IMPLEMENTATION OF A SENSE AND AVOID SYSTEM FOR UNMANNED AERIAL VEHICLES

J.-B. Park, P. Vörsmann
Institut für Luft- und Raumfahrtsysteme, Technische Universität Braunschweig
Hermann-Blenk-Str. 23, 38108 Braunschweig, Germany

ABSTRACT

This paper deals with the issues in developing a system for *Sense and Avoid* for Unmanned Aerial Vehicles (UAV). Since 2001 the Institute of Aerospace Systems of the Technische Universtität Braunschweig (TU Braunschweig) has been developing various UAVs for missions such as airborne geosciences, observation of volcanic activity and plant security. In order to fulfil a mission reliably, the necessity of implementing a Sense and Avoid system for UAV becomes increasingly important. This implementation is the prerequisite for the autonomy of such aircraft. The realization of autonomy implies not only the detection of obstacles (static as well as dynamic ones) but also the implementation of feasible algorithms for subsequent flight path planning for the aircraft.

Following this aim, the paper first gives a brief introduction of the current challenges in developing a *Sense and Avoid* system on the basis of the UAVs of the TU Braunschweig. Then, the designated UAV and the measuring instrument are presented. The next section demonstrates the structure of the simulation for the generation of an admissible flight path amidst obstacles. It is shown how information about detected obstacles is used to generate an occupancy map by means of distance transformations. Further, the A* algorithm is briefly introduced. Besides these approaches, an attempt to generate a feasible flight path is made. This approach has in its methodology some similarities to the Traffic Alert and Collision Avoidance System (TCAS). Finally, the paper ends with a discussion of the results and intended methods toward a *Sense and Avoid* system.

NOMENCLATURE

A*	= Search algorithm	w	= Velocity in z
B	= Bézier curve	WP	= Waypoints
D	= Distance	x	= Position in x
E	= Edge of Graph G	y	= Position in y
f(ν)	= Cost function	z	= Position in z
G	= Graph, Goal	α, β	= Mounting angle of the laser range finder
g(ν)	= Cost to node		
h(ν)	= Heuristic		
O	= Obstacle	ν	= Node
P	= Bézier points	γ	= Climb angle
S	= Starting point	χ	= Azimuth angle
T	= Turning point	$_k$	= Kinematic frame
t	= Time, parameter in Bézier curve	$_g$	= Earth-fixed frame
u	= Velocity in x		
v	= Velocity in y	$_{ob}$	= Obstacle
V	= Kinematic Velocity		

1. INTRODUCTION

Numerous UAVs have been developed at the Institute of the TU Braunschweig since 2001. The controlling of these aircraft is based on an on-board computer and adequate ground station software. The on-board computer is responsible for the flight control and the ground station software enables the UAV to a fly predefined flight path automatically. Closer information can be found in [9].
Besides the above-mentioned automatic flight, it is evident that the ability to detect obstacles and to generate an appropriate flight trajectory is essential in granting an autonomous flight.

Normally, in mission planning the operator has information about the terrain to be investigated and based on this fact he will decide where the UAV can fly without any collision. However, it is also possible that the UAV may encounter unpredicted obstacles while executing the mission. It is obvious that a collision of the aircraft with obstacles is a potential danger not only for the UAV but also for its counterpart and even for persons positioned around the site of collision. To avoid this, the aircraft should be able to detect obstacles and generate a feasible flight path to avoid them. The independent decision making how a collision-free flight path can be generated is the main aspect of the autonomy for UAVs.

Therefore, the focus of this paper is the investigation of the potential to implement a *Sense and Avoid* system whose detection of obstacles relies on a laser range finder solely. For this purpose, subsequent sections give an overview of the experimental aircraft as well as the measurement system. Additionally, diverse methods of approaching the problem of implementing a *Sense and Avoid* system are described.

2. UAV AND MEASURING INSTRUMENT

The intended UAV for the investigation is the P 200 shown in FIG 1. The wing span of this aircraft is 2 m with a maximum take-off mass of 5 kg of which a maximum of 1 kg is planned as payload. It has an average velocity of approximately 55 km/h and an endurance of more than 60 min.

Various sensors exist for analysing the terrain and generating a trajectory as applied in [1] with a laser sensor, in [6,8] by means of cameras. The measuring instrument chosen in this paper is a laser range finder from Opti-Logic Inc. shown in FIG.2. Conventional laser range finders applied in the filed of robotics and terrain analysis measure their environment mainly in 2D or in 3D [10].

FIG. 1: Experimental UAV P 200

Such devices are not suitable for this specific case, because the masses of such devicees are not less than 1 kg. However, the laser range finder for this investigation has a mass of only 230 g. Because the laser range finder in FIG. 2 measures only one point, the challenge in implementing a *Sense and Avoid* system is to find a reliable trajectory based on this point measurement. The specifications of the laser range finder are summarized in TAB 1.

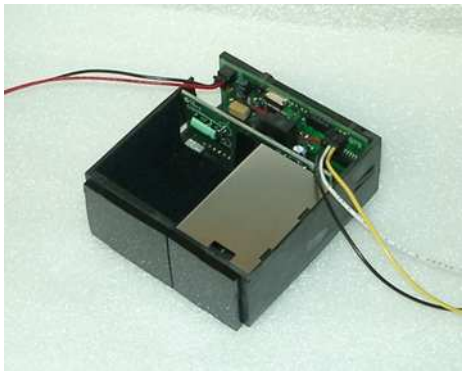

FIG. 2: Laser range finder RS 400 [11]

Power	7~9 Volts DC
Protocol	RS232
Laser Wavelength	905 nm +/- 10 nm (IR)
Laser Divergence	5 x 5 cm at 100 m
Accuracy	+/- 1 m
Measuring Range	max. 360 m
Measuring Frequency	10 ~ 200 Hz

TAB. 1: Specifications of the laser range finder

3. SIMULATION MODEL

Prior to real flight tests with the UAV, a simulation model was programmed in MATLAB. The purpose of this simulation is to obtain a first glance how a laser range finder will operate and how this device can be implemented in the UAV. Because of the fact that the applied laser range finder is able to detect only a single point as an obstacle, the assumed scenario after the possible detection is to initiate a steady turn. This means that once a sensed point is declared as an obstacle, the UAV initiates a steady turn and detects further obstacles or no other one depending on the environment. The acquired information is the basis of the following algorithms in finding a trajectory. In this context, the model consists of two parts described in FIG 3. At the beginning, both the flight path and the obstacles are generated simultaneously. Diverse strategies are investigated consecutively. The occupancy map, as the name suggests, is a method that gives a clear overview of the position (occupation) of the obstacles in the terrain. Parallel to the mapping, sampling based algorithm such as Rapidly-Exploring Random Tree (RRT) or the A* algorithm can be applied to find a flight path.

In addition, it should be investigated if the search for a collision-free flight path can be achieved by applying predefined flight path patterns depending on the distance to detected obstacles. This strategy may lead to a less time-consuming and more simple algorithm.

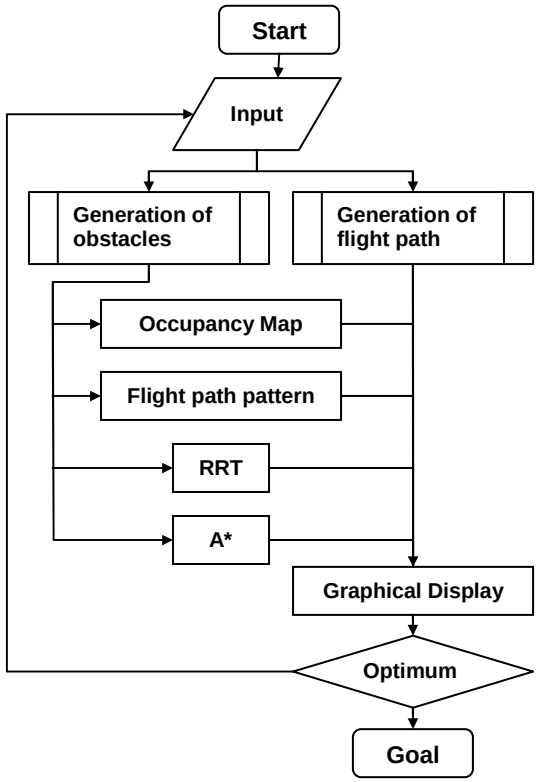

FIG. 3: Structure of the simulation model

The position of the UAV can be described in an earth-fixed frame as in equation (1). The equation can be simplified if it is assumed that the influence of the wind can be neglected and that the climb angle γ is equal to 0,.

$$(1) \qquad \begin{bmatrix} \dot{x}_g \\ \dot{y}_g \\ \dot{z}_g \end{bmatrix} = \begin{bmatrix} u_g \\ v_g \\ w_g \end{bmatrix} = \begin{bmatrix} V \cdot \cos\gamma \cdot \cos(\chi(t)) \\ V \cdot \cos\gamma \cdot \sin(\chi(t)) \\ -V \cdot \sin\gamma \end{bmatrix}$$

Furthermore, for the appropriate simulation of a steady turn, it can be assumed that the UAV P 200 executes the turn with a rate of 18°/s in the azimuth angle χ. By integrating equation (1) for each time interval, the position for distinct time can be formulated as in equation (2) – (4).

$$(2)\qquad \int_{x_i}^{x_{i+1}} dx = \int_{t_i}^{t_{i+1}} V \cdot \cos\gamma \cdot \cos(\chi(t))\, dt$$

$$(3)\qquad \int_{y_i}^{y_{i+1}} dy = \int_{t_i}^{t_{i+1}} V \cdot \cos\gamma \cdot \sin(\chi(t))\, dt$$

$$(4)\qquad \int_{z_i}^{z_{i+1}} dz = -\int_{t_i}^{t_{i+1}} V \cdot \sin\gamma\, dt$$

FIG. 4 accounts for the orientation of the detected obstacle O_i at the distance D_i with respect to the UAV. Unlike the conventional notation of α_i and β_i for the angle of attack and sideslip angle, these angles describe the mounting angle of the laser range finder. In the case that the device is not mounted along the longitudinal axis, these angles should be considered.

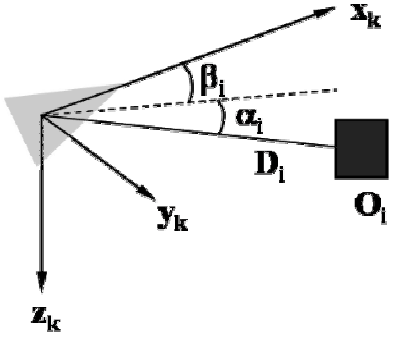

FIG. 4: Orientation of obstacle O_i with respect to the UAV

The position of the obstacle can be transformed from the kinematic frame k to the earth-fixed frame g with the matrix M_{gk} as in equation (5). Under the assumption that the laser range finder is mounted along the longitudinal axis, both mounting angles can be neglected and the position is calculated as in equation (6).

$$(5)\qquad \begin{bmatrix} x_{g,ob} \\ y_{g,ob} \\ z_{g,ob} \end{bmatrix} = \underline{\underline{M}}_{gk} \cdot \begin{bmatrix} D_i \cdot \cos\alpha_i \cdot \cos\beta_i \\ D_i \cdot \cos\alpha_i \cdot \sin\beta_i \\ -D_i \cdot \sin\alpha_i \end{bmatrix}$$

$$(6)\qquad \begin{bmatrix} x_{g,ob} \\ y_{g,ob} \\ z_{g,ob} \end{bmatrix} = \begin{bmatrix} D_i \cdot \cos\gamma \cdot \cos\chi \\ D_i \cdot \cos\gamma \cdot \sin\chi \\ -D_i \cdot \sin\gamma \end{bmatrix}$$

Additionally, the result of the simulation, shown in FIG. 5, takes into account the simplified boundary conditions adequately (climb angle $\gamma = 0$, velocity = const.). The UAV starts at the position (150,10,5) and initiates a steady 180° turn. As it flies, it continuously detects diverse obstacles which are marked with a black points and black lines representing the imaginary laser beam of the laser range finder. In the case that no obstacles are sensed, the beam of the laser range finder is marked with red lines.

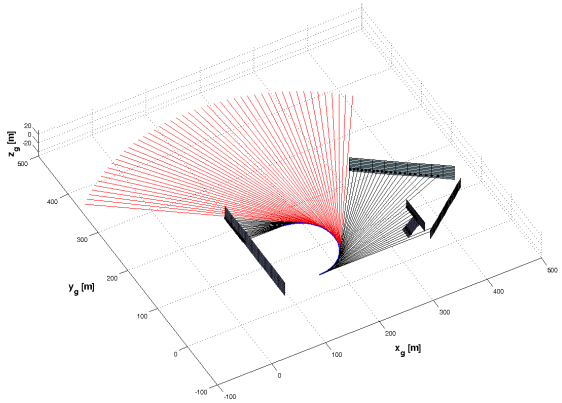

FIG. 5: Simulation of flight path

For better understanding refer to FIG. 6 that shows the x_g – y_g plane projection of the simulated flight path.

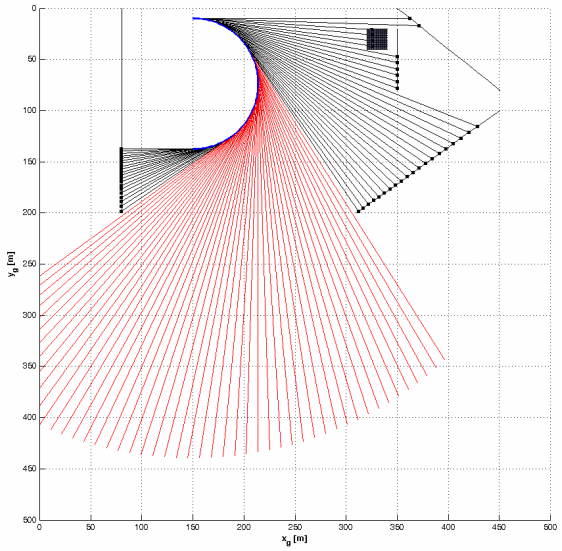

FIG. 6: Projection of the flight path

In this simulation the measuring frequency of the laser range finder is set to 10 Hz. The simulation gives an overview how the UAV equipped with the range finder detects obstacles. The information about the coordinates of the obstacles is essential for the methods in finding collision-free trajectories which will be shortly introduced in subsequent sections.

Besides the current features of the simulation, it is planned to expand the program by various sources of perturbation. As taking into account this sources the simulation can represent the environment more precisely.

4. FLIGHT PATH GENERATION

After acquiring information about the position of obstacles, a collision-free flight path is ought to be generated. There are numerous methods such as the distance transformation, A* search algorithm and flight path generation by means of Bézier curves.

4.1.1 Distance Transformation

The distance transformation is widely used in medical image processing and informatics [3]. The concept of this approach is to build a reference map of the environment of consideration. The reference map is also called graph G and consists of nodes and edges – named as N and E according to [4]. Each node possesses different values depending on the occupation through obstacles. The objective of this transformation is to acquire the distances from each pixel to its neighbour pixels with respect to the information about occupied nodes. This procedure results in a so called occupancy map showing endangered regions that the UAV has to avoid.

As already mentioned, the distance of each node is calculated successively. By doing this, different distances (also called metrics) are applied. Each metric has a different perspective in defining the distance between two distinct points. These metrics are denoted as the *Chessboard* distance, the *City-Block* distance and the *Euclidean* distance. The mentioned distances are shown in equation equations (7) – (9) for two points (x_1,y_1) and (x_2,y_2) in a plane. The distances D_{chess} and $D_{cityblock}$ do not seem to be optimal. The reason is that D_{chess} chooses only the maximum of the absolute value between the horizontal and vertical direction while $D_{cityblock}$ supposes only rectangular movements from one waypoint to the other waypoint. Contrary to this, $D_{Euclidean}$ calculates in direction of *line-of-sight* which is adequate for the implementation.

(7) $$\mathbf{D_{chess}}((\mathbf{x_1},\mathbf{y_1}),(\mathbf{x_2},\mathbf{y_2})) = \mathbf{max}(|\mathbf{x_1}-\mathbf{x_2}|,|\mathbf{y_1}-\mathbf{y_2}|)$$

(8) $$\mathbf{D_{cityblock}}((\mathbf{x_1},\mathbf{y_1}),(\mathbf{x_2},\mathbf{y_2})) = |\mathbf{x_1}-\mathbf{x_2}|+|\mathbf{y_1}-\mathbf{y_2}|$$

(9) $$\mathbf{D_{euclidean}}((\mathbf{x_1},\mathbf{y_1}),(\mathbf{x_2},\mathbf{y_2})) = \sqrt{(\mathbf{x_1}-\mathbf{x_2})^2+(\mathbf{y_1}-\mathbf{y_2})^2}$$

For the transformation, an imaginary window which is called the *distance mask* is defined. This window is built around the actual node depicted as X in FIG. 7. It calculates the horizontal, vertical and diagonal distances of neighbouring nodes to X. The concept is to execute a forward scan from the upper left corner to the lower right corner and a backward scan in opposite direction on the graph G.

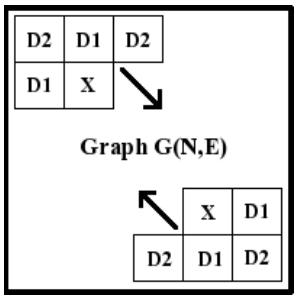

FIG. 7: Distance mask for forward and backward scan on the graph G

D1 represents the distance in horizontal as well as in vertical direction while D2 describes the diagonal distance from the current node X towards its neighbours. The result of this application is the occupancy map which is shown in FIG. 8 for each distance D_{chess}, $D_{cityblock}$ and $D_{Euclidean}$.

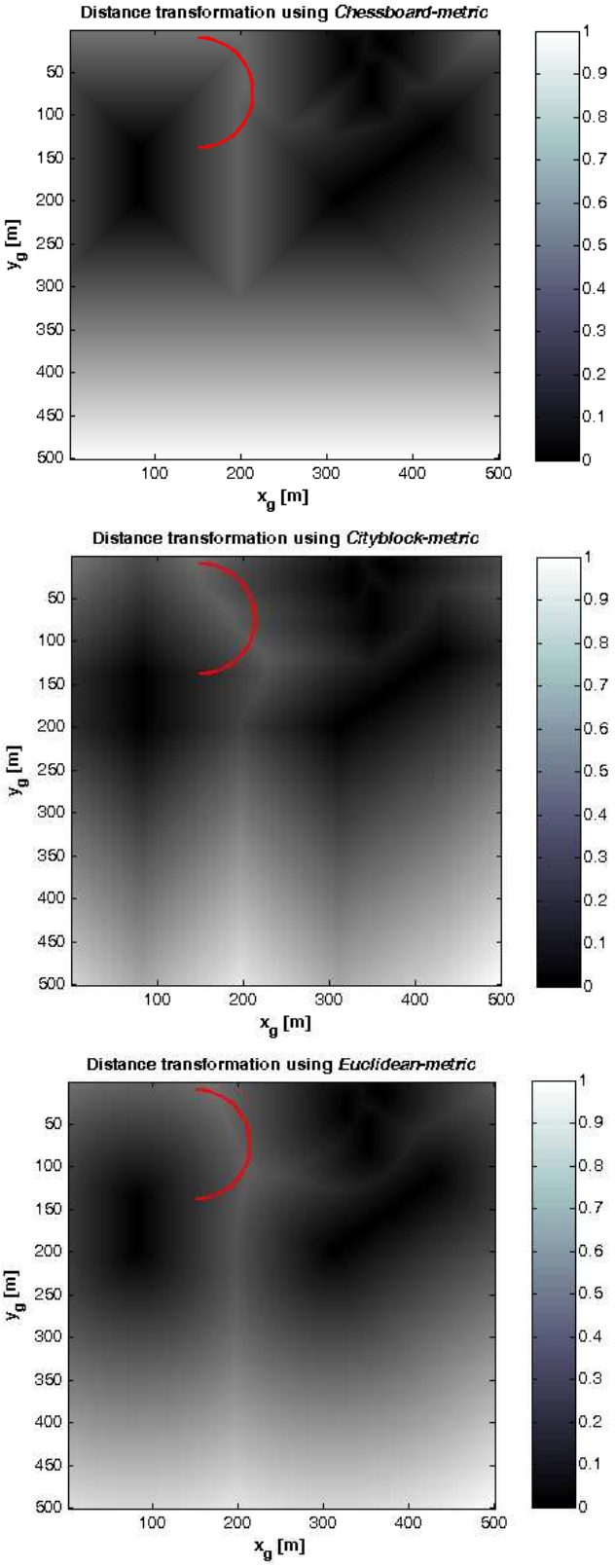

FIG. 8: Occupancy maps depending on different metrics

On each occupancy map, the simulated flight path is highlighted with a red curve. The areas where the obstacles exist are marked as the black regions with a value of 0 while the areas where no obstacles exist are the white one with a theoretical value of 1. However, the occupancy maps show in fact a shade from black to white. This can be interpreted as the influences of existing

obstacles. The darker the area in the occupancy map the higher is the danger of confronting with an obstacle. Therefore, the possible trajectory or passage is the consecutive movement along the white marked region or the area having approximate value of 1.
Unlike the representation by means of the *Chessboard* metric and *City-Block* metric, the *Euclidean* metric do not show overvaluation in the resulting shade from the transformation. Compared to FIG. 6, the *Euclidean* distance reproduces the surrounding area well.

The challenge in implementing distance transformations lies in the speed of the procedure to generate the mapping. Due to the fact that the on-board computing power is limited, it is one of the important aspects in implementing a *Sense and Avoid* system to adapt the transformation algorithm so that online application is possible. Another focus is the modification of the algorithm in a manner that only changed nodes and not the entire nodes are used to calculate the distance transformation on the next step.

4.1.2 The A* search algorithm

As the name implies, the objective of a search algorithm is the finding of a admissible path from a given start point to a desired end point in a discretized representative graph G of a region. This algorithm can be classified into an uninformed and informed one.

The breadth-first or deep-first methods are examples of uninformed search algorithms. Both build a tree-like representation of the region and begin their search at the root as depicted in FIG. 9.

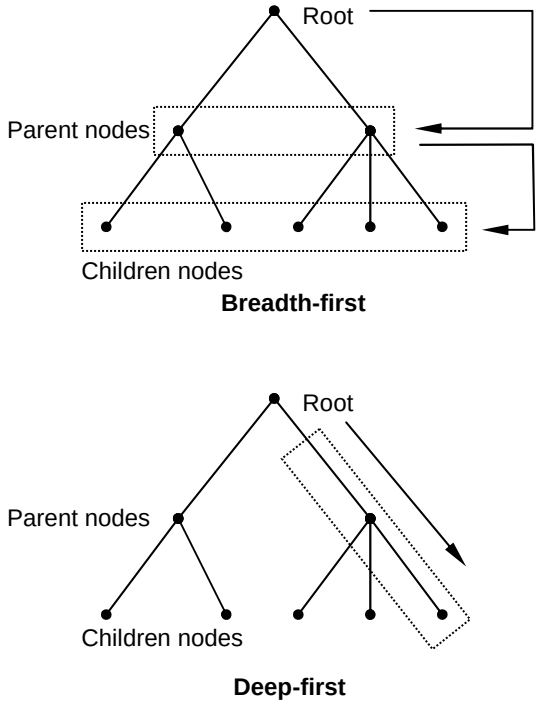

FIG. 9: Concept of breadth-first and deep-first algorithm

While the breadth-first method starts its search at the root of the tree and investigates the entire parent-nodes of one level first and then moves to the children-node, the deep-first method choose one parent node and continues its search from this chosen node to a children node of subordinates levels depending on predefined criteria.

Contrary to the uninformed search algorithm, the informed one has a specific preference in the direction of progress. This means that this method is goal-oriented and the numbers of considered nodes are probably less than those of an uninformed algorithm. In the following, the concept of the informed search algorithm A* is summarized. The possible implementation of this algorithm to the UAV will be one of the principal subjects of future investigation.

The alignment of the neighbouring nodes ν around the current node N in conjunction with the resulting cost g(ν) is depicted in FIG. 10.

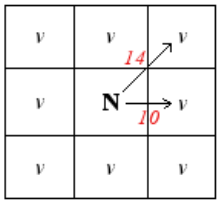

FIG. 10: Alignment of the nodes

The length of an edge between each node and the nodes are variable and has to be adapted depending on the situation. The number of nodes around the goal or obstacles should be more than those in free space. Assuming the UAV is located in the node N. Then, the function g(ν) accounts for the cost of a path from the current node N to the neighbouring node ν and can be defined by the distances from the previous section. The cost g(ν) in FIG. 10 has a value of 10 in horizontal direction and a value 14 in diagonal direction. Therefore, the ratio of both values yields the *Euclidean* distance.

Additionally, a heuristic function h(ν) is required for the implementation of this algorithm. Generally, a heuristic function is the estimation of the cost from a node ν to the goal [2,4,5]. Depending on how this estimation is made, the quality of the search is influenced. If the heuristic function assumes for example the *City-Block* distance that allows only rectangular movements, the result of this search will not be the optimum for a flight path because the UAV can also fly in diagonal direction. Summarized, the criterion for the A* algorithm to find a trajectory is the sum of g(ν) and h(ν) resulting in the total cost f(ν) in equation (10). The path is the sequence of nodes with the least cost f(ν) for each step.

$$f(\nu) = g(\nu) + h(\nu) \tag{10}$$

Based on this cost function, a simplified procedure of A* presented in FIG. 11. At the beginning, the information of the terrain with obstacle is known. In such a case, the algorithm is working *offline*. Since the UAV can sense also dynamic obstacles while approaching the waypoints, the application of this algorithm should work rather *online*. Because this implies a much complicated investigation, the offline search is presented for a conceptual understanding. The numbering of the nodes begins at the upper left corner and ends at the lower right corner. The search begins at the node 8 denoted as S. The obstacle is marked red and is located in the nodes 22, 27 and 28. The goal is the node 36 and is denoted as G. In the first step, the neighbouring nodes around S are considered

according to g(v) and h(v). In this example, g(v) and h(v) are calculated by the *Euclidean* and *City-Block* metrics, respectively. The rectangular movement is assigned with a value of 10 while the diagonal movement is assigned with the value of 14. Then, the total cost of node 9 is 80 and is calculated as follows. As node 9 is located horizontally to S, the cost g(v) is determined to 10 while h(v) is calculated by means of the *City-Block* distance and results in 70. The sum of g(v) and h(v) is the total cost for this node.

In the same manner, the cost f(v) of the other nodes around S are determined. In the next step, all neighbouring nodes of S are enlisted in a *priority-queue* (also called open set according to [4]) in ascending order depending on the cost function f(v) shown in TAB 2. It is important to consider the predecessor of each node as well and enlist them in the priority-queue.

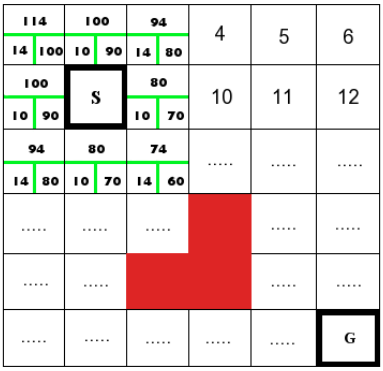

FIG. 11: Application of A* algorithm

Priority-Queue / Open Set				
Nodes	**f(v)**	**g(v)**	**h(v)**	**P**
S	-	-	-	-
15	74	14	60	S
14	80	10	70	S
9	80	10	70	S
13	94	14	80	S
3	94	14	80	S
7	100	10	90	S
2	100	10	90	S
1	114	14	100	S

TAB. 2: Priority-queue of the A* algorithm

Both the node S and the node with the least total cost f(v) are selected and moved to a newly defined closed set as in TAB. 3.

Closed Set				
Nodes	**f(v)**	**g(v)**	**h(v)**	**P**
S	-	-	-	-
15	74	14	60	S

TAB. 3: Closed set of the A* algorithm

The search continues then from the node 15 in the same way until either no nodes are remained in the open set or the goal G is reached. The resulting path in this example is shown in FIG. 13 and the completed closed set is given in TAB. 4. The trajectory is the sequence beginning in S and moving to the nodes 15, 16, 23, 30 and 36 consecutively.

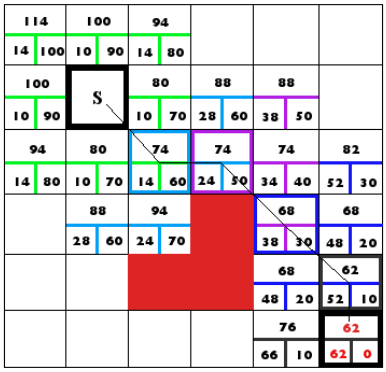

FIG. 12: Calculated trajectory with the A* algorithm

Closed Set				
Nodes	**f(v)**	**g(v)**	**h(v)**	**P**
S	-	-	-	-
15	74	14	60	S
16	74	24	50	15
23	68	38	30	16
30	62	32	30	23
36	62	62	0	30

TAB. 4: Completed closed set

This concept will be subject of further investigations. It is planned to adapt this algorithm adequately so that the generation of flight path can be performed *online*.

4.1.3 Flight path pattern

The methods introduced in the above sections are auspicious in finding a flight path. However, such procedures may be time-consuming for online generation and are even more complex when dealing with dynamic obstacles. Therefore, a different approach should be tested.

There exists effective methods in common air traffic for the avoidance of collision of aircraft as the Traffic Alert and Collision Avoidance System (TCAS) and the Ground Proximity Warning System (GPWS). Both systems are designed to alert the pilot of approaching airplanes and of crashing into the ground respectively. In the case of TCAS, the pilot is given either a Traffic Advisory (TA) or a Resolution Advisory (RA) depending on how close other airplanes are approaching. Whether a TA of a RA is selected depends on the constellation of the approach velocity, the bearing, altitude and range of the airplanes flying in the vicinity.

As conventional warning systems give the pilot recommendations for avoiding other airplanes or ground proximity, the UAVs can be provided with recommended trajectories based on predefined flight patterns. Similar investigations were carried out in [7]. The patterns depend on the information that is provided to the UAV. For example, the turning rate for avoiding obstacles can be classified differently depending on actual distances between the aircraft and obstacles. The different turning rates used in the simulation are listed in TAB. 5.

Category	Distance D	$d\chi/dt$
I	D < 150 m	18 °/s
II	150 m ≤ D < 300 m	9 °/s
III	300 m < D	5 °/s

TAB. 5: Category of initiated turns

In addition, Bézier curves are considered for the generation of a smooth trajectory. Bézier curves were applied primarily in the automobile industry for designing automobile bodies. Among various methods to generate such curves, the cubic Bézier curve is applied in this paper. FIG. 13 represents a path which consists of two consecutive curves and ends in the waypoint WP.

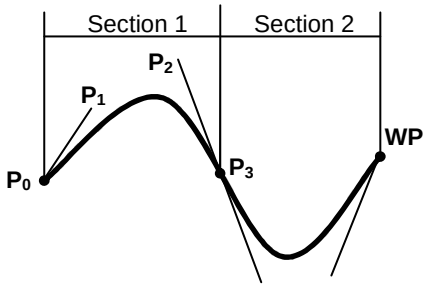

FIG. 13: Two consecutive Bézier curve

With four points $\mathbf{P_0}$, $\mathbf{P_1}$, $\mathbf{P_2}$ and $\mathbf{P_3}$, the curve in each section is given by the equation (11) with the parameter t ∈ [0,1]. Because a planar movement on the x_g-y_g plane is assumed, $\mathbf{B}(t)$ can be separated with $\mathbf{P_{i,x}}$, $\mathbf{P_{i,y}}$ (i = 0,1,2,3) into its x and y components, respectively.

$$\mathbf{B}(t) = \mathbf{P_0}(1-t)^3 + 3\mathbf{P_1}(1-t)^2 + 3\mathbf{P_2}t^2(1-t) + \mathbf{P_3}(1-t)^3 \quad (11)$$

This curve generation is conducted when no other subsequent obstacle is detected after an obstacle was noticed in the prior measurement. The Bézier curves are also generated in MATLAB. A simulated flight is shown in FIG. 14. The UAV starts at the point S (130, 20) and flies toward its waypoints which are marked as black square markers.

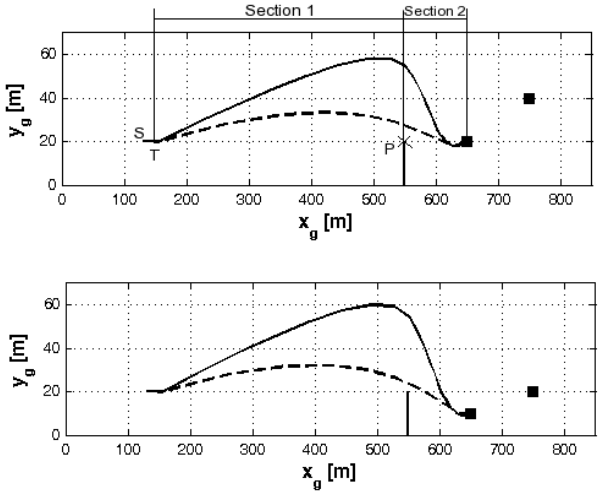

FIG. 14: Bézier curves for short $D_{ob\text{-}WP}$

Despite the distance between the obstacle and the first waypoint WP (650, 20) $D_{ob\text{-}WP}$ = 100 m, this is a small distance regarding the average velocity mentioned in section 1. From its actual position T (152, 20) an obstacle is located at P (550, 20). Because the distance between the aircraft and the obstacle D is 398 m, a turning rate of 5 °/s according to TAB. 5 is selected for initiatin g a turn. Then, the UAV measures from the new position (154, 20.026) again and does not detect further obstacles. At this point, the Bézier curves are applied. The generation of the trajectory can be conducted either for each section separately or for the whole section (section 1 + section 2). The solid line shows the case when the sections are considered separately while the dashed line represents the case in which the whole section is considered.

In contrast to FIG. 14, FIG 15 shows the Bézier curves when the distance between the obstacle and waypoint is long.

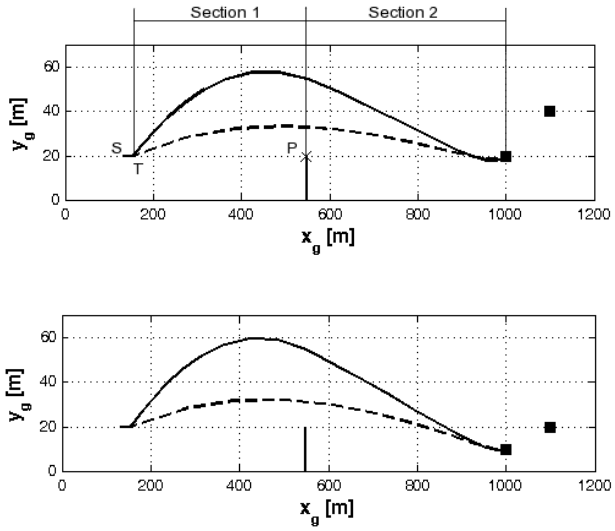

FIG. 15: Bézier curves for long $D_{ob\text{-}WP}$

The minimum *Euclidean* distance of the coordinates for the dashed Bézier curve toward P in the second diagram of FIG. 14 is approximately 18 m while the minimum for the dashed curve toward P in the second diagram of FIG. 15 is approximately 50 m. For each case, a safety zone such as in TCAS can be calculated to t_1 = 1.2 s and t_2 = 3.3 s. The higher the value of t the lesser is the danger of collision. At current, the threshold value is set to t = 2 s which should be investigated further. According to this, it can be inferred that the generation of Bézier curves based on one whole section (section 1 + section2) is appropriate for long $D_{ob\text{-}WP}$ while this consideration as one whole section brings the UAV in the vicinity of the obstacle for small $D_{ob\text{-}WP.}$ In this case, it is recommended to select the solid curve that generate two Bézier curves separately and connect them together. The length of the trajectory is longer but the danger of collision can be minimized.

Summarized, the generation of flight path pattern depends on the actual distance between the UAV and the obstacle as in TAB. 5, the distance between the obstacle and waypoint and the form of the Bézier curve itself with the definition of the safety zone as shown in FIG. 14 and FIG. 15. In further research the simulation is planned to be extended and correlated with data of flight test.

5. CONCLUSION AND OUTLOOK

The results of the applied methods show a potential for the implementation of a *Sense and Avoid* system. The simulation of a turn manoeuvre provided the information about the coordinates of obstacles. This information is needed when building an occupancy map by means of the distance transformation. The occupancy map reflects the environment in which the UAV operates well. Additionally, the A* algorithm was briefly introduced. This search algorithm is often applied in robotics to find a feasible path.

Since the occupancy map and the A* algorithm can be time-consuming for the online flight path generation, a method working in the way of conventional collision warning system such as the TCAS or GPWS was investigated. The result shows a well defined flight path avoiding the noticed obstacle. It should be investigated whether the occupancy map and the A* algorithm are qualified for the online path planning for UAVs and how feasible the generation of flight path pattern by means of Bézier curves is.

ACKNOWLEDGMENT

J.-B. Park would like to thank the DAAD (Deutscher Akademischer Austauschdienst) for granting the DAAD scholarship and Prof. Dr.-Ing. P. Vörsmann for thoroughly advising and supporting the current research at the Institute of Aerospace Systems of the Technische Universität Braunschweig.

REFERENCES

[1] Beard, R.W., Saunders, J.B.: Static and Dynamic Obstacle Avoidance in Miniature Air Vehicles, AIAA 2005-6950, 2005
[2] Bräunl, T.: Embedded Robotics, Springer, 2006
[3] Cuisenaire, O.: Distance Transformations: Fast Algorithms and Applications to Medical Image Processing, Ph.D. Dissertation, 1999
[4] Choset, H., et.al.: Principles of Robot Motion, MIT Press, 2005
[5] Latombe, J.-C.: Robot Motion Planning, Kluwer Academic Publishers, 1991
[6] Langelaan, J., Rock, S.: Navigation of Small UAVs in Forests, AIAA Guidance, Navigation and Control Conference, Providence, Rhode Island, 2004
[7] Richards, N., Sharma, M., Ward, D.: A Hybrid A*/Automaton Approach to On-Line Path Planning with Obstacle Avoidance. AIAA 2004-6229
[8] Schulz, H.W., et.al.: Affordable real time cartography using the MAV Carolo – limitations and prospects, AIAA 3rd "Unmanned Unlimited" Technical Conference, Workshop and Exhibit, Chicago, 2004
[9] Schulz, H.-W., et.al: The Autonomous Micro and Mini UAVs of the CAROLO-Family, AIAA Infotech@Aerospace 2005, Virginia, 2005
[10] Sukhatme, G.S., Schaal, S., Burgard, W., Fox, D.: Robotics, Science and Systems II, MIT Press, 2007
[11] http://www.opti-logic.com/industrial_rangefinders.htm (Last access: 05.July 2007)