Fuzzy-enhanced path-finding algorithm for AGV roadmaps

M. Sc. Sarah Uttendorf[1] , Prof. Dr. Ludger Overmeyer[2]

[1] M.Sc. Sarah Uttendorf, IPH- Institut für Integrierte Produktion Hannover, Hollerithallee 6
30419 Hannover, Germany, uttendorf@iph-hannover.de
[2]Prof. Ludger Overmeyer, IPH- Institut für Integrierte Produktion Hannover,
overmeyer@iph-hannover.de

Abstract

Path-finding algorithms (PFA) are successfully used to find the optimal path between two locations. Good results are obtained if they are used in scenarios where the entire environment can be described mathematically. Production environments of automated guided vehicles (AGVs) are not one of those. PFA find solutions that are mathematically correct but miss human expertise that would dismiss solutions of the algorithm that aren't applicable to a real production layout. This paper presents a hybrid algorithm consisting of an A* algorithm, and a fuzzy logic control in order to generate a fuzzy-enhanced A* algorithm (FEA*) that produces efficient and applicable road maps for AGVs. First computational results are shown.

Keywords: Path- finding algorithm, Fuzzy-logic, expert system, AGV, road maps

1. Introduction

With the increasing industrial automation of production processes, the use of automated guided vehicles (AGVs) in the manufacturing environment is increasing as well. Not only the tasks of the AGVs become more and more complex [01] but also the environment they are operating in is increasing in size and complexity. As a result the task of designing a feasible road map for AGVs becomes more demanding. The basis for the process of designing a roadmap is e.g. the transport matrix, the kinematic properties of the AGV or the layout of the production environment. These and more factors can be implemented in a traditional optimization problem in order to use it for classical PFA. The aim of those is to find an optimal roadmap in a given production environment. *Figure 1* shows a test environment of a track-guided AGV system.

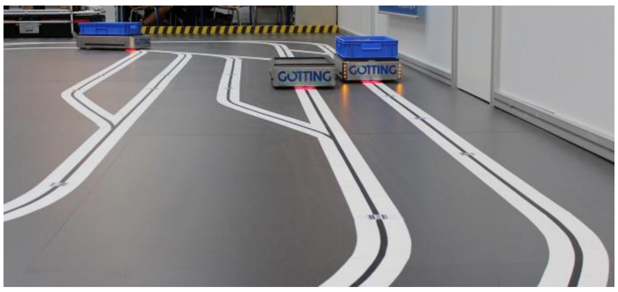


Figure 1: Test environment of a track- guided AGV system

The multiobjective nature of the optimization problem is a challenge as well because the objectives and their priorities change depending on the particular AGV system. For example in some cases it is more important to minimize the path lengths, in other cases the priority is to combine as many paths as possible in order to save space. An efficient algorithm has to take the variation of these priorities into account and should be able to be easily modified if the circumstances of the AGV system are changing.

Even though it would be possible to generate specialized multiobjective functions for the problem of finding the mathematically best AGV roadmap in a given layout, the PFA would still generate roadmaps that aren't applicable to a real-world production layout. Some real world problems can be easier described in a linguistic way, rather than a mathematically sense. For example the problem: "*If there are areas with many crossings of paths and the alternative route won't be too long, than take the alternative.*" It would be very difficult to actually implement this statement in a mathematical model. A linguistic phrase like "many" or "too long" is too vague to assign them an exact number but it is possible to assign them a value range. This way the information can be made useable in the process of finding an optimal roadmap.

Current planning processes of AGVs rely heavily on manual work. System planers have gained experience on how to design a feasible roadmap without the use of algorithms. Besides the extensive time consuming manual work that is necessary, this approach is only feasible in small production environments since the user has to define every single way. With an increasing transport matrix and higher dependencies with the stations it gets more and more complicated to get optimal paths layouts without the use of an optimization tool.

In order to obtain a feasible roadmap that is not lacking human expertise this paper presents a hybrid algorithm that combines established modified A* (**mod A***) path finding algorithm with a fuzzy controller. As a result road maps are generated that follow both a mathematically optimization process and also get input from human planning.

This paper is divided into the following sections:

Section 2 introduces the current state of technology. In section 3 the overall workflow of the FEA* is explained an in Section 4 the modified A* is discussed. Some modification resulting from the restrictions of the AGV environment have been implemented.
Section 5 deals with the fuzzy-logic controller and the combination of the fuzzy logic and the modified A*. All of the three main components, the fuzzyfication, the rule& data base and the defuzzyfication principle are discussed. In Section 6 a discussion of the results of the FEA* takes place and section 7 gives ideas for further research.

2. State of technology

Different systems already exist in order to obtain optimal roadmaps. [02] already implemented a modified A* algorithm for path planning for a mobile robot. In this approach several modifications have been imposed for the A*. The main focus here is the computational time and the path optimality. The performances of the different modifications of the algorithm were compared with the computational time, the length of the path, number of examined cells and the symmetry of the examined environment. The proposed algorithm is able to find shortest paths quickly but wouldn't be suitable in AGV environments, because too much free space is being consumed by the paths and important factors like the number of direction changes or the consideration of alternative ways are being neglected.

[03] introduces a light-assisted A* algorithm for path planning. He focuses on avoiding dynamic obstacles and expands fewer nodes than the traditional A*. The production environment is seen as a dark room where obstacles are highlighted and can therefore be foreseen in the path planning process of the A*. For each node of the map a brightness score is calculated that is later on included in the cost-function.

What all these papers have in common is that they are either focusing on finding the shortest path or increasing the time efficiency in finding one. In case of outlying an efficient roadmap for AGVs a short computation time and short paths are important as well, but generating an algorithm that is able to construct feasible and applicable roadmaps is from bigger importance.

The idea of building a hybrid with the help of Fuzzy Logic is already used with different integrated components. [04] combines two Fuzzy Inference Systems (FIS) where each FIS has a different task. The first FIS is responsible for the tracking and the second FIS for the reaction task of autonomous mobile robots. The outputs of both FIS are integrated through a weighted FIS.
[05] also uses a Fuzzy Logic controller to enable autonomous robot navigation. His hybrid system includes frameworks for goal determination, preprocessing, behavior design, behavior arbitration and finally a command fusion. The obtained results are good but computation time was not one of the concerns.

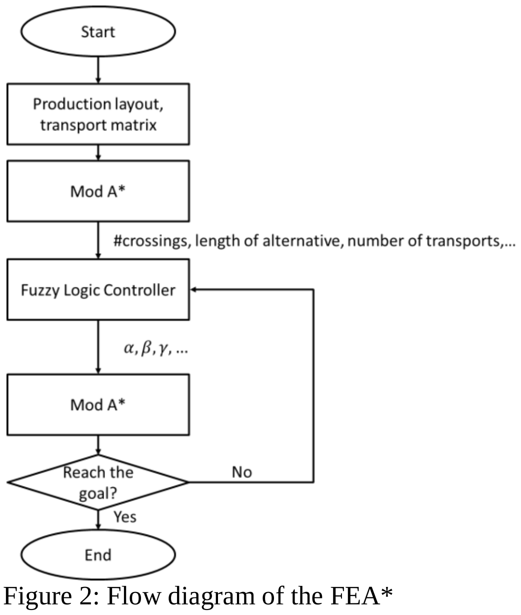
The approach of this paper is tough to combine a fast heuristic approach (the mod A*) with the Fuzzy-logic in order to generate feasible roadmaps in a short period of time. Computational effort is meant to be minimized.

3. The Fuzzy-enhanced A* (FEA*)

Figure 2 shows the overall workflow of the FEA*. In the beginning all the necessary information, e.g. the production layout or the transport matrix, are loaded into the system. The mod A* is run on these information. An initialed AGV roadmap is generated. The focus is primarily on finding the shortest path from e.g. station s_1 to station s_2 with some basic AGV forced constraints, such as a certain security distance from the wall or from existing paths to prevent collisions.
After the first run of the A* important characteristics of the initial road map are transmitted to the Fuzzy-Logic Controller (**FLC**). With the input of the mod A* the FLC is able to make "intelligent reasoning" and gives its output in the form of the parameters $\alpha, \beta, \gamma, \ldots$ back to the mod A*. These parameters are directly used in the cost function of mod A*.

After the second run of mod A* an inquiry is started if the desired goal for the road map is reached. This inquiry can be based on a predetermined number of runs of the A* or on other criteria, e.g. a desired length of the road map or shortest paths between stations. So far the algorithm stops after a certain count on loops. That way the user can determine the optimal level of trade between construction time and efficiency of the road map.

Afterwards the algorithm ends and the generated roadmap is transferred to a .csv file.


Start
Production layout, transport matrix
Mod A*
#crossings, length of alternative, number of transports,...
Fuzzy Logic Controller
$\alpha, \beta, \gamma, \ldots$
Mod A*
Reach the goal?
No
Yes
End
Figure 2: Flow diagram of the FEA*

4. Implementation of a modified A* (mod A*) algorithm

As *figure 2* shows a modified version of the traditional A* is used. Reason behind this is, that some changes (e. g. reduction of diagonal paths) to the A* can be made without the use of a FLC in order to make the generated road map more realistic.

4.1. The traditional A* algorithm

The A* algorithm was developed in 1968 by Peter Hart, Nils J. Nilsson and Bertram Raphael. The algorithm uses a heuristic in order to minimize computation time. The A* is based on the search strategy of finding a path with minimum costs ([06], [07]). [07] shows that the A* is suitable for solving origin-to-destination shortest path problems and is quicker than the commonly used Dijkstra algorithm.

In order to use the A* in an AGV environment, the given production layout is divided into an equally spaced grid where each cell resembles an area in the production layout.

The algorithm finds the cheapest paths between two points or in the AGV scenario between two stations s_1 and s_2 through the use of a cost function. The cost function can be expressed as:

$$f(x) = g(x) + h(x) \qquad (1)$$

where h(x) is a heuristic used to estimate the costs from the current cell x to the goal cell s_2. The estimated costs must never exceed the actual costs of the path. For the implemented A* in this paper the Euclidean distance is used as a heuristic:

$$h(x) = \sqrt{(x_x - x_{s2})^2 + (y_x - y_{s2})^2} \qquad (2)$$

g(x) ressembles the hitherto existing costs from the start station to the current point x.

$$g(x) = \begin{cases} x.pre.costs + \sqrt{2}c(x) & for\ diagonal\ cells \\ x.pre.costs + c(x) & else \end{cases} \qquad (3)$$

c(x) resembles the cost of the specific cell x. This cost depends on the weighting of the cell. Cells with no obstacles are assigned a value of 1, cells with some kind of constraints, e.g. crossing paths, are assigned a higher value, e.g. 3 and obstacles are assigned a value of 9999. In this case

$$c(x) = const. \qquad (4)$$

is assumed.

The A* process can be described with the following Pseudo-Code [08]:

1. Create a search graph G, consisting solely of the start node, s_1. Put s_1 on a list called OPEN.
2. Create a list called CLOSED that is initially empty.
3. If OPEN is empty, exit with failure.
4. Select the first node on OPEN, remove it from OPEN, and put it on CLOSED. Call this node x.
5. If x is a goal node, exit successfully with the solution obtained by tracing a path along the pointers from x to s_1 in G. (The pointers define a search tree and are established in Step 7.)
6. Expand node x, generating the set M, of its successors that are not already ancestors of x in G. Install these members of M as successors of x in G.
7. Establish a pointer to x from each of those members of M that were not already in G (i.e., not already on either OPEN or CLOSED). Add these members of M to OPEN. For each member, m, of M that was already on OPEN or CLOSED, redirect its pointer to x if the best path to m found so far is through x. For each member of M already on CLOSED, redirect the pointers of each of its descendants in G so that they point backward along the best paths found so far to these descendants.
8. Reorder the list OPEN in order of increasing f values. (Ties among minimal f values are resolved in favor of the deepest node in the search tree.)
9. Go to Step 3.

Important to note is that each visited cell has a pointer to its predecessor. By connecting all the predecessors a path is generated. The pointers will become important for the modification of the A* to the AGV setting. *Table 1* shows the transport matrix that was used for testing the different stages of the algorithm. Figure 2 shows the implemented A* algorithm run on an example factory layout.

station/ station	1	2	3	4	5	6
1	-	15	0	7	0	15
2	0	-	0	0	0	0
3	0	0	-	0	7	0
4	0	0	0	-	0	0
5	0	0	0	0	-	0
6	0	0	0	0	0	-

Table 1: Transport matrix for the example factory layout

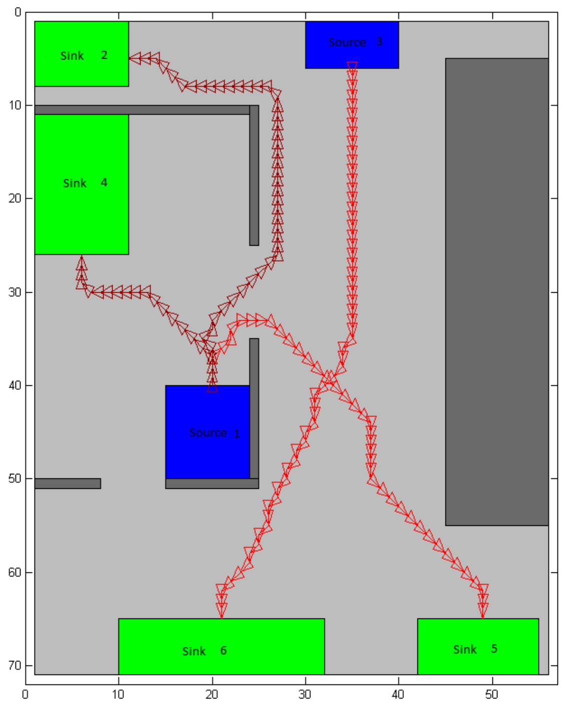


Figure 3: results of the A* algorithm on the example layout

4.2. The modified A* algorithm

As can be seen in *Figure 3* the generated road map from the A* is not feasible for real production environments. Two major problems of the road map are that the distances between the paths are in a lot of the cases too narrow in order to prevent collisions between the AGVs and that there are too many paths going diagonally through the layout and therefore unnecessarily decreasing free space.

These problems can already be overcome without using any fuzzy-logic because it is easy to define them in a mathematical way. Hence a modified A* (**mod A***) has been implemented and serves as a basis for the FEA*.

As an example the paths going diagonally through the production layout have been minimized through a closer look at the already selected cells for the path. Every selected cell has a pointer implemented that points to the previous cell. If the expected future direction equals the current direction, the previous cell will have a pointer pointing in the same direction. If the pointers have the same direction the predecessor cell will be assigned a lower cost in the cost-function. The mod A* is going to prefer the cell with the lower cost for the path. If the direction of the pointers doesn't match the costs will be set higher. The results of this modification and a distance modification for obstacles in the layout can be seen in *Figure 4*.

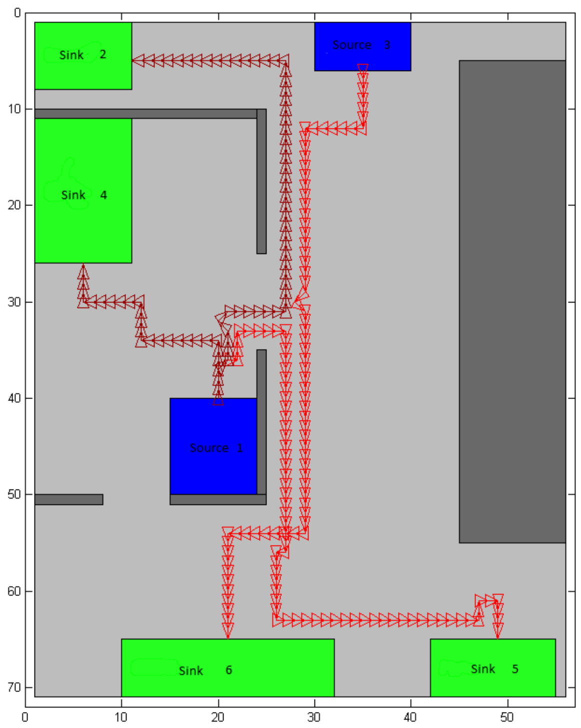


Figure 4: Generated road map from the modified A*

It can be seen that the amount of diagonal paths has been minimized to a minimum and a distance from the paths to obstacles has been achieved.

Even though this modified A* algorithm generates a road map that is more realistic than the road map of the A*, a lot of the objectives for an optimal AGV roadmap are still not met. These objectives, e.g. maximizing free space or combining roads if feasible, can't be realized merely by a code modification of the A*. At this point the fuzzy logic comes into effect.

5. Combining the mod A* with a Fuzzy-Logic Controller (FLC)

As described before the A* focuses on minimizing the cost function (1). g(x) included c(x) for assigning costs(4).
c(x) is assumed to be a constant numerical value in the traditional sense of the A* algorithm.

In the FEA* the FLC interacts with c(x). c(x) isn't constant anymore but follows a function like that:

$$c(x) = \alpha + \beta + \gamma + \delta + \cdots \tag{5}$$

The FLC assigns numerical values to the parameters $\alpha, \beta, \gamma, \ldots$ and therefore influence the cost-function f(x) of the mod A*.

Through the use of the FLC it is possible to include the human reasoning into the mod A* and to generate roadmaps that follow a mathematical optimization process and human reasoning.

5.1 Implementation of a fuzzy-enhanced A*

The fuzzy logic helps to include the knowledge of the AGV system planer into the overall optimization problem of finding an optimal path for AGVs. In the beginning of the paper the following example was introduced:
“If there are areas with many crossings of paths and the alternative route won’t be too long, than take the alternative.”

In a mathematical planning problem this linguistic statement couldn’t be used. Through the use of fuzzy-logic it is now possible to include this information. A corresponding rule could be:

“*IF many crossings AND length_AlternativeRoute medium THEN alternativeRoute strongly suggested*”

A set of rules was conducted through interviews with AGV system planers. These rules have been implemented in MatLab and a Fuzzy-Logic panel is used to easily change rules and properties of them. This allows the user to still have influence on the planning process and the path planning algorithm becomes more transparent.

[09] implements a fuzzy control system for collision avoidance of AGVs. He separates the fuzzy control system into four necessary components. Analog to that the FLC for this paper can be separated as followed:

1) The **Fuzzyfication Interface**: The mod A* algorithm delivers relevant numerical values to the interface. In the interface these crisp values are fuzzyfied based on their prior assigned membership functions of the linguistic sets. These linguistic sets are determined based on the knowledge and experience of the system planers.
2) The **Knowledge base**: The knowledge base contains all the rules in the form of IF-THEN commands. These rules are conducted from interviews with system planers and the comparison of results from the mod A* and real AGV road maps. Analog to [10] redundant rules have been removed and weights have been assigned to the rules.
3) The **Inference Engine**: the inference engine operates the actual rules. Based on the selected rule composition method in the inference system the outputs vary. The max-min composition method was chosen because of time efficiency and simplicity.
4) The **Defuzzyfication Interface**: This interface converts the fuzzy outputs of the inference engine into crisp values that can be used in a reapplied mod A* in the form of the parameters $\alpha, \beta, \gamma, \ldots$. Again different methods for defuzzyfication are possible, but the center of area (COA) is the chosen method because of short computation time. The COA method calculates the output as following [11], with x_{min} and x_{max} representing the range of the linguistic variable and m(x) being the membership function:

$$CoA = \frac{\int_{x\_min}^{x\_max} m(x) * x \, dx}{\int_{x\_min}^{x\_max} m(x) \, dx} \qquad (6)$$

Figure 5 shows the idea behind the FLC on the mentioned example.

In *step 1* the initial generated road map of the A* is analyzed and important parameters, e. g. number of path crossings or the length of alternative ways are transmitted to the fuzzy logic.

Afterwards in *step 2* the fuzzyfication of the parameters takes place. Depending on the predefined membership functions for the input, the parameters become translated into a linguistic term. For example 5 crossings in one area would be classified as 75% “many crossings” and 25% “medium amount of crossings”. The crisp parameters of the mod A* are now fuzzyfied and can be used in step 3.

In *step 3* the rules of the fuzzy controller are applied and used to generate a fuzzy output. All the rules are fulfilled with a certain percentage, ranging from 0% to 100% depending on the membership function in the fuzzyfication process.

In step 4 the membership functions of the rule base are used to defuzzyfy the output of the decision logic and to generate crisp values. In this step the different levels of fulfillments of the rules are taking into consideration and with the use of the COA method a crisp value is generated. These crisp values are saved in the parameters $\alpha, \beta, \gamma, \ldots$ and are used in the next iteration of the mod A*.
In the final step 5 the mod A* is run again with the updated cost function. In the given example a high amount of crossings and a short alternative route would lead to an additional path in the production layout.

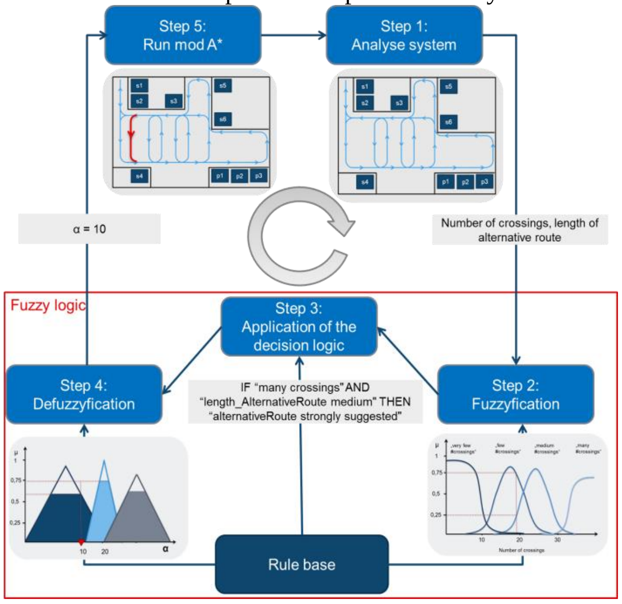


Figure 5: Interaction of the FLC and the mod A*

6. Discussion of the results & further research

Figure 6 shows the generated road map of the FEA*. The road map shows some significant improvements compared to *figure 4*. Compared to road map generated by the mod A* the amount of crossings has been decreased to a minimum. This way the likelihood of collisions of AGVs or traffic holdups is limited. In a lot of AGV systems a higher amount of crossings means also higher costs of the system since more transponders are necessary to guide the vehicles correctly and to avoid collisions.

Another substantial improvement of the road network is the merging of different roads going into the same direction. The FLC has several rules implemented that give inputs to the mod A* whether a merging of the road is beneficial or not. Important criteria for that decision are for example the length of alternative paths, the amount of goods that have to be handled in a certain amount of time or the future direction of the paths.

Compared to *figure 4* the amount of free space in the production layout has increased in size. This is due to the minimization of the change of directions of paths, the merging of paths and the consideration of safety distances.

The current state of the algorithm is still a very mathematical one and several important improvements have to be implemented to generate directly applicable roadmaps. So far the turnarounds are too steep and an inclusion of the different AGV specific curve radiuses is necessary.

An inclusion of the geometry of transport goods has to be implemented as well in order to guarantee collision free transportation. This will also have an effect on the distances between paths and objects.

In the future more fuzzy- rules will be conducted and different fuzzyfication and defuzzyfication method will be evaluated based on their adaptability in the AGV environment. With an increasing amount of fuzzy-rules the generated paths are expected to be even more realistic and the amount of manual labor for final road maps is expected to be minimized.

Through simulations in different softwaretools (e.g. Plant Simulation) the carrying capacity of the road map will be tested and compared to manual developed road maps. Important validation criterias are the amount of AGVs possible on the road map, the adherence to production schedules and the flexibility of the road map in case of holdups or collisions.

Even though a manual generated road map might show better results in the simulation, a cost-benefit analysis has to be done. A suboptimal automatically constructed roadmap might still be preferable if only a few changes have to be done manually and the generation time is significantly shorter.

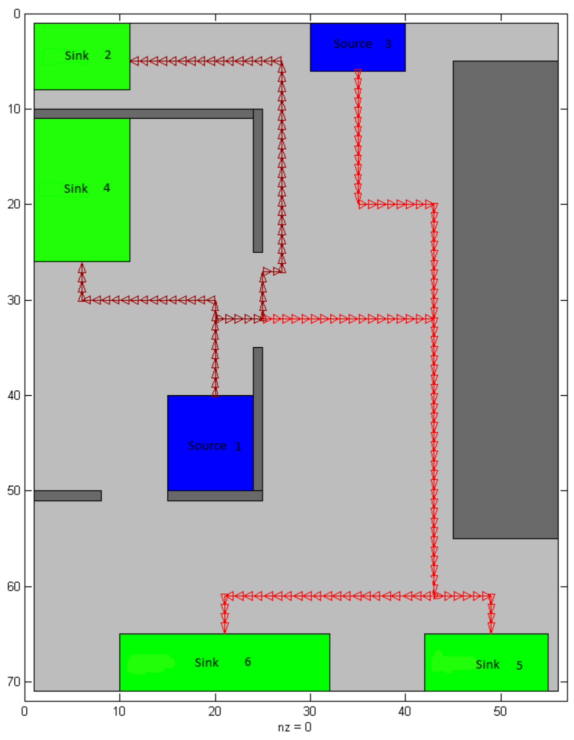


Figure 6: Generated Roadmap of the FEA*

7. Conclusion

In this paper a fuzzy enhanced path planning algorithm is proposed that designs an applicable and efficient road map for AGVs in complex production environments. The production environment is assumed to be known and static.

In order to obtain a realistic road map that follows security guidelines, norms, etc. a modified A* algorithm was implemented.

A fuzzy logic controller was implemented that uses a set of rules developed through interviews with AGV experts. The output of these rules is defuzzyfied and used in the minimization of the cost function of the mod A*.

Unlike most of the path planning algorithms a realistic road map is generated that can be used for production environments without a lot of manual changes. Producing near optimal, but applicable, road maps with the possibility of changing the rules later on is supposed to be a big benefit for the increasing size and complexity of AGV systems.

8. Acknowledgments

The work of Sarah Uttendorf is supported by the IGF project 18007 N/1 of the Bundesvereinigung Logistik (BVL) e.V. through a resolution of the German Bundestag.

References

[01]Niemann, B.; Baum, M.; Overmeyer, L.; Fricke, D.-H.: Aufbau von Fahrerlosen Transportsystemen (FTS) durch einzelne Datenstruktur. In: Logistic Journal, 2006.

[02]Duchon, F.; Babinec, A.; Kajan, M.; Beno, P.; Florek, M.; Fico, T.; Jurisica, L.:Path Planning with modified A star algorithm for a mobile robot, Procedia Engineering 96 (2014) 59 – 69, 2014.

[03]Hawa, M.: Light-assisted A* path planning, Engineering Applications of Artificial Intelligence 26, p. 888–898, 2013.

[04] Meléndez, A.; Castillo, O.: Evolutionary Optimization of the fuzzy integrator in a navigation system for a mobile robot. In: Recent Advances on Hybrid Intelligent Systems, SCI 451, pp.21-31.

[05] Wang, M.; Liu, J.: Autonomous Robot Navigation using Fuzzy Logic controller. In: Proceedings of the third international conference on machine learning and cybernetics, Shanghai, 26-29 August 2004.

[06]Delling, D. and Sanders, P. and Schultes, D. and Wagner, D.: "Engineering route planning algorithms". Algorithmics of large and complex networks, Springer, pp. 117–139, 2009.

[07]Zeng, W.; Church, R. L.: "Finding shortest paths on real road networks: the case for A*". International Journal of Geographical Information Science 23 (4): p. 531–543, 2009.

[08]Nilsson, N.: Artificial Intelligence: A New Synthesis, ISBN 1-55860-467-7, Morgan Kaufmann, 1998.

[09]Lu, H.; Chuang, C.: The Implementation of Fuzzy-Based Path Planning for Car-Like Mobile Robot, Proceedings of the 2005 International Conference on MEMS, NANO and Smart Systems, 2005.

[10]Carmona, P.; Castro, J.; Zurita, J.: FRIwE: Fuzzy Rule Identification with Exceptions, IEEE Transactions on Fuzzy Systems, Vol.12, NO.1, February 2004.

[11] Patel, A.v.; Mohan, B.M.: Some numerical apsects of center of area defuzzyfication method, Fuzzy Sets and Systems 132, p. 401-409, 2002.