VISION-BASED TARGET RECOGNITION AND AUTONOMOUS FLIGHTS THROUGH OBSTACLE ARCHES WITH A SMALL UAV

Florian-M. Adolf, Franz Andert, Lukas Goormann, and Jörg S. Dittrich

German Aerospace Center (DLR), Institute of Flight Systems
D-38108 Braunschweig, Germany

{florian.adolf, franz.andert, lukas.goormann, joerg.dittrich}@dlr.de

ABSTRACT

The challenge for unmanned aerial vehicles to sense and avoid obstacles becomes even harder if narrow passages have to be passed for which no precise a-priori position information is available. Inspired by recent UAV flight competitions, this work presents a vision-based approach to search for a narrow gate and to fly through it autonomously. The gate’s precise position has to be detected in order to avoid a collision. Using a GPS-based self localization, the camera alignment and the gate position are estimated simultaneously. The presented approach alters a set of waypoints to fly through the gate. All algorithms run onboard the vehicle such that the gate is passed autonomously. Results from flight tests are presented that underline the feasibility of the presented approach.

INTRODUCTION

Operations of unmanned vehicles in urban environments are of high interest for surveillance and reconnaissance tasks. However, autonomous and safe operations in such environments with plenty of obstacles are not simple tasks, but eventually necessary ones. Many small aerial systems that are suitable for urban operations do still rely heavily on the situational awareness of the remote operator. Since urban scenarios may require the vehicle to fly without line of sight, it becomes necessary that the vehicle has a sufficient situational awareness to avoid collisions with unanticipated obstacles. The challenge is to achieve sense-and-avoid capabilities close to ground and in cluttered terrain. This paper describes an approach for safe and precise navigation close to obstacles without user interaction based on onboard perception and reactive planning.

Within the aerial domain, unmanned rotorcraft have the great advantage to fly highly agile maneuvers at moderate speed while still being able to hover or even fly sideways and backwards. Helicopters have the ability to fly at very low altitudes in complex environments with a confined operating space—which makes them the platform of choice for urban scenarios.

Unmanned aerial systems have to manage a variety of difficulties. The payload capacity for sensors, computing equipment and energy supply is limited to a few kilograms which does not allow to completely copy approved arrangements, e.g. used for autonomous cars. In addition to that, full three-dimensional maneuverability is provided and must be also covered.

In order to reduce computational load, some research activities focus on sole offline computation of the mission execution process. However, during mission execution, a major drawback is the requirement of safe and secure high-speed data links to the ground control station. Additionally, a wireless connection can be jammed, especially behind obstacles and is slow by means of comparison to wired links. One pragmatic approach is the use of optical fiber connection to operate in the vicinity of a ground control station [1]. This setup allows the usage of smaller vehicles since a high bandwidth data link is provided, dispensing with complex onboard computations. However, cable handling is very complicated and highly restricts the flexibility.

Most research focuses on the development of aerial robots by increasing the system autonomy. Such intelligent vehicles demand operator commands as high-level task inputs (e.g. adjacent waypoints), which makes a continuous data link dispensable during flight. For example, current research covers obstacle avoidance behaviors [2, 3], mapping [4] and map-based path planning [5, 6], onboard route planning [7], autonomous landing [8], detection and tracking of specific targets [9] or following moving ground vehicles [10].

The next sections present an approach that focuses on onboard environment perception and improved intelligence. The test bed is an auto-piloted small helicopter equipped with vision sensors and full computational power for image processing and mission control applications. The following section describes the details of a scenario for which the unmanned vehicle is used. Throughout this paper, dedicated sections address the test vehicle, the vision-based gate perception, and the reactive planning component. To underline the feasibility of the proposed approach, experimental results are shown subsequently and concluded at the end.
SCENARIO AND PROBLEM DESCRIPTION

As a methodology demonstration, the paper presents a solution to a specific mission scenario as illustrated by figure 1. A gate with a size of $6 \times 6$ meters is assembled on a flight field, and the helicopter shall fly through it without any operator interaction. The obstacle size and appearance is previously known as well as its approximate global position in GPS-coordinates.

![Figure 1. Obstacle gate for autonomous flight demonstration.](image)

This scenario addresses problems that are often negligible when flying in terrain with a lot of free space, but of high importance for flights into narrow passages, under bridges or through windows. The major challenge to solve the task setting is due to gate and vehicle properties. A helicopter has to be guided with a deviation of much less than three meters through the gate. This requirement disqualifies classical waypoint navigation or map-based path planning strategies since GPS does not provide the demanded accuracy of vehicle localization with respect to geo-referenced obstacles. Hence, the vehicle must “see” the environment to increase knowledge about true object positions and to compute its own position relative to them. Environmental sensing leads to two problems that must be solved simultaneously:

- gate position estimation relative to the camera, and
- camera alignment estimation relative to the vehicle.

While the first problem is understandable intuitively, the second problem appears as a result of preliminary flight tests. On the one hand, alignment errors are caused by uncertainties in manual camera orientation measuring. On the other hand, navigation sensor uncertainties, drift, and imprecise calibration lead to biases of the estimated vehicle position where the camera alignment is initially related to. Since flight instructions must fit with the navigation solution, a very accurate relative orientation between sensor and navigation system is required. Due to drift, this camera alignment changes over longer time periods and cannot be calibrated before flight. To avoid separate calibration flights shortly before the proper task, it is estimated while approaching to the gate, based on image-based gate position measurements. Both problems are solved in the image processing section of this paper.

Beside gate recognition abilities, the complete flight consists of multiple intelligent behaviors from autonomous take-off to landing. For example, basic behaviors are waypoint instructions to search for the gate and to fly through. To combine the behaviors, a mission schedule combines

- automatic take-off and landing,
- vehicle guidance to initially given, but only roughly known waypoints,
- gate crossing instructions if an accurate true position has been estimated,
- failure consideration due to unconfident gate detections, and
- the transition to other tasks, e.g. the handling of multiple gates.

Details are explained in the dedicated section of this paper.

HARDWARE EQUIPMENT

In order to cope with sensor and software problems, it was decided to put together a demonstrator of such a combined sensing and planning system for urban operations. DLR’s “Autonomous Rotorcraft Testbed for Intelligent Systems” (ARTIS) was selected as the test UAV for these experiments. ARTIS is an inexpensive multi-purpose research platform for unmanned flight [11]. There are two ARTIS helicopters available for flight tests with a gross weight of 13 kg and 25 kg, respectively. The smaller vehicle of the two, the midiARTIS helicopter, is shown in figure 2. The avoidance sensor described in this paper can be fitted to each of the vehicles. However, all testing of the sensor has been done on the presented helicopter, which has been equipped with a dedicated image processing computer and a calibrated monocular color camera for these experiments.

![Figure 2. Research UAV ARTIS: a combustion-engine-powered autonomous helicopter, 1.9m rotor diameter, 13kg gross weight.](image)
IMAGE PROCESSING

While flying towards the initially given gate position, the position relative to the camera is estimated with image processing techniques. At the same time, sensor misalignment is estimated by comparing the vehicle trajectory from navigation with visual odometry during flight.

The presented algorithms enable these tasks without the need of separate calibration flights. Output is a gate position and orientation estimation relative to the navigation reference. This position is used to guide the vehicle through the gate on a determined heading. Due to drift errors in relative navigation, it is recommended to completely repeat the gate localization and sensor misalignment estimation process in case another gate shall be passed.

The approach has similarities with simultaneous localization and mapping techniques [12] where environmental measurements are used not only to locate object positions, but also to re-locate the vehicle so that actual sensing information aligns with a map built from previous measurements. In the presented scenario, the relative vehicle position in the world is of major importance without regarding the global world map consistency. Hence, the world is re-located so that the sensor readings match with the map while the vehicle position is kept. An advantage is that no changes of vehicle navigation and guidance are needed. It is easy to readjust this image-based application for other vehicles, including those with proprietary auto-piloting systems having limited modification options.

Gate Recognition

Each gate is constructed with three flags at each post for automatic recognition. Figure 3 shows the main steps used to locate the flag positions that will be the basis for locating the vehicle relative to the gate. The image processing approach to find and locate the gate flags consists of the following steps:

- Segmentation of the color image (a) to produce a binary image (b). Red spots are filtered using thresholds (hue, saturation, value) applied to the input image.
- Merging of neighboring white pixels of the binary image to clusters that represent individual objects.
- Cluster filtering that will remove too small or too large clusters and those who do not have an approximate squarish shape (c).
- Finding lines with exactly three clusters on it or close to them, and that are almost vertical in the image, and out of these similar line couples where the segment lengths between the clusters are nearly equal. If more than one line couple is found, the smaller ones are omitted (d).
- Generating six “Features” defined by the remaining clusters. These features mark the positions of the six flags in the image (e).

As described in the later sections, these features are the basis to estimate the gate position. Furthermore, the features are tracked [13, 14] so that the detection process has not to be repeated for each new image. If features get lost during the tracking process, and a new detection will result in completely different flag positions, it is assumed that these features belong to a different gate that is regarded separately. Within this paper, the term image sequence refers to the same instance of a gate.

Estimation of the Gate Position

Now, the orientation of the gate relative to the camera is estimated for images where all six flags have been found. It is solved with camera resectioning techniques as used in photogrammetry. Internal camera parameters such as image center, focal length, and lens distortion are static and known through a preceding calibration. Normalized image coordinates and the pinhole camera model are used to estimate the external orientation of the camera with respect to the gate.

Figure 4 shows how the image feature points are related to the real flag positions. The perspective transformation for corresponding points \( \mathbf{p}_j \in \mathbb{R}^3 \) and \( \mathbf{q}_j \in \mathbb{R}^4 \) in homogenous coordinates is

\[
\lambda_j \mathbf{p}_j = [\mathbf{R}_{q_j}, \mathbf{t}_{q_j}] \mathbf{q}_j
\]
with a scale factor $\lambda_j$ and the camera orientation given by rotation $R_{q_i}$ and translation $t_{q_i}$ of the $i$-th image. Following [8], this equation can be transformed into

$$\begin{bmatrix} p_i \end{bmatrix} e_3^T \begin{bmatrix} 1 \end{bmatrix} \begin{bmatrix} R_{q_i} & t_{q_i} \end{bmatrix}q_j = 0 \quad (2)$$

with $e_3 = (0, 0, 1)^T$. This equation is valid for all corresponding points in an ideal case. Due to noise, there are uncertainties which leads to the problem finding $R_{q_i}$ and $t_{q_i}$ that minimizes the error

$$E(R_{q_i}, t_{q_i}) = \begin{bmatrix} p_i \end{bmatrix} e_3^T \begin{bmatrix} 1 \end{bmatrix} \begin{bmatrix} R_{q_i} & t_{q_i} \end{bmatrix}q_j. \quad (3)$$

A solution to this problem is presented in [8]. It assumes coplanar points $q_j (j = 1, \ldots, 6)$ and consists of a linear initialization step that produces approximated results of $R_{q_i}$ and $t_i$, followed by a non-linear multivariate Newton-Raphson iteration that produces a more accurate result by minimizing the reprojection error.

The origin of the gate coordinate system marks the point where the vehicle shall pass the gate in $x$-direction. Hence, $R_{q_i}$ and $t_{q_i}$ give a transformation between the actual camera orientation and the desired target.

### Calculating the Global Gate Position

This section describes the process to calculate the global position of the gate. It is based on the camera position relative to the gate as described above and uses information about camera assembly and vehicle navigation required for the coordinate transformations.

Let $R_{g_i}$ and $g_i$ denote the helicopter orientation relative to the gate estimated using the $i$-th image. With known camera alignment $R_{hc}$ and $t_{hc}$, it is

$$R_{g_i} = R_{hc} R_{q_i}, \quad \text{and} \quad g_i = R_{hc} t_{q_i} + t_{hc}. \quad (4)$$

Using the corresponding helicopter orientation $R_{h_i}$ and position $h_i$ from the navigation filter, the global target $R$, $t$ is determined by

$$R = R_{h_i} R_{g_i}, \quad \text{and} \quad t = -R_{h_i} g_i + h_i. \quad (5)$$

Based on this rotation, the desired direction angle $\Psi$ to fly through the gate is determined by the rotation matrix elements. It is

$$\Psi = \text{atan}2(R_{21}, R_{11}). \quad (6)$$

### Camera Alignment Optimization

Experiments have shown that this is only roughly valid, see the results presented later. While the helicopter approaches the gate, the estimated gate position moves dependent on the helicopter position. This strongly suggests systematic errors since the gate does not move and is fixed to the ground. These errors are not large with few meters and may be negligible when navigating in large canyons. In the case of narrow passages as represented by the obstacle gates, it has become indispensable to perform a calibration of the external camera orientation with respect to the vehicle localization. Especially rotational errors must be minimized since they induce an incorrect gate orientation and too large position errors when viewing the gate from far distances.

![Figure 5. Compensating errors in camera and helicopter alignment by estimating the transformation between camera and navigation solution.](image-url)
There will be an orientation error between the true helicopter position and its localization by the navigation filter. This error may change over time due to drift errors, but is assumed to be constant for a short time period, i.e. for a single gate passage. Differences between the world used here and real world coordinates, e.g. from obstacle maps, are assumed to be static.

Now, sensor fusion methods are applied, see (d). Both camera and navigation filter provide localization and with that, relative movements in the same reference system. The paper shows an approach to compensate the camera misalignment in order to get an accurate gate localization.

The final conclusion is shown in (e) where the estimated orientation between camera and navigation coordinate system, and the orientation between navigation and world coordinates are successively used to get the camera position in the world and with that, the gate position in the world. It is not referenced to a true world of previously known maps, but to the navigation coordinate system, which is necessary to fly through the gate. The true helicopter position is never used since all commands refer to the navigation coordinate system.

The camera alignment optimization approach is based on the idea that an ideal camera alignment would lead to a projection of all estimated gate positions from different vehicle locations to the same global coordinate as illustrated in figure 6. The problem is solved by searching for a camera alignment \( R_{hc} \) and \( t_{hc} \) that minimizes the deviation of global gate positions taken so far from different locations.

### Figure 6. Minimizing the gate position estimation uncertainty (a) by finding the correct camera alignment (b).

After taking \( n \) valid images from different vehicle positions, a point cloud with \( n \) points is determined using the global gate translation from equations 4 and 5 for each point. The error vector, denoted by \( \sigma = (\sigma_x, \sigma_y, \sigma_z)^T \), represents the standard deviations of the point cloud in geodetic \( x \), \( y \), \( z \)-direction.

Let the camera alignment be determined by three position elements \( x, y, z \) and three rotation angles \( \omega, \phi, \kappa \). For fixed gate measurements \( \{ R_{g_i}, g_i, i = 1, \ldots, n \} \) and their corresponding vehicle positions \( \{ R_{h_i}, h_i, i = 1, \ldots, n \} \), the point cloud of resulting gates is only dependent on the camera alignment, i.e. \( \sigma : \sigma(x, y, z, \omega, \phi, \kappa) \). Tests have shown that the initial alignment translation \( (x, y, z) \) can be measured with comparatively high accuracy and that the camera roll angle \( \phi \) is of low importance to locate the gate. The gate position mainly depends on the camera pitch \( \omega \) and the yaw angle \( \kappa \). Only they are optimized, and it is \( \sigma : \sigma(\phi, \kappa) \). The optimal camera alignment will result in a minimum of \( \sigma \), which is approximated using a multivariate Newton-Raphson iteration.

The algorithm uses the Jacobian matrix \( J_\sigma \) of \( \sigma \)

\[
J_\sigma = \begin{pmatrix}
\frac{\partial \sigma}{\partial \phi} \\
\frac{\partial \sigma}{\partial \kappa}
\end{pmatrix}
= \begin{pmatrix}
\frac{\partial \sigma_x}{\partial \phi} & \frac{\partial \sigma_y}{\partial \phi} & \frac{\partial \sigma_z}{\partial \phi} \\
\frac{\partial \sigma_x}{\partial \kappa} & \frac{\partial \sigma_y}{\partial \kappa} & \frac{\partial \sigma_z}{\partial \kappa}
\end{pmatrix}, \tag{7}
\]

where the derivatives of single values \( \phi \) and \( \kappa \) are approximated using the difference quotients

\[
\frac{\partial \sigma}{\partial \phi} \approx \sigma(\phi + \epsilon, \kappa) - \sigma(\phi, \kappa) / \epsilon, \tag{8a}
\]

\[
\frac{\partial \sigma}{\partial \kappa} \approx \sigma(\phi, \kappa + \epsilon) - \sigma(\phi, \kappa) / \epsilon, \tag{8b}
\]

with a small \( \epsilon = 0.001 \).

Now, the iterative approximation starts with \( \phi_0 \) and \( \kappa_0 \) from initially given camera alignment angles, and the \( n \)-th iteration is

\[
\begin{pmatrix}
\phi_{n+1} \\
\kappa_{n+1}
\end{pmatrix} = \begin{pmatrix}
\phi_n \\
\kappa_n
\end{pmatrix} - \begin{pmatrix}
J_{\sigma,n}^T J_{\sigma,n}
\end{pmatrix} \sigma(\phi_n, \kappa_n) \tag{9}
\]

where \( J_{\sigma,n}^T \) is the Moore-Penrose pseudo inverse of the Jacobian, with the partial derivatives calculated for the single values \( \phi_n \) and \( \kappa_n \). Termination conditions are a maximal number of iterations or \( \| J_{\sigma,n} \sigma(\phi_n, \kappa_n) \| < \epsilon \) with a threshold parameter, e.g. \( \epsilon = 0.001 \).

Finally, a new camera alignment \( R_{hc} \) and \( t_{hc} \) and a target position as done in the equations 4 and 5.

### GATE-PASSING BEHAVIOR

The previous section highlighted a vision-based smart sensor concept that is capable of measuring the center position of the gate and its direction to fly through. This section presents a behavior-based approach that triggers the smart sensor to detect the gate, and to use its information to safely maneuver through the gate. Before the actual gate passing behavior is addressed, the necessary building blocks for mission planning and execution are highlighted.

### Offline Planning Step

The underlying mission scenario expects an operator planning a mission before the flight. He needs to specify an approximately known position and fly through direction of the desired gate. In this work, this type of offline planning process is based on previous work presented in [15]. The planning system generates an executable mission plan which is a sequence of high level behavior commands used for waypoint
navigation. Each command in the sequence is parameterized to match the operator’s specification, e.g. the exact waypoint position or the desired vehicle velocity. However, the effective waypoint parameters guiding the vehicle through the gate safely, remain unknown during this planning phase. Hence, an operator defines a mission plan for which the vehicle needs to determine the exact waypoints autonomously when the smart sensor detected the gate.

**Onboard High Level Control**

Based on the requirement of an online mission plan specialization, the onboard system has to fulfill a set of principle mission objectives. The system has to implement an Assistive Artificial Intelligence [16], comprising the knowledge and capabilities to solve the flight through the gate autonomously, while following the global mission plan (e.g. waypoints before and after passing the gate). This way, the complexity of the offline planning phase is reduced considerably, since the vehicle is required to implement a problem solver for an eased problem scope [17, 18].

Previous work in [19] implements a high level control architecture based on [20]. Generally, this architecture implements a set of predefined elementary flight behaviors, e.g. take off, land or hover to a position, as well as a set of high level behaviors that use elementary flight behaviors to achieve a higher mission goal, e.g. flying back to the starting point. In the context of this work the gate passing task is implemented as a deliberate behavior.

**Reactive Planning Step**

As mentioned before, the exact sequence of commands and parameters of each of the necessary elementary flight maneuvers cannot be known a-priori. Thus, the gate passing behavior applies the reactive planning principle [21] in order to generate and alter the flight maneuver parameters.

The planning objective comprises the position update using the smart sensor’s gate detection capabilities, and collisions with the gate poles have to be avoided. Since it is not possible for the sensor to detect the gate at every instant of time, three phases approach, detect and pass are proposed. These consecutive phases implement the following objectives:

1. **Approach**: Fly to a start position that is about 30m in front of the a-priori estimated gate location.
2. **Detect**: Attempt to detect the gate along a linear calibration path until the vision-based gate detection was successful. Whenever the gate was detected sufficiently well, alter the waypoint coordinates accordingly.
3. **Pass**: Once the gate is localized successfully, fly to a start position from which the gate passing will be started.

Hence, during the detection phase the waypoint position updates yield reactive plan sections that are generated online. Figure 7 illustrates the plan generation for each phase. First, the single line command ‘GM’ from the original mission plan is expanded into a set of elementary flight maneuvers (here: hovering to a position). The first set of commands generate the detection phase until the sensor returns the first detection result. In this example, the event comprises a detection confidence value that was high enough to trust the sensor input, but too low to start passing through the gate. Thus, another set of commands has to be generated attempting to detect the gate again. This is performed along the updated calibration path relative the newly detected gate position. This procedure will be repeated until the operator aborts or a high confidence value of the detection is available.

The three phases of the gate passing behavior are illustrated in figure 8. The illustrations show the scenario from a bird’s eye view with the vehicle and its flown path, future waypoints and the gate positions. Each gate is represented by its two posts, center and pass direction.

**Figure 7. Online plan expansion based on reactive planning, exemplified for the gate passing behavior.**
and approach to the gate up to the nearer waypoint \( S_3 \). In this way, images are processed to search for the true gate position (fig. 8b).

If the gate is visible and its position is estimated with a high confidence, two new waypoints \( S_4 \) and \( S_5 \) are added at a distance of \( d = 10m \) in front of and behind the gate. They will guide the vehicle through the gate (fig. 8c). Based on a estimated gate position \( t \) and direction \( \Psi \) (eq. 6), the effective set of waypoints is altered.

\[
S_{5/4} = t \pm d \left( \begin{array}{c}
\cos \Psi \\
\sin \Psi \\
0
\end{array} \right).
\]

If the search is not successful, another attempt is performed by flying back to \( S_2 \).

After a successful attempt of the gate passing behavior, a further execution of the remaining commands is performed. Thus, by passing through the gate the mission is either completed or the next gate can be approached (fig. 8d). The initial map is transformed by using the deviation between the initial and the estimated positions of the first gate. New waypoints \( S_3 \) and \( S_6 \) are set and the mission continues with the same procedure used for the first gate.

**Procedural Execution Model**

Similarly to the underlying system in [19], the gate passing behavior itself and the handling of the smart sensor are modeled using state charts. This way, model based testing procedures are possible before integration into the flight vehicle. Figure 9 shows the executable model for the gate passing behavior. It can be seen that more than three logical states are necessary to implement the three principle phases: approach, detect and pass. The gate passing behavior’s first state ‘Init’ performs the command expansion exemplified in figure 7. Once position \( S_2 \) has been reached, ‘Calibrate Start’ is set up as soon as the gate detection sensor is ready. The transition into state ‘Calibrate Stop’ occurs, once the calibration finishes and when point \( S_3 \) is reached. In this state it is decided whether a re-calibration has to be done or whether the flight through the gate can be started. In both cases, if the detection result is sufficiently confident, the waypoints \( S_2 \) to \( S_5 \) are moved to the correct location accordingly. State ‘Fly Through Start’ is entered if the detection returns trustworthy measurements. This state moves the vehicle towards the starting position \( S_4 \) in front of the gate such that the flight through the gate can be started from a minimized offset to the desired straight path through the gate. A flight to position \( S_5 \) is commanded for state ‘Fly Through Stop’ and the state transits to state ‘Fly Through Final Climb’ where the vehicle is commanded back to a safe mission height. In order to pass the gate, it might have been necessary to temporarily fly closer to the ground than desired. Thus, this last state avoids long periods of flying too close to ground.

Since computational resources onboard are limited, the gate detection is performed during the calibration behavior phase.

The model shown in figure 10 shows the effort to handle the smart sensor, here the gate detection component. This model is executed asynchronously to the gate passing model such that both, the gate passing behavior and this smart sensor handling, needs to implement measures for synchronization. Thus, the model approaches this logical synchronization problem using a set of acknowledgment events, as well as a decomposition into logical states comprising a general sensor setup phase (state ‘Stand By’), a configuration phase for gate detection (state ‘Gate Mission Init’), and a productive phase where the gate is detected effectively (states ‘Gate Mission Start’, ‘Gate Mission Stand By’, ‘Gate Mission Stop’).

**EXPERIMENTAL RESULTS**

The gate detection approach was tested using real test sensor data acquired during manually controlled flights. After optimizing the gate detection algorithms, the integrated overall approach was tested with full autonomous flights that include integrated vision, mission planning and flight control. To assess a correct assignment when multiple gates are visible, the test scenarios during the data acquisition in manual flights comprise two gates of equal appearance, located at a distance of 50 meters of each other. The autonomous tests assess the integrated overall approach comprising offline mission planning, plan execution, visual gate detection and reactive plan adaptation. The onboard avionics hardware comprises two PC-class computers, one for computer vision, one for planning and flight control. With exception to the offline
mission planning, every phase of the approach is computed onboard and performed without the operator.

**Gate recognition and localization**

Image processing, gate position and camera alignment estimation is developed and optimized using image sequences taken from manual helicopter flights. Different lighting conditions, camera shaking on the vehicle, navigation uncertainties and the realistic setup provide a case close to full autonomous flight. The quality of gate detection methods is determined with respect to the flight trajectory assuming that the vehicle flew through the gate centers.

The evaluation of the detection algorithm focuses on the accuracy of gate orientation estimation at any time. The paper presents processing results for most of the images where a gate is recognized. Figures 11–14 show plots of different measure values that are required for gate detection. Results are produced as follows:

- Both gate positions are calculated independently since a sequence cut is automatically performed when the first gate leaves the field of view during flight. The results are projected to frame numbers of the overall image sequence with both gates.
- For each gate, sequences with a minimum number of 20 frames are used to estimate the gate position. For each new image, an updated gate position is estimated.
- Maximal sequence size is 50. Older images are removed from the sequence and are not taken into account for position estimation.

Figure 11. Flight trajectory (blue) and estimated gate positions \((x,y)\) using a-priori camera alignment assumptions (left, green) and results when using the described camera alignment optimization algorithm (right, black).

Figure 12. Extract of figure 11 by focusing on the first gate (left) and second gate (right), respectively. Both plots show the results using initial and optimized camera alignment.

Figure 13. Gate estimation errors of the first (left) and second (right) gate during the image sequence, i.e. Euclidean distance between estimated gate position and the closest trajectory point.

Figure 11 shows a bird’s eye view on the flight trajectory and the estimated gate positions using the different algorithms. Flight direction is from the right to the left. The left plot shows the processing results when using the initially given camera alignment. All estimated gate positions are located left to the flight path, but this error decreases while approaching to the gate as later shown. The right plot shows the
results when applying the improved algorithm. While the first gate was detected with good quality with both algorithms, the performance of the camera alignment optimization algorithm is increased at the second gate (black markers). Figure 12 shows the same results with a higher resolution, focusing on both gates separately. The position estimation is better at the first gate, which is supposed to be due to a larger number of valid images taken in the vicinity of the gate, which means at a distance between 7 and 10 meters. At lower distances, the gate gets out of view. Camera alignment optimization has a good performance with an error of less than 0.5 meters to the reference flight path. In contrast to that, this was not achieved when using known camera alignment. The optimization provides stable results from image to image, which leads to a better qualification for autonomous flights. The performance is decreased at the second gate. Here, the optimized camera alignment is also better qualified than the initially given alignment. The position estimation errors in all three translational degrees of freedom are summarized in figure 13.

Figure 14 shows estimation charts of height and gate direction. The results show that the estimated gate height is suitable for autonomous flights. As a drawback, the desired flight direction has always some errors since the ideal case would produce a strictly horizontal spread of the markers. The observed direction deviation of five degrees for the first gate should be unproblematic.

Both, the offline image sequence evaluation as well as flight tests show that the camera misalignment compensation is necessary. It has turned out to be qualified and is installed on the onboard image processing computer to be used for autonomous flights.

Tests of the overall approach

Figure 15 illustrates the overall approach in a test scenario. The helicopter is instructed to the first waypoint (a) in order to start the gate search. After successful search (b), the vehicle adjusts the desired flight path by inserting the two waypoints in front of and behind the gate. Afterwards, the vehicle guides itself precisely through these waypoints (c) crossing the detected center position of the gate. Note that the gate passing behavior’s waypoint replanning has a constant runtime complexity (here: five waypoints).

Additional two-dimensional plots show the quality of gate detection from two different flights. For initial gate positions with uncertainties like in a real test, the helicopter performs a flight path correction as seen in figure 16. After the image-based gate recognition has been performed, the helicopter flies a curve to the new waypoints and afterwards through the gate.

The results show that autonomous flights through gates are performed successfully using the presented image processing and mission execution strategy.

CONCLUSION

This paper presents an approach to visually guide an unmanned aerial vehicle through narrow passages. Once a mission objective has been set by the UAV operator, the implemented system operates autonomously and without the need for a data link to the operator. This multi-disciplinary approach comprises a vision-based gate detection, a camera mis-
alignment compensation, a geo-referenced gate localization, as well as autonomous plan adaption and plan execution.

From real outdoor flights, the gate recognition algorithms were validated using image sequences and navigation data acquired onboard the helicopter. The overall approach is tested successfully in an autonomous flight test. As a result, the helicopter UAV is able to fly through a narrow gate without collision. The test scenarios show that without a online correction of the estimated gate position, the vehicle would have collided with the gate posts.

The onboard replanning capabilities require a low computational resources as these run at constant runtime. Moreover, the reactive planning and a state chart-based modeling of the gate detection sensor handling and the gate passing behavior decompose this complex mission task into manageable phases.

The presented approach requires a-priori knowledge about expected obstacles. Hence, upcoming research is going to extend the presented algorithm to handle a gate, window, or other small passages where fewer attributes are known a-priori. Thus, future research steps need to relax the assumption of a specific type of narrow passage, e.g., the gates presented in this work.

REFERENCES