig. 3 Points connections in one plane

In the other way of explanation, only those cubes that have the direct connectivity to further space will be saved into next process, the rest cubes will be eliminated because the can be represented by the

cubes selected. After this process, only 79 cubes left.

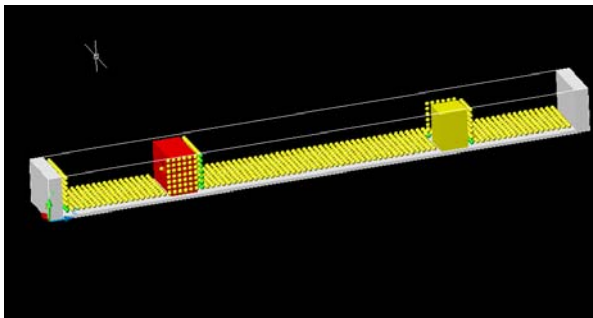


Fig. 4 KPA level 3/4

In the picture above [Fig. 4], the yellow dots (light grey if printed B/W) are the cubes that left from step three; the green dots (darker grey if printed in B/W) are the cubes that left from step four in the primitive problem.

The last step is to eliminate the cubes that have connection with other two cubes in one direction, either X axis, Y axis or Z axis. Same reason, keep the cubes that have the direct connectivity to further space and eliminate the ones can be represented by those selected cubes. By now, there are 16 cubes left, and those 16 points are the key points we was looking for and the locations are saved for further processing.

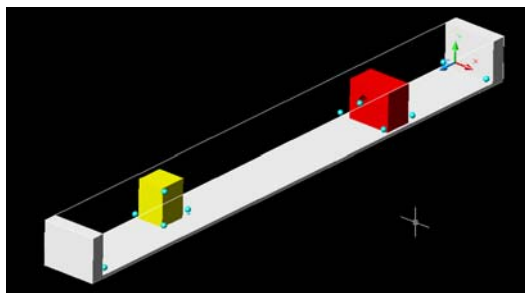


Fig. 5 KPA Level 5

The blue dots (light grey if this paper printed in B/W) are the key points of the given space found by KPS.

3.2.2 KPA Matrix

After key points have been found in the given space, a matrix will be created to holds all the information that needed.

First layer of the matrix is the connectivity between key points and maximum volume that between 2 key points. Since 15 points have been found in the primitive problem, the matrix will look like:

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	
0	x																
1		x	16	16	16							16	16				
2			x	16	16							16	16				
3				x	16			16	16					16	16		
4					x			16	16					16			
5						x	16		16	16						16	
6							x		16	16						16	
7								x	16	16	16				16	16	
8									x	16	16	16			16	16	
9										x	16				16	16	
10											x				16	16	
11												x	16				
12													x				
13														x	16		
14																x	
15																	x

Table. 1 KPA Matrix

Every two different key points will be tested to see if they have direct connectivity by connecting these two key points with a hose same size of cubes. If the hose created not intersect with any of the wall that means these two key points have direct connectivity. Otherwise, they don't. If two key points have direct connectivity, the maximum pass through volume will be checked next. The maximum pass-through volume is to see how many hoses can be route though these two key points. Of course, if the maximum pass through volume is bigger than all the hoses that mean all the hoses can be route though these two key points.

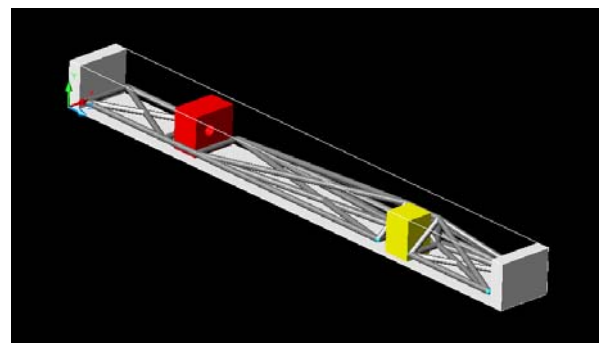


Fig. 7 KPA space frame

In the primitive problem, Hose A is size of 10 and Hose B is size of 5, so if the maximum pass-through volume of two key points is bigger than total size of Hose A and Hose B which is 15, that means both hoses can be route though these two key points in the same time.

The way to check maximum pass-through volume of two key points is to connect a hose with increasing size till the hose intersected with the wall or obstacle. Key points at this stage will be allowed to move in small steps to make sure the maximum or

near maximum pass-through is reached. Fig. 7 presents a full connectivity frame formed by direct connections between key points.

From the first layer of the matrix[Table.1], all the positive numbers in the matrix shows connectivity between key points also the maximum pass-through volume, for example key points 1 and 2 have direct connectivity, and both hoses can be route though key point 1 and key points 2.

The second layer of the matrix is the length between 2 key points. Since length is a very important factor of the problem and it will be referred so many times in the calculation of the routing process, it's a better way to save it so can be referred later on in the process. It saves the time of every time when length of the hose needed in the calculation.

3.2.3 Select Routes

There's one more step from key point matrix to final stage that can be used by GA which is picking the routes for hoses. Even from 15 key points, there are still a big number of possibilities of route locations. This is just in the primitive problem, in the real case, there will be lot more key points and the possibilities of the route will be even bigger. But the most interesting thing is those key points are not be picked totally randomly like GA will do since they do have connectivity as relationships between each two of them.

Here is the example of connectivity of primitive problem:

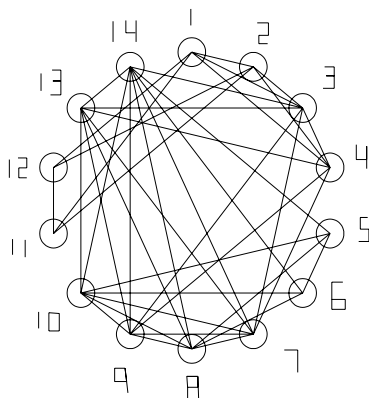


Fig. 8 Key Point Relationship

When the program starts to route a hose though the space, it checks the starting point and end point along with the size of the hose. First of all, check how many key points will have direct connectivity with the starting point of the hose. Then do the

same for the ending point. For example, starting point of Hose A has direct connectivity with key points 2, 3, and 12, and ending point of Hose A has direct connectivity with key points 13 and 14. Then program starts with the starting point, check if all those 3 key points has direct connectivity will have enough volume to pass Hose A which is size 10 and it will. Program will then randomly pick one of the key points from those 3 candidates. After that, it will check if the picked key point has direct connectivity to ending point. If yes, then a route has been created, if not, then start to find the next level of key points which is connected to the key points program picked. For example, program pick 12 randomly, then program will check the Key Point Matrix to find out key points 1,2,11, since key points 2 is one of the choice in the previous level, only key point 1 and 11 will be available for program to pick with. If there's nothing left to choose from next level, program will then pick a different candidate to start with from current level of the key points. If there's not a possible route found after all the possibilities, which means there is not a possible route for this hose to route though the give 3D space.

Once a route has been found, the volume will be subtracted from the Key Point Search Matrix before the second hose begin to route. So hoses and wires will not compete for space from each other.

A group of selected designs will then be put into the GA pool to form the first generation of the GA.

4 Conclusions

The results from the combination of KPA and GA are significant.

With GA only, maximum turning points of a hose is set to 4 and the very basic constrain is the hoses not intersect with the obstacle or themselves.

Max Turing Point	Number of Design	Time
4	400	643 mins
4	400	655 mins
4	400	631 mins

Table. 2 GA performance

With KPA + GA, turning point of a hose is generate automatically depends on the route that KPA matrix choose. And length(cost) constrain took into account as well.

KPA	GA	Total
6 mins	< 1 min	7 mins
6 mins	<1 min	7 mins
6 mins	<1 min	7 mins

Table. 3 KPA+GA performance

Both tests were running in the same machine which is a dual Xeon 2.66GHz, 2GB Ram with windows SP2.

Key point search actually transform the character of a given space to a mathematical matrix to largely reduced the search space for evolutionary systems like GA. KPA can very much improve the performance of GA and also it's not bonded to GA only. It's also flexible to combine with other evolutionary systems. I am not saying KPA is the best or the only techniques that can solve this kind of particular problems, but at least it is a new approach to automate the pipe/wire routing in a practical time.

References:

[1] Nigel Cross, Natural intelligence in design, *Design Studies*, Vol.20, No.1, 1998, pp. 25-39

[2] Peter J. Bentley & David W. Corne, *Creative Evolutionary System*, Academic Press, 2002

[3] Peter J. Bentley, *Evolutionary Design by Computers*, Morgan Kaufmann, 1999

[4] Holland, J.H, *Genetic Algorithms*. Scientific American, 1992, pp. 66-72

[5] Goldberg, D. E. *Genetic Algorithms in Search, Optimization and Machine Learning*. Addison-Wesley. 1989

[6] Davis, L. *The handbook of genetic algorithms*. Van Nostrand Reinhold, New York 1991

[7] Fogel, D. B. *Asymptotic Convergence Properties of Genetic Algorithms and Evolutionary programming: analysis and experiments*. J. of cybernetics and system 25, Taylor and Francis Pub. Pp.389-407