Offline path planning of automated vehicles for slow speed maneuvering

Árpád Barsi
Department of Photogrammetry and Geoinformatics
Budapest University of Technology and Economics
Budapest, Hungary
barsi.arpad@epito.bme.hu

Ádám Nyerges
Department of Automotive Technologies
Budapest University of Technology and Economics
Budapest, Hungary
adam.nyerges@gjt.bme.hu

Vivien Potó
Department of Photogrammetry and Geoinformatics
Budapest University of Technology and Economics
Budapest, Hungary
poto.vivien@epito.bme.hu

Szilveszter Siroki
Department of Automotive Technologies
Budapest University of Technology and Economics
Budapest, Hungary
s.szili14@gmail.com

Viktor Tihanyi
Department of Automotive Technologies
Budapest University of Technology and Economics
Budapest, Hungary
viktor.tihanyi@gjt.bme.hu

Márton Virt
Department of Automotive Technologies
Budapest University of Technology and Economics
Budapest, Hungary
virtm96@gmail.com

***Abstract*—In the last century in road transport the main motivation was to make driving easier or more comfortable. Today lower fuel consumption, higher traffic safety and reduced environmental impact are in the focus of the developments. To reach these future objectives it is necessary to increase the level of automation of road vehicles.**

Driving a road vehicle by a software is a complex controlling task. In connected and automated vehicles the control algorithm has several steps. An important step is, when the vehicle plans its own trajectory. The trajectory planning process has several parts for instance the geometry of the path-curve or the speed during the way.

This paper presents a basic approach for path design. To reach the aim a map will be given as a binary 2204 x 1294 size matrix where the roads will be defined by ones, the obstacles will be defined by zeros. The map presents a smaller area of the campus of the Budapest University of Technology and Economics. The aim is to make an algorithm which can find the shortest and a feasible path for vehicles between the start and the target point. The vehicle speed will be assumed slow enough to ignore the dynamical properties of the vehicle.

Keywords—Connected and Automated Vehicles, Autonomous Driving, Self-driving Vehicles, Offline Trajectory Planning, Path Planning, Shortest Path Algorithm, Clothoid Fitting

I. INTRODUCTION

Thanks to the recent revolution of science and technology, vehicles have more and more automated features or systems. To reach future aims it is necessary to increase the level of automation of road vehicles. At the end of the decade, automated vehicles are going to appear in everyday transportation. Automated vehicles will better than today's cars in efficiency, comfort, safety, velocity and traffic density. Connected cars have another advantage: with intelligent traffic control systems, traffic jams can be decreased or even avoided.

When talking about the motivation of automated driving one of the biggest expectation is the radical reduction of the accident number and severity, since 94% of the current traffic accidents can be traced back to the human drivers [1].

Due to the new components and increased in-vehicle system complexity, vehicle testing and validation became different as earlier. Testing the vehicle, the driver-controller and the traffic situations together requires new testing methods and strategies. The aim is the same as earlier: to guarantee road safety with reliable operation of the systems.

In this paper the next step of the trajectory planning process will be discussed which is related to a university research [2,3]. The vehicle now can do the lateral and the longitudinal control and it can solve some traffic situations for instance traffic jam assist. The next aim is to make the valet parking function in the parking area of the University. The main requirement of this new feature is to create a new, faster trajectory planner algorithm, because the previous ones were too slow. The test vehicle can be seen on Fig. 1.

978-1-7281-1117-9/18/$31.00 ©2018 IEEE

Fig. 1. Photo of the test vehicle

II. Trajectory planning layer

The trajectory planning process of automated vehicles has several parts. Apart from the curve on a map [8] it is necessary to take into consideration the dynamical properties of the vehicle and the driving comfort for the passengers [9].

Different features of automated vehicle require different complexity from the control algorithm. In many traffic situations it is enough to handle the vehicle as a point-like body and due to its low speed the accelerations can be neglected. An example could be a vehicle when it searches a parking place. In this paper this basic approach is going to be analyzed. In further researches the sizes of the vehicle is not going to be neglected and for instance in urban or in rural roads the speed planning also should be defined.

Another important aspect is the fixedness of the environment. When only one vehicle drives on the road it is enough to use a fix map. But, for example in a car park where there are full and empty places, the map has to been changed, these are fix obstacles. When there are other road users, moving obstacles have to appear on the map.

The trajectory planning process can be done in two types. If the vehicle has information about the environment (GPS, HD map, etc.), the path can be planned before the cruise on the coordinate system of the map. If there is not a pre-known map the vehicle can go by the information from its sensors (real time path planning) [10,11]. In this case the path is planned from the coordinate system of the vehicle. Of course the two paths can transform into each other.

In this paper the curve planning part of the trajectory planning process will be discussed. Besides the maps are going to be fixed and there are not going to be other road users.

III. The applied map

The map of the pilot site was compiled based on combined survey with GNSS and terrestrial laser scanning technologies.

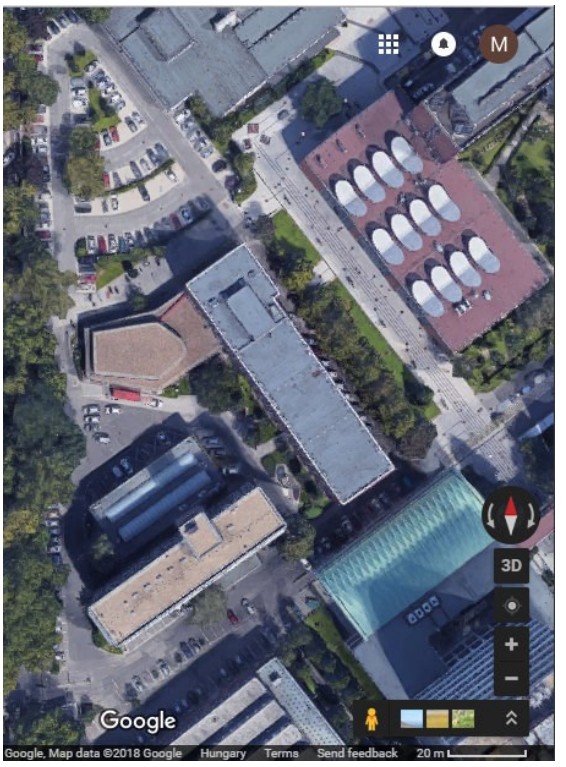

Fig. 2. The map of the applied territory

Terrestrial laser scanning was performed in April 2017 by a Faro Focus 3D 120S instrument, which has a 360° horizontal scanning and ~120 m range measurement capability. The geometrical resolution was set to 6 mm in 10 m. Several stations were applied to achieve less shadowing effect. The raw result was a unified point cloud having x,y,z coordinates and the received laser pulse intensity. Although GPS measurements enable to georeference the laser scanned data, for this study we ignored them.

Then the point cloud was manually reduced on the surface height and all relevant object borders were drawn in Autodesk AutoCAD environment manually. The obtained situation drawing was to be checked against topology and after accepting it, all polygons were filled by texture patterns considering that four layers are requested: (1) always available road surfaces, (2) potential parking slots, (3) surfaces not available for vehicles and (4) background (extension to minimal bounding rectangle).

The topologically correct and layer-organized drawing was imported into QGIS, where further checks were executed, then shp format was exported. The 10 cm resolution final occupancy grid was derived by an in-house developed Matlab script. The resulted raster has a size of 2204 × 1294 elements (Fig. 3).

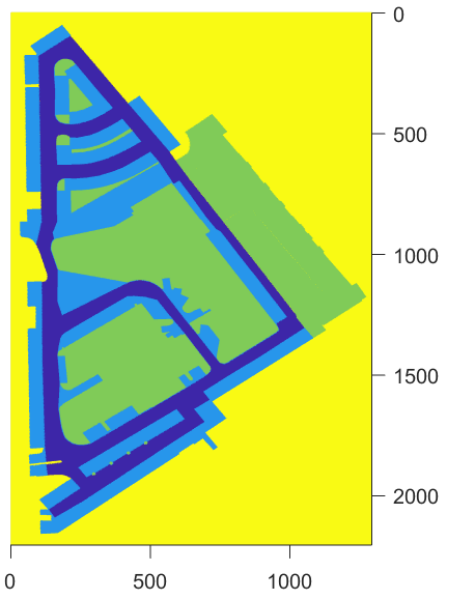

Fig. 3. The applied map matrix

IV. Finding the Shortest Path

The presented problem was solved with a simplified graph-system. In graph calculations, the run time of the software can be improved by the low number of the nodes. In this method the road junctions were defined as graph nodes (see on Fig 5), the streets are edges which can be seen on Fig 4. Because of the traffic rules (the two one-way road) the graph has to be directed. Between node 8 and 7 and between node 8 and 10 the direction of the edges are defined by the traffic rule on the map. The weights of the edges are the road distances between the nodes, as it can be seen on Fig 6. The weights were measured by DGPS data. In the graph the nodes have a number, but the nodes also have coordinates to. They can be the GPS coordinates, but in this paper the row and the column number was used.

Fig. 4. The applied map matrix

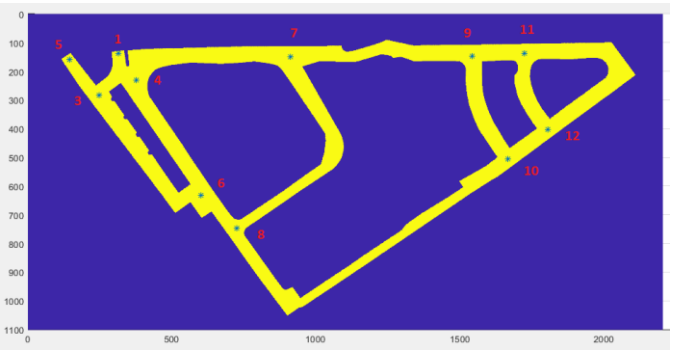

Fig. 5. The applied simplified graph (the stars are the nodes)

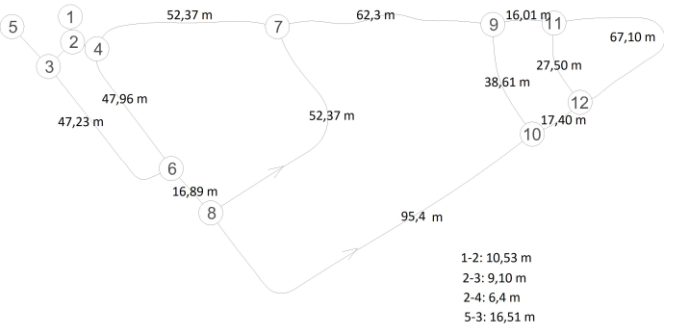

Fig. 6. The simplified directed graph

The actual position of the vehicle will be an additional node in the graph. To add this extra node to the graph, the coordinates of the start point could help. All connected node coordinates create a line, where the coordinates of the marked positions are known and with the point-line distance formula the two nearest and connectable main node can be calculated. The algorithm runs through all the possibilities and finds the minimum distances. When the distance of the vehicle position is the smallest from the line, the two nearest node is found. If it is necessary the target point also can be handled is a similar way.

To set the distances for the new nodes (start and target points) the distances (weights of the edges) also important to set. Formula (1) was used in the algorithm with its notation:

$$d = \frac{|(y_2 - y_1) * x_0 - (x_2 - x_1) * y_0 + x_2 * y_1 - x_1 * y_2|}{\sqrt{(y_2 - y_1)^2 + (x_2 - x_1)^2}} \quad (1)$$

- d: distance,
- x1,y1: the first main or assist node coordinates,
- x2,y2: the second main or assist node coordinates,
- x0, y0: vehicle position coordinates (start or target).

The next step in the path planning is the finding of the shortest path to the target place. The first method for this aim was the Dijkstra algorithm [12]. This algorithm calculates the shortest path between a starting node and all of the other nodes in a graph, included the target too. Matlab has several functions based on the Dijkstra algorithm, for this paper the "shortestpath" function is suitable [13]. This function is faster

than the original Dijkstra algorithm, and it is fast enough to use it.

Fig. 7. An example for the shortest way between two points

The result of the algorithm is a vector which gives the order of the nodes from the start point to the target point. Fig 7 and Fig 8 present some examples from the shortest path finder algorithm. Here the curves were made just for illustration, the exact path will be planned is the following chapter.

Fig. 8. An example for the shortest way between two points

V. Making a Path for a Vehicle

The next task of the path planning algorithm is to find a feasible curve for vehicle driving. Basically, the feasibility has three important criteria. Firstly, the curve's maximum curvature needs to be below the maximum curvature that the vehicle can realize. Secondly, the calculated path must be collision free, and finally the path has to be tangentially continuous. There are two more criteria that helps to minimalize the tracking error: the path's curvature has to be continuous, and differentiable. In this chapter the first three criteria are going to be studied.

The core of the path planning method is the clothoid curve. The algorithm fits tangentially connected clothoids to given reference points. This is called Hermite G1 interpolation, and this is how we make the curve tangentially continuous. The tangents of the two connected segments are the same at the connection points. The advantage of this method is that it fits easily the curves to the reference points, but it has a drawback: we cannot constrain the tangent and the curvature of the reference points at the same time, because clothoid curves do not have enough degree of freedom, so we cannot connect the segments to each other with continuous curvature. Therefore, the vehicle will have a basic tracking error. Our researches show that this is not a big problem, the path tracking is still accurate enough.

The path planning algorithm uses predefined ways. This method is preferred because parking places in smart city applications can also have predefined ways that vehicles can use to calculate their own path in a short time. This method reduces both the computation tasks and the time of the calculation.

Before describing the exact method, we need to define two expressions. Waypoints are the junctions of the roadmap and the shortest path algorithm results these waypoints in the correct moving order. Reference points are the points that the program uses to calculate the clothoid segments. Each waypoint has several reference points (at least one point to each direction of the junction) but there are some reference points which do not belong to any waypoints. These are the assist points and they are usually located at bends. Reference points are previously given to the program manually. These points will result a feasible curve to all the possible paths. In real use, vehicles can get the points via WiFi or 5G. With these data and the previously resulted shortest path, the calculation method can be started.

At first, the location of the starting and the endpoint must be determined. A function gets the reference points and the last or the first waypoint of the shortest path, depending on which point (start or end) needs to be determined. With the help of a connection matrix it calculates all the possible path segments starting from the proper waypoint. The function searches the closest calculated segment to the tested point. Knowing this segment, the program finds the two reference points between which the tested point is located. One of them belongs to the given waypoint. The other reference point belongs to a different waypoint and it is called neighbouring point. Note that the neighbouring points are not parts of the final path, they are outside of it. First, we make an extended path between the neighbouring points of the starting and the endpoint. After that the extended path needs to be cut at the closest points to the starting and endpoints.

The calculation method of the extended path is the following. The waypoints of the neighbouring points are the neighbouring waypoints. The extended waypoint vector is determined by adding these to the waypoint vector that the shortest path function resulted. This extended vector will be the input of the extended path calculator. While creating the path segments, the algorithm leads the curve through the current waypoint and after that it leads it to the next waypoint. A loop is repeating the process until it reaches the final waypoint. The first waypoint is the neighbouring waypoint of the starting point. In this case, it is marginal how the curve goes through the waypoint because this part of the extended

path will be cut off. In order to create the path leading to the next waypoint, the requested reference points are given by a matrix called UT. This UT matrix only contains the assist points. After the first waypoint, the loop reaches the general cases. The reference points that are needed for the path leading through the current waypoint are given by a matrix called CSP. The data is stored in the rows of the matrix. For example, if the value of the first three columns of the row is 3, 6, 5, it means the case when the current waypoint is wp. 6, and the previous waypoint where the curve came from is wp. 3, and the next waypoint where we would like to go is wp. 5. The columns after the first three columns contain the x, y, and alpha coordinates of the reference points that the clothoid making function needs to use to make the correct clothoid curves to go through the waypoint. After that, in this example, if there are some assist points between wp. 6 and wp. 5, the matrix UT will add them to the reference point vector of the path. The exact implementation of the last waypoint is also marginable, because it will also be cut off. When this function finished running, the reference point vector is determined, so it can be given as an input to the chlotoid making function to create the extended path.

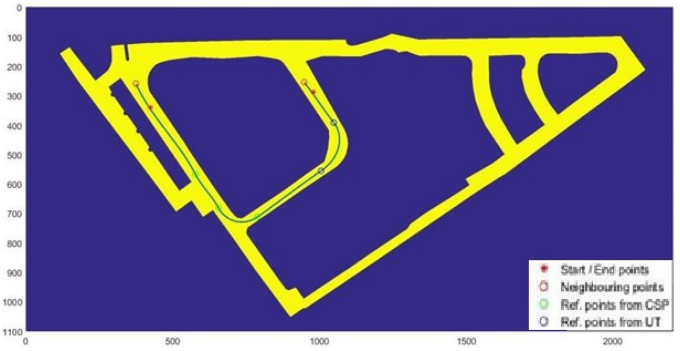

Fig. 9. An example for the extended path on the map

The clothoid fitting algorithm is based on [15]. The function gets the coordinates and tangents of the two points and as an output it returns the clothoid segment's coordinates between the two given points.

The creation of the final path from the extended path is an easier process. The cut function gets the starting point, the endpoint and the extended path. It finds the path-points with the minimum distance from the two given points. These path-points will be the modified starting and endpoints. The function clears the points before the modified starting point and after the modified endpoint. With this step, the process is finished. As it can be seen, most of the time there will be a non-marginable distance between the vehicle's starting position and the path's starting position. This will cause a tracking error at the beginning of the way but as the car goes ahead, the vehicle control will drive the car to the path, so for the rest of the road it will not be a problem. However, it also is not a problem at the start because the predefined paths are mostly in the middle of the road, so the control system always leads the car to the safer regions.

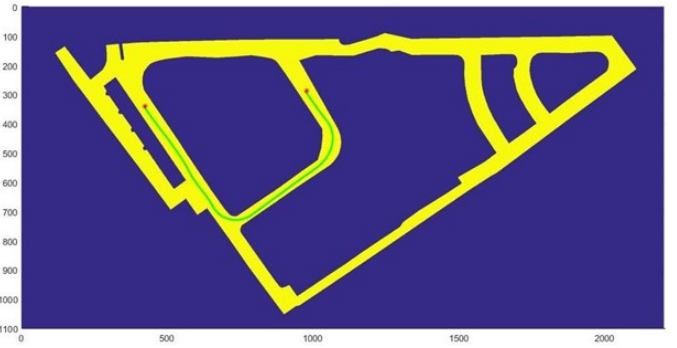

Fig. 10. An example for the final path on the map

The predefined paths need to be collision free. To guarantee this criterion the path were checked with an algorithm while making them. The algorithm checks every points of the path if there are any points of the car that are not placed in the road. The car is modelled with a rectangle. The sides' lengths of the rectangle are a bit longer than the actual sizes of the car because we defined a safety offset around it. The points of the sides are calculated from the vehicle's position and tangent, and a loop checks the points' values on the map matrix. If there are any 0 values the examined path-point will cause a collision.

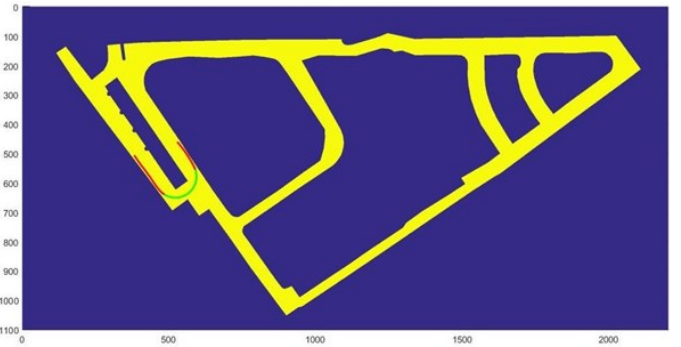

Fig. 11. An example path with collision checking

The main task of this algorithm is to drive the vehicle from position A to position B in the parking place in a static environment. This does not require precise maneuvering, so the tracking errors that the method causes are acceptable. The computation generally takes a few seconds in an average up-to-date computer.

VI. Concluding Remarks

In this paper a basic path planner algorithm was presented for automated vehicles. The algorithm handled a point-like vehicle model and the obstacles on the map were fixed, there were not any other road users. The speed of the vehicle assumed to be enough slow to neglect the dynamical properties of the vehicle.

In the first section of the paper the map making was presented. The map shows an existing area about as a binary matrix. The next step was to make a simplified graph from the map. In the graph the nodes were determined by the junctions,

the edges were determined by the roads and the weights were determined by the distances between the nodes. The simplified graph was directed, due to that the one-way roads were be able to handled.

In the next section the task was to find the shortest way on the graph between the given start and target point. There are many suitable algorithm to find shortest ways on graphs, but in this simple map Matlab's built-in algorithm was fast enough.

Finally the path maker algorithm was presented which makes a feasible curve for vehicle driving. The algorithm could make the path for the vehicle through the pre-defined reference points by the clothoid fitting algorithm and it also can do collision check.

The aim of the research is to realize valet parking function on the automated car of the university. This offline path planning algorithm can be implemented in it if it is handled with a state switcher before the vehicle starts to go.

ACKNOWLEDGMENT

The project has been supported by the European Union, co-financed by the European Social Fund. EFOP-3.6.2-16-2017-00002

REFERENCES

[1] ISO, "International standard road vehicles - functional safety," International Organization for Standardization, Geneva, Switzerland, ISO26262

[2] Nyerges, Á., Tihanyi, V.: Trajectory Planning for Autonomous Vehicles – A Basic Approach, 2nd International Conference on Vehicle and Automotive Engineering, 23-25 May 2018, University of Miskolc, Hungary

[3] Barsi, Á., Nyerges, Á., Potó, V., Tihanyi, V.: An Offline Path Planning Method for Autonomous Vehicles, Production Engineering Archives 19 (2018) 37-42

[4] SAE International: 'Taxonomy and definitions for terms related to on-road motor vehicle automated driving systems'. SAE standard, nr. J3016__201401, 2014-01-16, http://standards.sae.org/j3016_20140 (Accessed 30 October 2018)

[5] Szalay, Zs., Nyerges, Á., Hamar, Z., Hesz, M.: 'Technical specification methodology for an automotive proving ground dedicated to connected and automated vehicles'. Periodica Polytechnica, Transportation Engineering, Hungary, 45(3), pp. 168-174, 2017

[6] Szalay, Zs.: 'Structure and architecture problems of autonomous road vehicle testing and validation'. Paper presented at 15th Mini Conference on Vehicle System Dynamics, Identification and Anomalies – VSDIA, 11th November 2016. Budapest, Hungary

[7] Tettamanti, T., Varga, I., Szalay, Zs.: Impacts of autonomous cars from a traffic engineering perspective, Periodica Polytechnica, Transportation Engineering, Hungary, 44(4), pp. 244-250, 2016

[8] Gu, T., Snider, J., Dolan, J., M., Lee, J.: Focused trajectory planning for autonomous on-road driving. IEEE Intelligent Vehicles Symposium (IV) June 23-26, 2013, Gold Coast, Australia 2013

[9] Hult, R., Tabar, R., S.: Path planning for highly automated vehicles Master's Thesis in Systems, Control and Mechatronics. Gothenburg, Sweden 2013

[10] Chebly, A., Tagne, G., Talj, R., Charara, A.: Local trajectory planning and tracking for autonomous vehicle navigation using clothoid tentacles method. HAL Id: hal-01139316, 2015

[11] Dolgov, D., Thrun, S., Montemerlo, M., Diebel, J.: Path planning for autonomous vehicles in unknown semi-structured environments. The international journal of robotics research Vol. 29, No. 5, pp. 485–501 2010

[12] Dijkstra, E., W.: A note on two problems in connexion with graps. Numerische Mathematik 1 (1959), 269-271

[13] MathWorks: Graph and network algorithms toolbox: User's guide (R2015b) 2015. www.mathworks.com/help/pdf_doc/ (Accessed 30 October 2018)

[14] MathWorks: Curve fitting toolbox: User's guide (R2015b) 2015. www.mathworks.com/help/pdf_doc/ (Accessed 30 October 2018)

[15] Bertolazzi, E., Frego, M.: Fast and accurate clothoid fitting, ACM Transactions on Mathematical Software 2012

[16] Zöldy, M. (2018) 'Legal Barriers of Utilization of Autonomous Vehicles as Part of Green Mobility' N. Burnete and B. O. Varga (Eds.): AMMA 2018, PAE, pp. 1–6, 2019 https://doi.org/10.1007/978-3-319-94409-8_29 (Accessed 30 October 2018)

[17] Gáspár, P., Szalay, Zs., Aradi, Sz.: 'Highly automated vehicle systems' BME MOGI 2014, http://www.mogi.bme.hu/TAMOP/jarmurendszerek_iranyitasa_angol/index.html (Accessed 30 October 2018)

[18] Nyerges, Á., Szalay, Zs.: 'A new approach for the testing and validation of connected and automated vehicles'. Paper presented at 34th International Colloquium on Advanced Manufacturing and Repairing technologies in Vehicle Industry, 17-19 May 2017, Visegrád, Hungary

[19] Barsi, Á., Nyerges, Á., Potó, V., Tihanyi, V.: Offline Path Planning for Autonomous Vehicles, 35th International Colloquium, Advanced Manufacturing and Repair Technologies in Vehicle Industry, 23-25 May 2018, Zielona Góra – Lagow Lubuski Poland