Guided Hybrid A-star Path Planning Algorithm for Valet Parking Applications

Saeid Sedighi, Duong-Van Nguyen
Panasonic Automotive Systems Europe GmbH
Langen, Germany
e-mail: saeid.sedighi@uni-siegen.de,
duongvan.nguyen@eu.panasonic.com

Klaus-Dieter Kuhnert
University of Siegen
Siegen, Germany
e-mail: kuhnert@fb12.uni-siegen.de

***Abstract*—Throughout the last decades, the Robotics Community has influenced the Autonomous Vehicles field in multiple different areas ranging from Scene Understanding and Decision Making to Vehicle Control and Optimal Path Planning. Existing path planning algorithms such as A-star, Dijkstra and Graph-based approaches, although providing good optimal path approximations, are suffering from high-runtimes to convergence. This paper presents an innovative and computationally efficient approach of fusing the well-known Hybrid A-star search engine with the Visibility Diagram planning to find the shortest possible non-holonomic path in a hybrid (continuous-discrete) environment for valet parking. The primary novelty of our method stems from two points: at first, we use the Visibility Diagram to introduce an improved and application-aware cost function for the Hybrid A-star algorithm, and then the derived shortest path is used to provide the correct waypoints for the Hybrid A-star to plan the optimum path in regard to the non-holonomic constraints. The method has been extensively tested and proven to up to 40% (20% in average) faster than Hybrid A-star algorithm.**

Keywords-component; path planning; hybrid A-star; visibility diagram; valet parking; autonomous driving

I. Introduction

Robotics supports the automotive industry in several aspects from the manufacturing level to the driving comfort. Nowadays, all manufacturing sites of the automotive industry are mainly handled by robots ranging from robotics arms to the moving autonomous load carriers [1]. For driving comfort point of view, by the outset of automatization of cars as so-called *autonomous driving* technology, robotics shared its practical ideas of autonomous mobile robotics with the automotive industry. For executing complex tasks in an autonomous way, a mobile robot (or an autonomous vehicle) requires three core capabilities [2]: first, to understand its surrounding by sensing all important information as much as possible, call this part as *perception*.

The second part is using the perception part and its collected information to decide how to proceed from the current configuration state ($q_c(x_c, y_c, z_c, \theta_c, ..)$) to the next desired state ($q_g(x_g, y_g, z_g, \theta_g, ..)$). This part includes plans and moving decisions such as stop, move, idle and so on, this step is named as *Planning*. The third part is a low-level controller in which the main purpose would be executing the commands of the planning section. This part (*Control*) communicates with the perception part as well as re-tracks itself while performing the action (Fig. 1). In the literature, some researchers use different terms for the above-mentioned parts or also so-called layers [3], but the work order always has the same standard form as in Fig. 1.

As it is shown in Fig. 1 any time latency in any stage of the system architecture drops the performance of the whole system at the end, which must be taken in the consideration to use the following system architecture.

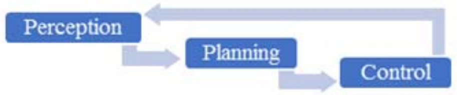

Figure 1. Schematic view of the system architecture in an autonomous vehicle.

This work concentrates more on the planning part of an autonomous vehicle or generally automated mobile robots in which one of the most important parts is path planning. In robotics, path planning is the most important task of planner layer to steer a robot (e.g. in a continues environment) to navigate the ego mobile robot to reach the goal state from the start state in a purely geometric manner.

In artificial intelligence (AI), the term planning takes on a discrete aspect. Instead of using continuous motion, the vehicle motion planning problem is solved discretely [4]. Although this problem could be modeled with continuous spaces, by defining a finite set of actions that can be applied to a discrete set of states.

One of the challenges of path planning for autonomous vehicles is using a discrete input (such as occupancy grid map) and plan a path in a continuous domain to feed the controller stage. Path planning for a continuous domain has been already explored by upgrading some path planning algorithms of a discrete domain such as A-star manipulating continuous maneuvers (Hybrid A-star [5]). Hybrid A-star is a robust path planning method for non-holonomic robots such as autonomous vehicles [5][6].

In this paper, we propose a method to guide the A-star search algorithm to find its optimum path in a hybrid (continuous-discrete) environment faster than the common Hybrid A-star. We use Visibility diagram algorithm to find the shortest possible holonomic path from the start to the goal position, which allows us to establish some waypoints to support A-star search engine to find its way faster. Using our method, the running time of the Hybrid A-star algorithm for big discrete maps (e.g. valet parking) was reduced up to 40 percent (on average by 20 percent).

A. Related Works

The very first algorithms in path planning for robotics such as A-star stablished the path planning platform for autonomous vehicles [5]. The problem of using these methods is the non-holonomic constraints of autonomous vehicles.

Some recent works solved this problem while upgrading the old methods of trajectory planning and using the kinematic of the vehicle in the planning process such as rapidly random trees (RRTs) and Hybrid A-star [5].

Hybrid A-star is one of the most efficient path planners for non-holonomic autonomous vehicles. The very first version of this planner has been introduced in DARPA Urban Challenge 2007 [5]. Using this path planning approach, the most important point which connects discrete and continuous state space to each other is how to expand the search algorithm and choosing the correct successor nodes to move further. This method implements the kinematics of the vehicle to select the next potential nodes as successor nodes (Fig. 3).

Nevertheless, the above method was struggling with some problems which made the realistic applications of this method a bit problematic, such as following several waypoints and having a more realistic heuristic cost function for the A-star search engine. These problems will be more visible in a big map where the target position is very close (Euclidian distance) to the start position but the actual driving distance to the target area is quite significant as shown in Fig. 2.

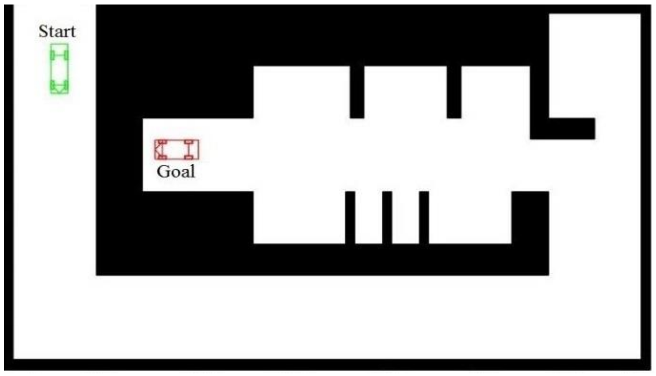

Figure 2. Grid map view of a sample valet parking area including start and target positions.

Some recent works have improved the performance of Hybrid A-star to follow two waypoints [6]. This application of Hybrid A-star lets us to follow several waypoints without interrupting the goal tracking of the algorithm. However, the waypoint tracking of Hybrid A-star algorithm got improved, the algorithm still needs a better heuristic cost function which lets it to have a better understanding of the global map and its correct tracking path.

Here, we have concentrated more on this problem to create some useful waypoints for the algorithm to being followed by using the capability of Visibility diagram algorithm.

II. Hybrid A-star Algorithm

In engineering, the hybrid term is always assigned to interdisciplinary methods which combine several branches of engineering to benefit the potential of all in a moderated way. Here we use this term because the algorithm uses an occupancy grid map in which the environment is illustrated in a discrete way to plan the motion of the vehicle in a continuous domain.

A-star algorithm is a well-known method in path planning [3][7]. The normal A-star search algorithm uses occupancy grid map to assign a cost for each surrounding node around the current node ($F(n) = g(n) + h(n)$ in which $h(n)$ represents the cost to reach the center of the next successor node and $g(n)$ is the cost to reach the target node from the current successor node) and moving to the next successor node by choosing a linear (left-right, up-down) maneuver [4]. As it is clear here, this method works just for holonomic robots (or vehicles).

Hybrid A-star search algorithm improves the normal A-star algorithm to be implemented on a non-holonomic robot (e.g. autonomous vehicles). The key to do that is using the kinematic of the ego vehicle to predict the motion of the vehicle depending on the speed, gear, steering angle and other parameters of the vehicle. This method helps the planner to select the right successor node which a non-holonomic robot is able to follow. Fig. 3 illustrates how the Hybrid A-star algorithm selects the next successor nodes depending on the several positions of the steering angle of the ego vehicle.

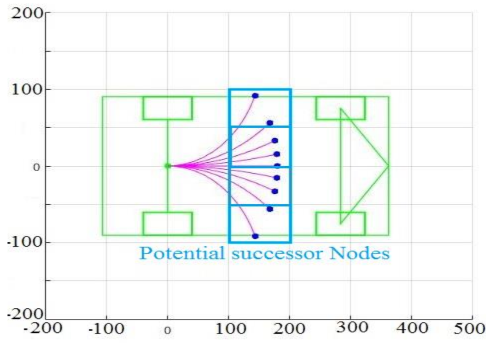

Figure 3. Hybrid A-star on a grid map (All dimensions are in *cm*).

Hybrid A-star uses the same cost function as the normal A-star algorithm to find the path for reaching the target node, additionally, it will add some more costs to that depending on the motion of the vehicle and its moving conditions.

As it was explained in the first section, some recent researches improved the basic version of Hybrid A-star in finding a global path for a non-holonomic vehicle (*Veh*) in a provided map (*M*) by following several waypoints [6]. This algorithm which represents the state of art of Hybrid A-star algorithm is presented here (an application of using two-waypoints):

Hybrid A-star Path Planner Algorithm with Two Waypoints

```
algorithm HybridAstarPlanner($M$, $Veh$, $qS$, $qG_1$, $qG_2$)
    $n_s \leftarrow (qS, 0, h(qS, qG_1, qG_2), 1)$
    $O \leftarrow \{n_s\}$
    $C = \emptyset$
    while $O \neq \emptyset$ do
        n $\leftarrow$ node with the minimum $f$ value in $O$
        $O \leftarrow O \setminus \{n\}$
        $C \leftarrow C \cup \{n\}$
        if $n_s = 1$ then
            if $n_x \in qG_1$ then
                UpdateSeccessors($M, Veh, O, C, n, 2$)
            else
                UpdateSeccessors ($M$, $Veh$, O, C, n, 1)
            end if
        else if $n_s = 2$ then
            if $n_x \in qG_2$ then
                return planned path
            else
                UpdateSeccessors ($M$, $Veh$, O, C, n, 2)
            end if
        end if
    end while
    return no path
end
function UpdateSeccessors ($M, Veh, O, C, n, s$)
    for all defined steering angles $\delta$ do
        $n'$ $\leftarrow$ succeeding state of $n$ using $Veh$ ($n_\theta$, $\delta$)
        $n'_s \leftarrow s$
        if $n' \notin C$ then
            if $M(n')$ = obstacle then
                $C \leftarrow C \cup \{n'\}$
            else if $\exists n \in O : n_x = n'_x \land n_s = n'_s$ then
                compute the optimum node cost
            else
                $O \leftarrow O \cup \{n'\}$
            end if
        end if
    end for
end
```

The above-mentioned algorithm is just a static extension of Hybrid A-star algorithm [6] which is technically constrained to some limited number of waypoints and does not have a realistic heuristic cost function either.

III. Visibility Diagram Planner

Visibility diagram is one of the very first graph-based search algorithms which was used in path planning in robotics [4], [8]. This method guarantees finding the shortest path from the start to the target position. This method was introduced to the robotics with the naïve algorithm and time complexity of O(n^3) [8]. Lee's Visibility graph algorithm, improved this method to reduce the performing time of the algorithm to O($n^2 logn$) [8]. Because this graph-based search algorithm uses direct Euclidian distances between edges of the objects and obstacles in the map ($S(V)$), it always returns the shortest possible path from start to the target position. The state of art of this method is represented below:

Graph-based Visibility Diagram Path Planner

```
algorithm VisibilyDiagramPlanner($S(V)$, $qS$, $qG$)
    $E, G = 0$
    for each $v \in V$ do
        W $\leftarrow$ VisibleVertices($v$, $S$)
        $\exists w \in W \quad E \leftarrow E \cup \{v, w\}$
    end for
    if $E \neq \emptyset$ then
        return $G$ = AStarSearch[1]($E$, $qS$, $qG$)
    end if
end
function VisibleVertices($P$, $S$)
    ($w_1, \ldots, w_n$) = sort($S$) // clockwise angle
    $\rho$ = Half-line parallel to the positive $x$-axis starting at $P$
    $T$ = Balanced search tree of obstacle edges intersect $\rho$
    $W, T = 0$
    for $i \leftarrow 1$ to $n$ do
        if Visibility($w_i$) then
            $W \leftarrow W \cup w_i$
            $T \leftarrow T \cup$ {clockwise edges which incident
              to $w_i$ and lie on $\rho w_i$}
            $T \leftarrow T \setminus$ {counterclockwise edges which
              incident to $w_i$ and lie on $\rho w_i$}
        end if
    end for
return $W$
function Visibility($w_i$)
    if $\rho w_i$ intersects the interior of the obstacle of which
      $w_i$ is a vertex then
        if $e$(one leaf edge)$\in T$ and $e$ intersects $\rho w_i$ then
            return $0$
        else
            return $1$
        end if
    else if $w_{i-1}$ is not visible then
        return $0$
    else
        search in $T$ for an edge $e$ that intersects $w_{i-1}w_i$
        if $e \neq \emptyset$ then
            return $0$
        else
            return $1$
        end if
    end if
end
```

1) Due to the limited space, the algorithm has been simplified. For further studies please refer to [4].

As mentioned before, this above-mentioned method provides the shortest possible path (holonomic) from the start to the goal position, the only concern about the application of this method in autonomous driving is the non-holonomic constrains of autonomous vehicles. By combining the Visibility diagram and Hybrid A-star methods, the benefits of each algorithm can be used to cover constrains of both.

IV. Guided Hybrid A-star

To improve the Hybrid A-star algorithm, in particular, its cost function, this method was combined into Visibility diagram. For this purpose, first, the Visibility diagram path planner is implemented on the given grid map including initial, target and obstacles positions. Then the Visibility diagram provides the shortest path to the target area. This path will provide the correct waypoints which Hybrid A-star uses to plan an optimum path regarding the non-holonomic constraints of the ego vehicle. These steps are explained in detail here:

Input: Occupancy grid map of the environment (*M*), starting position (q_S), target position (q_G), car specification (kinematic model of the ego vehicle) (*Veh*) and the location of obstacles (O_M).

Output: the optimum path for an autonomous vehicle (or a car-like mobile robot) from the given starting position to the desired target position (*Path*).

The following steps create the desired output from the given inputs:

Step one: The occupancy grid map is provided by locating the start, target and obstacle positions on that ($M(q_S, q_G, O_M)$) (Fig. 2) .

Step two: The Visibility diagram algorithm, uses the given map to find the shortest possible path from q_S to q_G as illustrated in Fig. 4:

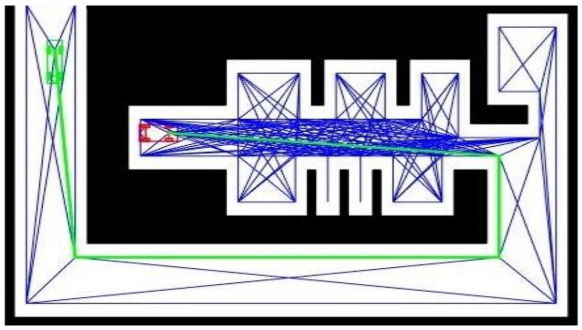
Figure 4. Planning the shortest path using Visibility diagram.

Step three: By using the planned path of Visibility diagram path planner, the vertices of the path apart from start and target points are extracted to set the waypoints which will be followed by Hybrid A-star (Fig. 5).

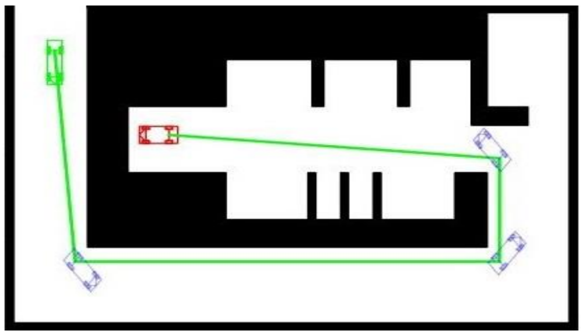
Figure 5. Setting the vertices of planned path as waypoints.

Step four: Using the above-mentioned version of Hybrid A-star to follow extracted waypoints to find the final optimum path to the goal position due to the kinematic constraints of the ego vehicle. The kinematic model of the vehicle with a global location of (*X, Y, θ*) was calculated by using a simplified bicycle model as follow [9]:

$$\beta = (L/W_b) \times \tan\alpha \tag{1}$$

$$R = L/\beta \tag{2}$$

$$X_i = X_{i-1} + R \times (\sin(\theta + \beta) + \sin\theta) \tag{3}$$

$$Y_i = Y_{i-1} + R \times (\cos(\theta) - \cos(\theta + \beta)) \tag{4}$$

$$\theta_i = (\theta_{i-1} + \beta) \bmod 2\pi \tag{5}$$

where L represents the desired travel length, W_b is the wheelbase and α the steering angle of the ego vehicle.

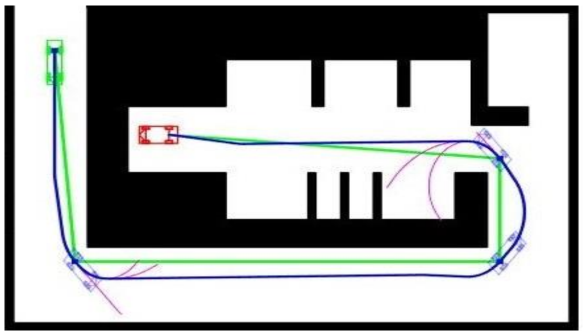
Figure 6. Finding the optimum path by following waypoints.

As it is shown in Fig. 6, guided Hybrid A-star planner does not need to scan the whole map and instead it is directly pointed to the waypoints which lead the cost function of the search algorithm to find the required optimum path much faster in a moderated way which satisfies the non-holonomic constraints of the vehicle as well.

These steps establish a new algorithm of path planning which is presented below:

Guided Hybrid A-star Path Planner

```
1 algorithm GuidedHybridAstar(M, Veh, qS, qG)
2   V, G ← Ø
3   G = VisibilyDiagramPlanner(M, qS, qG)
4   if G ≠ Ø then
5     V ← V ∪ qS
6     V ← G ∪ qG
7     i = 0
8     Path, PathSeg ← Ø
9     while qG2 ≠ qG do
10      qG1 = Vi , qG2 = Vi+1
11      PathSeg=HybridAstarPlanner(M, Veh, qS, qG1, qG2)
12      Path ← Path ∪ PathSeg
13      i++
14    end while
```

```
    if $Path \neq \emptyset$ then
      return $Path$
    end if
    end if
return $0$
```

This method was implemented on our test vehicle in Panasonic Automotive Systems Europe GmbH for several different (real and simulated) scenarios(more than 1000 use-cases) of occupancy grip map including obstacles (M), initial (q_S) and target position (q_G) as follow (Fig. 7):

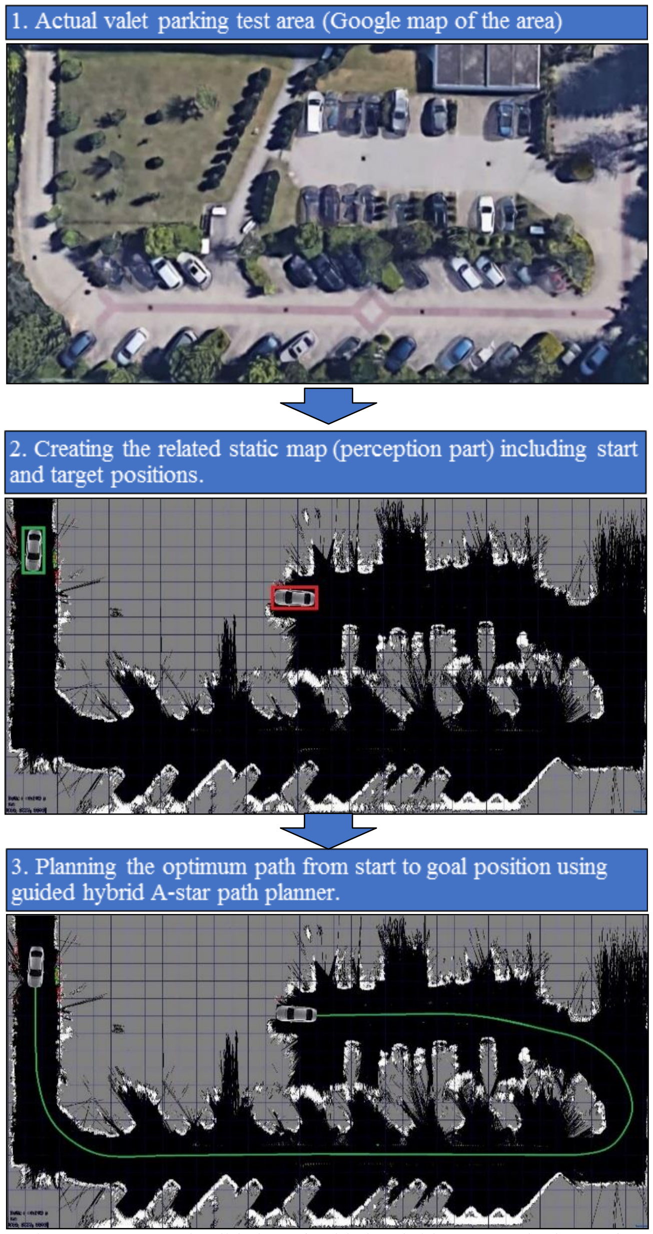
1. Actual valet parking test area (Google map of the area)
2. Creating the related static map (perception part) including start and target positions.
3. Planning the optimum path from start to goal position using guided hybrid A-star path planner.

Figure 7. Test and validation of guided Hybrid A-star path planner for actual valet parking scenarios.

The results of this study show that the average computing time of Hybrid A-star path planner is reduced by 20 percent in comparison to the normal Hybrid A-star.

Indeed, the following diagrams represent the comparison between the performance of normal Hybrid A-star and guided Hybrid A-star in finding the optimum path for a car-like mobile robot (an autonomous vehicle) for 1000 different M, q_S and q_G values.

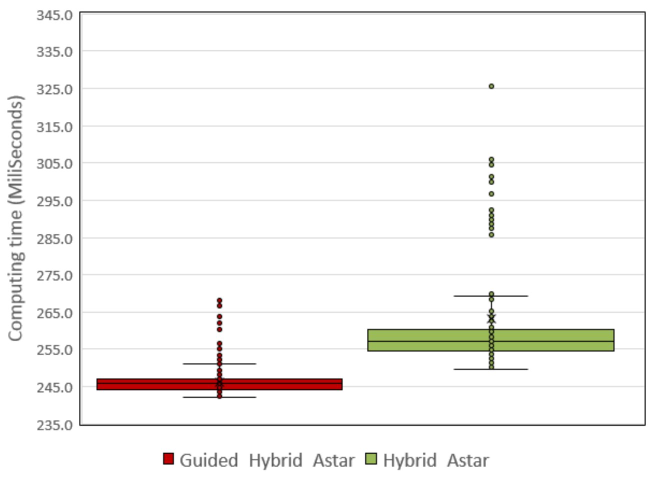

Figure 8. Comparison between the running time of normal Hybrid A-star and guided Hybrid A-star.

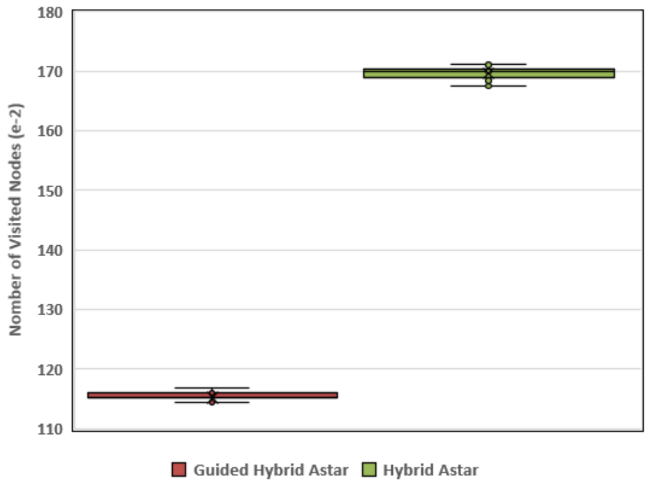

Figure 9. Comparison between the number of visited nodes by normal Hybrid A-star and guided Hybrid A-star.

As it is illustrated in Fig. 8, for some complicated scenarios in which normal Hybrid A-star has its maximum runtime, the guided Hybrid A-star has reduced the runtime by ~40 percent and has runtime even less than the average runtime of Hybrid A-star for the same scenarios.

Another noticeable point is the stability of performance of guided Hybrid A-star algorithm, as we see in figure 8, guided Hybrid A-star has a pretty stable computing (standard deviation) for all scenarios with different complexity. The deviation between the maximum and minimum runtimes regarding the average runtime in normal Hybrid A-star is about 18.9% and 5.7% respectively, which in guided Hybrid A-star the same deviations are about 6.1% and 1.6% which actually makes the application of guided Hybrid A-star path planner more reasonable for valet parking with a dynamic environment.

The main time consuming part of the normal version of Hybrid A-star is visiting the nodes while exploring the unknown environments. As it is shown in Fig. 9, in a guided way of path planning using Hybrid A-star, the number of visited nodes drops obviously. In our test and validation process, the number of visited nodes by guided Hybrid A-star was reduced by 35 percent. Less visited nodes let us upgrade our occupancy grid map resolution up to 4×4 (*cm*) in a big provided map (up to 200×200 (*m*)). A higher resolution of the map helps the planner to plan a safer and smoother trajectory due to the more available states to proceed.

V. Results

By using the new improvement of Hybrid A-star, the performance of the normal approach was improved significantly. The results which were achieved by implementing the algorithm for more than 1000 different scenarios (real and simulation) of map and car locations prove that using the Visibility diagram algorithm to find waypoints for guiding the main search engine of A-star improves the performance of the decision-making level of an autonomous vehicle for valet parking application with a dynamic environment significantly.

The mentioned test and validation use-cases were implemented by MATLAB R2015a on Win7 64-bit with Intel (R) Xeon(R) CPU 3.50 GHz processor.

The average reduction of the running time for several different test cases was about 20 percent and up to 40 percent for some test cases with big free spaces and short distance between the start and target positions (with an obstacle in between). The algorithm was tested and validated for both static and dynamic scenarios which proves the reasonable application of guided Hybrid A-star algorithm for real valet parking applications where the ego vehicles deal with very dynamic environments.

VI. Conclusions

In this paper, we presented a new method of path planning for autonomous vehicles for valet parking applications which benefits the advantage of Visibility diagram algorithm to improve the performance of Hybrid A-star path planner. As it was proved here, the fusion of two different path planners in a correct way reduced the runtime of the decision-making stage of autonomous vehicles up to 40 percent.

Having a faster decision maker helps the whole AI of autonomous vehicles to make decisions faster at the right time to handle dynamic scenarios which may save lives as well.

The results of this work show the great application of hybrid methods in engineering in particular robotics, the automotive industry, and autonomous driving. Hybrid methods of engineering help us to use the advantages of each method individually and put down their weak points at the same time. Back to this paper, fusing of path planning methods might be used for other path planners to improve their performance as well.

Likewise, by combining this fast environmental-exploration algorithm to a local optimum path planner for parking scenarios [10], the path planning problem of valet parking use-cases could be solved. This efficient approach of fusing and its realistic runtime leads the decision maker level of valet parking systems to plan an optimum path more efficient.

References

[1] Pfeiffer, Sabine. 2016. "Robots, Industry 4.0 and Humans, or Why Assembly Work Is More than Routine Work." Societies 6, no. 2:16. DOI=https://doi.org/10.3390/soc6020016.

[2] Pendleton, Scott D.; Andersen, Hans; Du, Xinxin; Shen, Xiaotong; Meghjani, Malika; Eng, You H.; Rus, Daniela; Ang, Marcelo H. 2017. "Perception, Planning, Control, and Coordination for Autonomous Vehicles." Machines 5, no. 1:6. DOI= https://doi.org/10.3390/machines5010006.

[3] Sung, Kyung-Bok and Dong-yong Kwak. “System architecture for autonomous driving with infrastructure sensors.” 2012 6th International Conference on Signal Processing and Communication Systems (2012): 1-6. DOI= https://doi.org/10.1109/ICSPCS.2012.6508023.

[4] LaValle, Steven M. Planning Algorithms. Cambridge: Cambridge University Press, 2006. DOI= doi:10.1017/CBO9780511546877

[5] D. Dolgov, S. Thrun, M. Montemerlo, J. Diebel, “ Practical search techniques in path planning for autonomous driving”, Proc. of the First International Symposium on Search Techniques in Artificial Intelligence and Robotics (STAIR-08), June 2008.

[6] J. Petereit, T. Emter, C. W. Frey, T. Kopfstedt and A. Beutel, "Application of Hybrid A* to an Autonomous Mobile Robot for Path Planning in Unstructured Outdoor Environments, " ROBOTIK 2012; 7th German Conference on Robotics, Munich, Germany, 2012, pp. 1-6.

[7] P. E. Hart, N. J. Nilsson and B. Raphael, "A Formal Basis for the Heuristic Determination of Minimum Cost Paths," in IEEE Transactions on Systems Science and Cybernetics, vol. 4, no. 2, pp. 100-107, July 1968. DOI = 10.1109/TSSC.1968.300136.

[8] Mark de Berg, Otfried Cheong, Marc van Kreveld, and Mark Overmars. 2008. Computational Geometry: Algorithms and Applications (3rd ed. ed.). TELOS, Santa Clara, CA, USA. DOI=10.1007/978-3-540-77974-2

[9] Rajamani, R. (2006). Vehicle Dynamics and Control. DOI = 10.1007/0-387-28823-6.

[10] S. Sedighi, D. Nguyen, K. D. Kuhnert, “A New Method of Clothoid-Based Path Planning Algorithm for Narrow Perpendicular Parking Spaces”, in press.