Reactive MPC for Autonomous MAV Navigation in Indoor Cluttered Environments: Flight Experiments

Julien Marzat [*] **Sylvain Bertrand** [*] **Alexandre Eudes** [*]
Martial Sanfourche [*] **Julien Moras** [*]

[*] *ONERA - The French Aerospace Lab, F-91123 Palaiseau, France, firstname.lastname@onera.fr*

Abstract: This demonstrator paper describes a flight-tested, fully integrated perception-control loop for trajectory tracking with obstacle avoidance by micro-air vehicles (MAV) in indoor cluttered environments. For this purpose, a stereo-vision system is combined with an inertial measurement unit to estimate the vehicle localization and build a 3D model of the environment on-board. Emphasis is placed on a model predictive control (MPC) algorithm for safe guidance in unknown areas using the perception information. It combines an analytical linear quadratic solution for trajectory tracking and an efficient discretization strategy for collision avoidance. Experimental results in a flying arena and at an industrial site provide an overview of the demonstrator capabilities.

Keywords: Flight experiments, Micro-air vehicles, Model Predictive Control, Vision-based localization and mapping.

1. INTRODUCTION

Autonomous micro-air vehicles (MAV) can be of major interest for industrial applications that require the coverage of large areas or accessing high locations, e.g. for inspection or surveillance missions. However, the deployment of such platforms in real-world indoor cluttered environments with no prior information on the environment remains a challenging issue. In the absence of an absolute localization sensor (such as GPS), the vehicle should be able to achieve simultaneous localization and mapping using only its embedded sensors. Monocular or stereo-vision cameras are the most popular choices in this context, since these devices are compatible with MAV payloads and of limited cost. Several vision-based localization and mapping algorithms have then been proposed in the literature (Scaramuzza and Fraundorfer (2011)). The incorporation of such a localization system in an integrated perception and control loop that can run on-board of a MAV requires a huge effort in algorithmic optimization and many hours of testing. Only a few fully-operational MAV demonstrators with autonomous navigation embedded capabilities relying on vision measurements have been recently reported (see in particular Heng et al. (2014); Burri et al. (2015); Faessler et al. (2016)). In line with this research, we propose a demonstrator of autonomous MAV navigation in cluttered environments relying on stereo-vision and IMU measurements. It is built upon our previous work on vision-based localization and mapping (Sanfourche et al. (2014)), embedded perception and control loops (Roggeman et al. (2014)), and model predictive control (MPC) with reduced computational cost (Rochefort et al. (2014)).

The main contribution described in this paper is a new MPC strategy for trajectory tracking with obstacle avoidance in an unknown environment. It is integrated in a complete perception and control loop embedded on a multi-rotor MAV, which has been validated via extensive flight experiments in both controlled facilities and real-world environment. Many MPC algorithms have been designed for trajectory tracking, the reader is referred to the recent work of Mellinger et al. (2012); Hofer et al. (2016); Neunert et al. (2016). Most of these approaches are based on non-linear models and numerical optimization methods, which entail a high computational cost that can be detrimental when the controller should run alongside other algorithms on an embedded processor with limited computational power. Two classical strategies can be adopted for obstacle avoidance, either the planning of an avoidance trajectory which should then be tracked by another controller (as in Park et al. (2009); Mellinger et al. (2012)) or reactive strategies which make a direct use of the current estimated states of the vehicle and the environment (Lee et al. (2011); Belkhouche and Bendjilali (2012)). Our MPC algorithm is a reactive strategy which combines a linear analytical solution for trajectory tracking with a systematic search procedure for obstacle avoidance using the vision-based environment model built online. The overall algorithm has a limited computational cost, which makes it suitable for the considered application.

This paper is organized as follows. Section 2 provides an overview of the embedded algorithmic chain. Section 3 recalls the vision algorithms for state estimation and mapping. The new MPC algorithm is described in detail in Section 4. Experimental results of the demonstrator are presented in Section 5, and the associated demonstration possibilities are finally summarized in Section 6.

2. PERCEPTION AND CONTROL ARCHITECTURE

The target vehicle for flight experiments is an Asctec Pelican quadrotor controlled in diamond configuration (Figure 1). It includes an integrated IMU and a proprietary low-level controller, which makes it possible to send directly acceleration control inputs to the MAV (through a conversion to thrust, roll angle, pitch angle and yaw rate inputs). It also embeds a quad-core i7 CPU on which the complete perception and control algorithmic chain from Figure 2 is processed, which makes the MAV fully autonomous and robust to the loss of communication with the ground station.

The main sensor for localization and mapping is a stereo rig pointing in the x-axis direction, composed of two USB cameras including 4mm S-mount lenses, separated by a baseline of 22cm and synchronized electronically. The other sensors are the Asctec IMU (which provides a reliable pre-filtered attitude estimation) and a laser telemeter pointing downward for height measurement. These sensors are used to obtain a state estimate and information on the environment that are exploited by a MPC-based guidance module to follow reference trajectories or reach waypoints while avoiding unknown obstacles. This module provides acceleration control inputs directly compatible with the Asctec low-level controller.

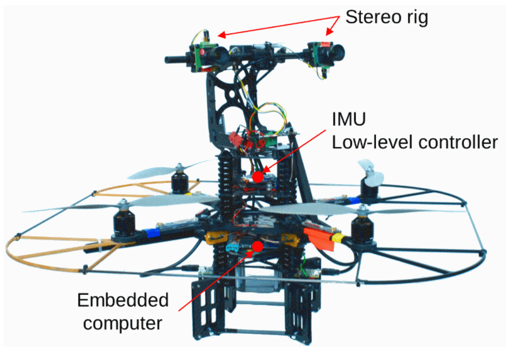


Fig. 1. Asctec MAV with stereo-vision sensors and embedded processing capabilities

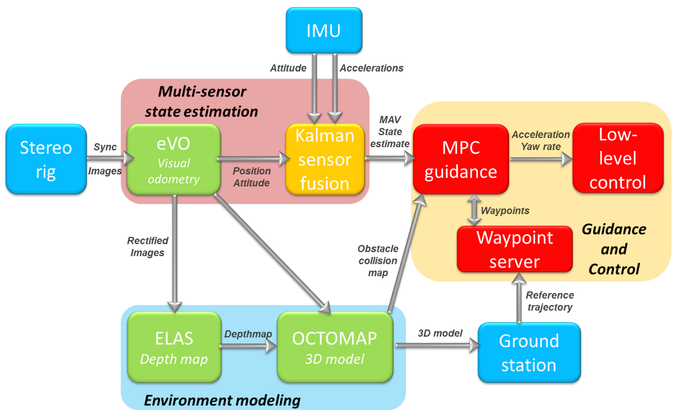


Fig. 2. Perception and control loop for autonomous navigation with obstacle avoidance

3. VISION-BASED LOCALIZATION AND MAPPING

3.1 Visual odometry

Localization is achieved by our embedded visual odometry algorithm eVO (Sanfourche et al. (2013)), whose input is the image pair captured by the stereo-rig. The operating principle is to build a map of isolated landmarks, which is updated in a key-frame scheme as proposed in Klein and Murray (2007). To be compatible with real-time processing, this update mechanism presents the following simplifying characteristics.
(1) Landmarks are pruned when they fall outside the sensor field of view (as in Mouragnon et al. (2006)), which allows to manage only a local map.
(2) The initial localization estimate of the landmarks in the inertial frame is kept during all the key-frame lifetime. With more computational power available, this position could be adjusted by minimizing a multi-view reprojection criterion.
The structure of eVO combines two macro-functions: localization and map management (Figure 3).
The **localization function** estimates position and attitude by tracking previously mapped landmarks in the left image. This feature tracking is carried out by an efficient implementation of the KLT algorithm of Shi and Tomasi (1994). The pose is robustly inferred from the subsequent 2D-3D matching in a two-stage process. First, a RANSAC step on the Perspective-3-Points algorithm provides the most probable estimate with respect to 2D-3D matching and a set of known inliers. Then, a nonlinear least-square optimization step (Lourakis and Argyros (2009)) is employed to minimize the reprojection error over the inlier matches.
The **mapping function** manages the 3D landmarks in a local map. The inclusion of new landmarks is activated whenever the ratio between the successfully tracked features and the number of 3D landmarks visible on the last key-frame falls below a threshold. Then, new Harris points are detected in the left image and are matched within the right image, using a multi-scale exhaustive search along the epipolar lines. A triangulation operation provides the localization estimate in the sensor frame (linked to the IMU), which is finally transformed into the inertial frame. Using the inlier/outlier classification of the RANSAC step from the first thread, a landmark is discarded when it is repeatedly classified as an outlier at several successive time steps.

3.2 State estimation

The eVO estimated position is combined with the accelerometer measurements and the built-in attitude estimate from the Asctec IMU in a Kalman filter scheme, as in Lynen et al. (2013). A test on the filter innovation amplitude (as in Roggeman et al. (2014)) is performed to detect inconsistencies between the visual odometry and the IMU. In case of fault detection, a sequence of reconfiguration actions is carried out: reset of eVO, switch to relative height regulation on the z-axis, and in last resort control is given back to the human pilot. An independent Kalman filter estimates the ground height and vertical velocity from laser telemeter measurements and IMU attitude.

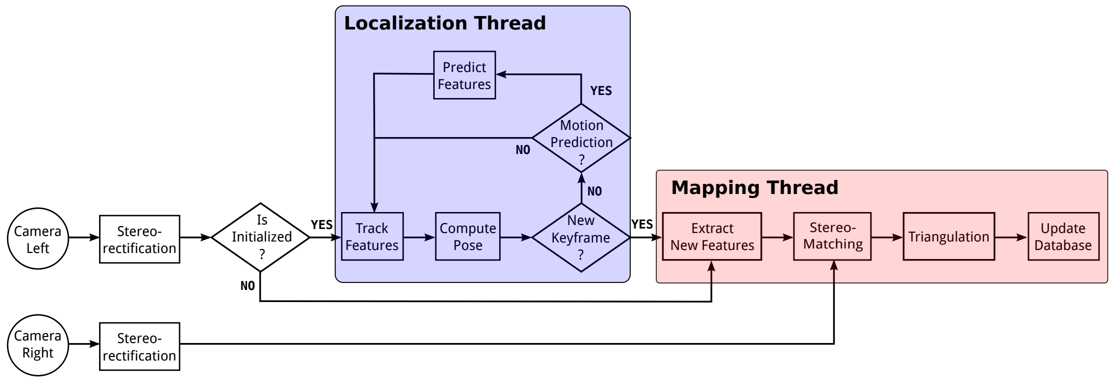


Fig. 3. Architecture of eVO algorithm (Sanfourche et al. (2013))

3.3 3D environment modeling

To be able to navigate in a cluttered environment, a 3D environment model is built and updated on-line by incorporating instantaneous sensor-to-scene distance measurements, which are transformed into the global frame thanks to the current vehicle state estimate. The rectified pair of stereo images is used to compute a depth map with the ELAS (Efficient Large-scale Stereo Matching) algorithm by Geiger et al. (2010). This algorithm achieves a robust matching of image features, followed by the computation of a dense disparity through the probabilistic propagation of the local depth information estimated on the matched features. This depth information is then included in a 3D occupancy grid, which divides the space in a lattice of cells (Elfes (1989)). The widely-used open-source Octomap model (Hornung et al. (2013)) is exploited. It is based on an octree data structure whose elements (voxels) include the probabilities of occupancy and free-space. These quantities are updated by a ray-tracing method to incorporate the depth information. The occupancy probability of a voxel is increased whenever a new 3D point is located inside, while at the same time all voxels on the ray between the sensor location and this end-point witness an increase in their probability of being free. Octree is very competitive in terms of memory occupation thanks to a hierarchical structure that is able to adapt the map resolution to the needed level of detail. Examples of maps built on-board of the MAV can be found in Figures 5, 7, 8.

Since testing all voxels for collision around a potential MAV position consumes a lot of CPU resources, an Euclidean Distance Transform (EDT) is applied to obtain a 3D distance map to the closest obstacle (Lau et al. (2013)). At each update of the Octomap, this map is built on the same structure via a search of the closest occupied voxel at each position (with predefined probability thresholds and within a parameterized maximum distance). The grid update strategy is quite computationally demanding and cannot be achieved at high frequency. In practice, the map update frequency is reduced to 2 Hz at VGA resolution and two EDT structures are managed in parallel to ensure concurrent access to the distance map even during the EDT update. To test for collision with obstacles, the guidance algorithm sends a query on a position ξ to be evaluated, for which the EDT map returns the distance $d_{obs}(\xi)$ and direction unit vector $n_{obs}(\xi)$ to the nearest obstacle.

4. MPC GUIDANCE WITH OBSTACLE AVOIDANCE

4.1 MAV model

For control design purpose, the translational dynamics of the MAV are described by

$$\begin{aligned} \dot{\xi} &= V \\ \dot{V} &= -\frac{\mathcal{T}}{m} R e_3 + g e_3 \end{aligned} \tag{1}$$

where $\xi \in \mathbb{R}^3$ and $V \in \mathbb{R}^3$ respectively denote the position and velocity of the MAV in an inertial frame $\mathcal{I} = (O; e_1, e_2, e_3)$, $\mathcal{T} \in \mathbb{R}^+$ is the module of the resulting aerodynamic forces assumed to be mainly driven by the thrust generated by the rotors, m is the mass of the vehicle, g is the gravity constant and R is the rotation matrix defining the orientation of the frame $\mathcal{B} = (G; e_1^b, e_2^b, e_3^b)$ attached to the MAV with respect to $\mathcal{I}$. By classically defining the control input $u = -\frac{\mathcal{T}}{m} R e_3 + g e_3$ and assuming that the inner attitude control loop ensures a faster convergence of the orientation dynamics, the system (1) can be considered as a double integrator for the guidance algorithm. The discrete-time state-space representation $x_{k+1} = Ax_k + Bu_k$ is implemented, with

$$x_k = \begin{bmatrix} \xi_k \\ V_k \end{bmatrix}, \quad A = \begin{bmatrix} I_3 & T_e I_3 \\ 0_3 & I_3 \end{bmatrix}, \quad B = \begin{bmatrix} \frac{T_e^2}{2} I_3 \\ T_e I_3 \end{bmatrix} \tag{2}$$

where 0_3 and I_3 respectively denote the zero and identity matrices of $\mathbb{R}^{3\times 3}$, and where T_e is the sampling period.

4.2 MPC guidance algorithm

As previously mentioned, the considered control problem consists in making the MAV track a given reference trajectory defined in terms of positions ξ_k^r and velocities V_k^r. This trajectory can be generated by two means. If only a waypoint is defined, the sequence is built using a constant nominal velocity v_{nom} and the MPC time step T_e with additional velocity ramps at the beginning and at the end. If a pre-defined trajectory is specified, an interpolation is performed to fit the discretization with the MPC time step. If an obstacle leading to a collision risk is detected during the flight, the designed control law must ensure that the MAV will deviate enough without generating a new reference trajectory (reactive avoidance). Therefore the control input u_k applied to the MAV is composed of two components

$$u_k = u_k^n + \mathbf{1}_k^{obs} u_k^a \tag{3}$$

where $\mathbf{1}_k^{obs}$ is conditional to a collision risk with an obstacle (see Section 4.2.2), or is equal to 0 otherwise. The first component u_k^n will enable the tracking of the reference trajectory in a nominal case, i.e. without obstacles. The second one, u_k^a, will make the MAV deviate from the reference trajectory just enough to ensure the absence of collision with obstacles.

Algorithm 1 Summary of the reactive MPC algorithm

```
1:  OFFLINE
2:  define the set $\mathcal{S}^{c^r}$ of reference candidate vectors
3:  define avoidance planes $\mathcal{P}^c$
4:  ONLINE
5:  loop   (time step $k$)
6:      Compute nominal control $u_k^n$ using (9)
        and predicted MAV positions $\{\bar{\xi}^*_{k+i}\}_{i=0..N}$
7:      Test collision risk along $\{\bar{\xi}^*_{k+i}\}_{i=0..N}$ using (10)
8:      if (collision risk detected) then
9:          for each avoidance plane $\mathcal{P}^c$
10:             (start with feasible plane at $k-1$)
11:         do
12:             Look for a solution to problem (11)
                by systematic search over $\mathcal{S}^c$
13:             if a solution has been found then
14:                 Set $u_k = u_k^n + u_k^a$ and exit loop
15:             else if no solution but $\exists$ other avoidance
16:                         planes not tested yet then
17:                 Switch to next avoidance plane
18:             else if no solution and no other avoidance
19:                         planes left to be tested then
20:                 Exit loop and switch to emergency
                    hover mode for the MAV
21:     else (no collision risk detected)
22:         Set $u_k = u_k^n$
23:     Apply $u_k$ as control input to the MAV
```

Nominal MPC

To classically deal with control-offset while reference tracking, let us define the control increment $\delta_k = u_k - u_{k-1}$ and the augmented state $z_k = \begin{bmatrix} x_k^T & u_{k-1}^T \end{bmatrix}^T$.

For a given prediction horizon of size N we also introduce the vector of control increments

$$\Delta_k = \begin{bmatrix} \delta_k^T & \delta_{k+1}^T & \dots & \delta_{k+N-1}^T \end{bmatrix}^T \tag{4}$$

and the reference vector

$$X_k^r = \begin{bmatrix} (x_k^r)^T & (x_{k+1}^r)^T & \dots & (x_{k+N}^r)^T \end{bmatrix}^T \tag{5}$$

where $x_k^r = \begin{bmatrix} (\xi_k^r)^T & (V_k^r)^T \end{bmatrix}^T$.

The nominal control component u_k^n is designed by considering the following MPC problem:

$$\Delta_k^* = \arg\min_{\Delta_k} J_k \tag{6}$$

such that

$$J_k = \sum_{i=0}^{N-1} \left\{ \left\| \bar{x}_{k+i} - x_{k+i}^r \right\|_Q^2 + \left\| \delta_{k+i} \right\|_R^2 \right\} + \left\| \bar{x}_{k+N} - x_{k+N}^r \right\|_P^2 \tag{7}$$

and

$$\begin{aligned} \bar{x}_{k+i+1} &= A\bar{x}_{k+i} + Bu_{k+i} \\ u_{k+i} &= u_{k+i-1} + \delta_{k+i} \qquad i = 0,..,(N-1) \\ \bar{x}_k &= x_k \end{aligned} \tag{8}$$

Assuming $P = Q$, the following analytical control law is classically obtained:

$$\Delta_k^* = -\left(\mathcal{B}^T \mathcal{Q} \mathcal{B} + \mathcal{R}\right)^{-1} (\mathcal{Q}\mathcal{B})^T \left[\mathcal{A} z_k - X_k^r\right] \tag{9}$$

where matrices $\mathcal{A}$ and $\mathcal{B}$ depend on A and B and are used to explicitly compute the predictions $\bar{x}_{k+i}$ ($i = 0,..,N$), and where $\mathcal{Q}$ and $\mathcal{R}$ are block-diagonal matrices with respectively Q and R as diagonal elements. From the first component of the solution (9), the control input $u_k^n = u_{k-1}^n + \delta_k^*$ is deduced. It hence consists in a state feedback controller and the gain matrices Q and R can be chosen so as to ensure the closed loop stability of the tracking error dynamics.

Collision test and avoidance MPC

A collision test is performed for each position $\{\bar{\xi}^*_{k+i}\}_{i=0..N}$ of the predicted trajectory $\{\bar{x}^*_{k+i}\}_{i=0..N}$, computed using the solution Δ_k^*. For a given predicted position $\bar{\xi}^*_{k+i}$, the EDT map provides the distance $d_{k+i}^{obs} = d_{obs}\left(\bar{\xi}^*_{k+i}\right)$ and direction unit vector $n_{k+i}^{obs} = n_{obs}\left(\bar{\xi}^*_{k+i}\right)$ to the closest obstacle. A collision risk is considered if the following condition holds for a least one point of the predicted trajectory:

$$(n_{k+i}^{obs})^T . \begin{bmatrix} (d_a^x)^2 & 0 & 0 \\ 0 & (d_a^y)^2 & 0 \\ 0 & 0 & (d_a^z)^2 \end{bmatrix}^{-1} . n_{k+i}^{obs} \le \frac{1}{\left(d_{k+i}^{obs}\right)^2} \tag{10}$$

where d_a^x, d_a^y and d_a^z define *activation* distances with respect to each direction. In absence of collision risk along the predicted trajectory, $\mathbf{1}_k^{obs}$ is set to zero in (3) and $u_k = u_k^n$ is applied. In case of collision risk, $\mathbf{1}_k^{obs}$ is set to one and the second control component of (3) is computed as follows.

The main objective of the second control component u_k^a is to deviate the MAV trajectory. Defining, for a given control horizon $0 < N_c \le N$, the vector of *avoidance* control actions $U_k^a = \left[(u_k^a)^T \, (u_{k+1}^a)^T \dots (u_{k+N_c-1}^a)^T\right]^T$, u_k^a is computed as the first component of the solution of the optimization problem

$$\begin{aligned} U_k^{a^*} &= \arg\min_{U_k^a} J_k^a \\ \text{s.t. } & u_{k+i} \in \mathbb{U} \qquad i = 0,..,(N-1) \\ & u_{k+i} = \begin{cases} u_{k+i}^n + u_{k+i}^a & \text{if } i \le N_c - 1 \\ u_{k+i}^n & \text{otherwise} \end{cases} \end{aligned} \tag{11}$$

where $\mathbb{U}$ defines the set of constraints on the control input applied to the vehicle.

The *avoidance* cost J_k^a is defined so as to achieve a good trade-off between obstacle avoidance, reference trajectory tracking and control energy by

$$J_k^a = w^{obs}.J_k^{obs} + w^{nav}.J_k^{nav} \tag{12}$$

where w^{obs} and w^{nav} are weighting positive factors.

The first cost component, J_k^{obs} is introduced to penalize proximity to the obstacle. It is defined as

$$J_k^{obs} = \sum_{i=0}^{N} f_{obs}\left(d_{k+i}^{obs}\right) \tag{13}$$

where f_{obs} is a smooth function mapping the distance to the nearest obstacle into a cost between 0 and 1 (see Rochefort et al. (2014) for an example and weight tuning guidelines). The second cost component, J_k^{nav} penalizes deviations from the reference trajectory to be tracked and magnitude of avoidance control actions:

$$
\begin{aligned}
J_k^{nav} &= \sum_{i=0}^{N} \left\|\bar{x}_{k+i} - x_{k+i}^r\right\|^2 + \sum_{i=0}^{N_c-1} \left\|u_{k+i}^a\right\|^2 \\
\bar{x}_{k+i+1} &= A\bar{x}_{k+i} + Bu_{k+i} \qquad\qquad i = 0,..,(N-1) \\
\bar{x}_k &= x_k \\
u_{k+i} &= \begin{cases} u_{k+i}^n + u_{k+i}^a & \text{if } i \leq N_c - 1 \\ u_{k+i}^n & \text{otherwise} \end{cases}
\end{aligned} \tag{14}
$$

Finding online a solution to the constrained optimization problem (11) may be computationally expensive. To reduce computation burden, a simple systematic search procedure similar to Frew (2005) and Rochefort et al. (2014) has been adopted. For simplicity of implementation, the avoidance control actions are assumed to be of constant value over the control horizon, i.e. $u_{k+i}^a = u^c\,(i = 0, \ldots, N_c - 1)$. The systematic search procedure adopted to solve the optimization problem (11) consists in finding the candidate vector u^{c^*}, over a predefined finite set $\mathcal{S}^c = \{u_s^c\}_{s=1,..,N_S}$ of N_S candidate vectors, that minimizes the cost function while satisfying the control constraints. Assessing one by one the candidates in $\mathcal{S}^c$ allows to guarantee a constant computation time at each iteration, which can be tuned directly through the value assigned to N_S.
Since generating $\{u_s^c\}_{s=1,..,N_S}$ by spanning the entire 3D control space may result in a very high number of candidate vectors in $\mathcal{S}^c$, it has been chosen to generate a set of reference candidate vectors leading to trajectory deviations in a predefined reference avoidance plane (as in Belkhouche and Bendjilali (2012)). Several avoidance planes and corresponding candidate vectors are then deduced from these references, to allow 3D obstacle avoidance for the vehicle.

Let us denote by $\mathcal{P}^r$ the reference avoidance plane defined by $(e_1, -e_2)$ and its normal unit vector $-e_3$. The corresponding set $\mathcal{S}^{c^r} = \left\{u_s^{c^r}\right\}_{s=1,..,N_S}$ of reference candidate vectors are generated in this reference frame $(e_1, -e_2, -e_3)$, under the constraint $(u_s^{c^r})^T(-e_3) = 0$.
Let us define the unit vector

$$
n = \frac{\xi_{k+1}^r - \xi_k^r}{\left\|\xi_{k+1}^r - \xi_k^r\right\|} \tag{15}
$$

along the local portion of the reference trajectory to be tracked. Avoidance planes are then defined as planes containing n and obtained by a rotation of $\mathcal{P}^r$ mapping e_1 into n. For a given avoidance plane $\mathcal{P}^c$, and corresponding rotation matrix R^c such that $n = R^c e_1$, the set $\mathcal{S}^c$ of candidate vectors can be deduced from $\mathcal{S}^{c^r}$ by $u_s^c = R^c u_s^{c^r}, s = 1,..,N_S$.
Two examples of avoidance planes are the local *"horizontal"* plane $\mathcal{P}_H^c$ defined by (n, n_H) and the *"vertical"* plane $\mathcal{P}_V^c$ defined by (n, n_V), with

$$
n_H = \frac{n \times e_3}{\|n \times e_3\|} \qquad n_V = \frac{n \times n_H}{\|n \times n_H\|} \tag{16}
$$

Rotation matrices associated to these planes are given by

$$
R_H^c = [n \,|\, n_H \,|\, n_V] \qquad\qquad R_V^c = [n \,|\, n_V \,|\, -n_H] \tag{17}
$$

The definition of the avoidance planes can be achieved offline, as well as the computation of the corresponding candidate vectors. During the flight, if an obstacle is detected, the systematic search procedure is run by considering successively all the possible sets of candidate vectors, i.e. all the possible avoidance planes. The search is stopped and considered as successful as soon as an avoidance plane leads to a feasible solution to problem (11). At the next time instant, if an obstacle is still detected, the avoidance plane that has provided a feasible solution at the previous time step is considered first for the systematic search. This improves the smoothness of the avoidance trajectory achieved by the vehicle. If no feasible solution can be found, the search is considered as unsuccessful and the vehicle enters in an emergency mode while stopping trajectory tracking and starting hovering.

5. FLIGHT EXPERIMENTS

Several flight experiments have been performed to validate the proposed perception and control architecture. Validation of vision-based perception and trajectory tracking by the nominal component of the MPC has been performed first in a flying arena, as presented in Section 5.1. Real-world scenarios have then been experimented in an industrial warehouse, where tracking of longer trajectories and obstacle detection and avoidance capabilities have been validated (see Section 5.2).

5.1 Vision-based perception and trajectory tracking

The results presented here were obtained in a flying arena equipped with a motion capture system as ground truth reference. Two reference trajectories are analyzed: a circle of circumference 20 m in a plane at low speed (0.3 m/s) and a more complex 3D trajectory of 30 m length with sharp turns at higher speed (0.65 m/s). These trajectories were generated using the polynomial tools described in Oleynikova et al. (2016). The localization results are evaluated by computing the RMS between the position estimated on-line by our vision-based algorithm and the motion capture system. The tracking accuracy is evaluated by comparing the reference trajectory with the actual one. Localization and tracking results are plotted in Figures 4 and 6, while the corresponding RMS values are reported in Table 1. An evaluation of the on-board 3D model conducted in the flying arena (Figure 5) provided a very high Matthews Correlation Coefficient (Matthews (1975)) of 0.94 with respect to a ground truth model obtained by a combination of lidar scanning and motion capture positioning.

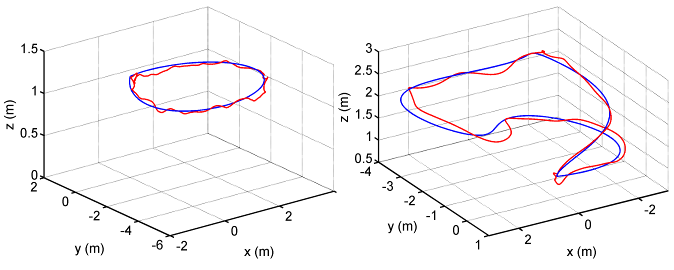
5.2 Obstacle avoidance in industrial environment

In the context of the research partnership between ONERA and SNCF (French Railways), real-world experiments have been conducted in a large warehouse (Figure 7) to demonstrate autonomous inspection capabilities. The full perception and control loop was running on-board of the MAV and achieved safe trajectory tracking and environment modeling using only the embedded vision measurements. One scenario included following a reference trajectory crossing the path of a pillar, as the environment model was not known in advance. As shown in Figure 8, this obstacle was detected and avoided using the proposed MPC algorithm, with a desired separation distance of 2 m.

6. CONCLUSIONS

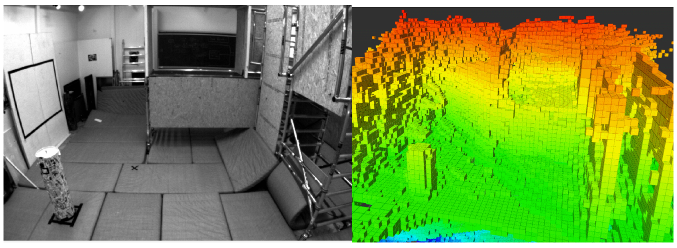
A complete loop including vision-based localization and environment mapping as well as MPC trajectory tracking with obstacle avoidance for autonomous MAV navigation in GPS-denied cluttered environments has been described in this paper. Emphasis has been particularly placed on the control algorithm and on the experimental flight validation of the embedded algorithms in a flying arena with ground truth reference but also at an industrial facility. Building on these elements, this flight-tested MAV demonstrator is able to perform the following tasks:

- Autonomous take-off and landing.
- Fully embedded vision-based localization and real-time 3D environment modeling, with associated ground station visualization.
- Trajectory tracking with autonomous avoidance of obstacles, within a desired safety distance.



(a) Circular trajectory (b) 3D trajectory

Fig. 4. Trajectory tracking in flying arena during trajectory tracking (blue: reference, red: MAV trajectory)



(a) Scene view (b) Octomap built on-board

Fig. 5. 3D model construction in flying arena

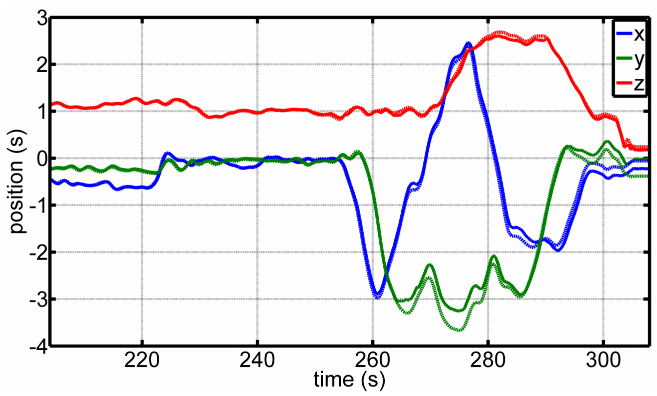


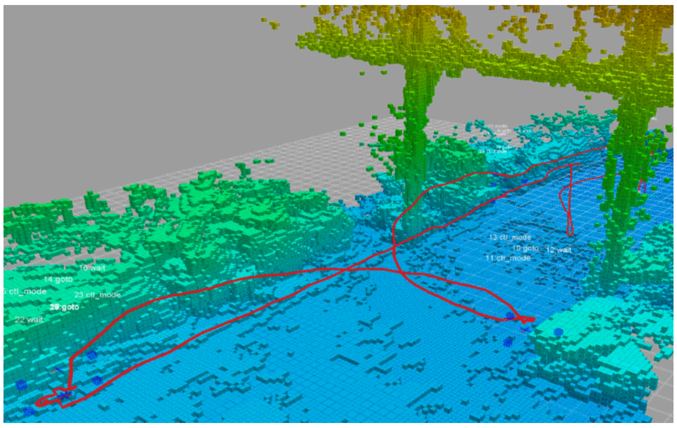
Fig. 6. Evaluation of vision-based localization in flying arena (full: eVO, dashed: motion capture)

Table 1.

Localization and tracking accuracy in flying arena

	RMS of localization error (m)	RMS of tracking error (m)
Circular trajectory	0.067	0.131
3D trajectory	0.077	0.219

(a) SNCF industrial site for flight experiments



(b) 3D model autonomously built on-board by the MAV while following a reference trajectory (in red) composed of portions of ellipses (onward) and straight line (return)

Fig. 7. Flight experiments at SNCF warehouse, Sotteville-lès-Rouen, France

REFERENCES

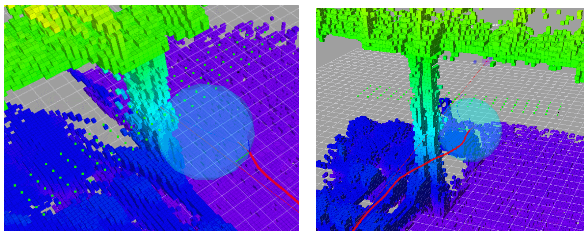
Belkhouche, F. and Bendjilali, B. (2012). Reactive path planning for 3-D autonomous vehicles. *IEEE Transactions on Control Systems Technology*, 20(1), 249–256.

Burri, M., Oleynikova, H., Achtelik, M.W., and Siegwart, R. (2015). Real-time visual-inertial mapping, re-localization and planning onboard MAVs in unknown environments. In *IEEE/RSJ International Conference on Intelligent Robots and Systems, Hamburg, Germany*, 1872–1878.

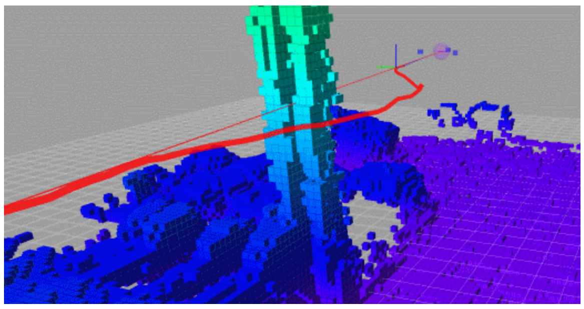
Elfes, A. (1989). Using occupancy grids for mobile robot perception and navigation. *Computer*, 22(6), 46–57.

Faessler, M., Fontana, F., Forster, C., Mueggler, E., Pizzoli, M., and Scaramuzza, D. (2016). Autonomous, vision-based flight and live dense 3D mapping with a quadrotor micro aerial vehicle. *Journal of Field Robotics*, 33(4), 431–450.

Frew, E.W. (2005). Receding horizon control using random search for UAV navigation with passive, non-cooperative



(a) Collision detection and control input computation



(b) Final avoidance trajectory

Fig. 8. MPC trajectory tracking and obstacle avoidance

sensing. In *Proceedings of the AIAA Guidance, Navigation and Control Conference, San Francisco, CA, USA.*

Geiger, A., Roser, M., and Urtasun, R. (2010). Efficient large-scale stereo matching. In *Proceedings of the 10th Asian Conference on Computer Vision (ACCV), Queenstown, New Zealand*, 25–38.

Heng, L., Honegger, D., Lee, G.H., Meier, L., Tanskanen, P., Fraundorfer, F., and Pollefeys, M. (2014). Autonomous visual mapping and exploration with a micro aerial vehicle. *Journal of Field Robotics*, 31(4), 654–675.

Hofer, M., Muehlebach, M., and D'Andrea, R. (2016). Application of an approximate model predictive control scheme on an unmanned aerial vehicle. In *IEEE International Conference on Robotics and Automation, Stockholm, Sweden*, 2952–2957.

Hornung, A., Wurm, K.M., Bennewitz, M., Stachniss, C., and Burgard, W. (2013). OctoMap: An efficient probabilistic 3D mapping framework based on octrees. *Autonomous Robots*, 34(3), 189–206.

Klein, G. and Murray, D. (2007). Parallel tracking and mapping for small AR workspaces. In *Proceedings of the 6th IEEE and ACM International Symposium on Mixed and Augmented Reality, Nara, Japan*, 225–234.

Lau, B., Sprunk, C., and Burgard, W. (2013). Efficient grid-based spatial representations for robot navigation in dynamic environments. *Robotics and Autonomous Systems*, 61(10), 1116–1130.

Lee, D., Lim, H., and Kim, H.J. (2011). Obstacle avoidance using image-based visual servoing integrated with nonlinear model predictive control. In *50th IEEE Conference on Decision and Control and European Control Conference, Orlando FL, USA*, 5689–5694.

Lourakis, M.A. and Argyros, A. (2009). SBA: A Software Package for Generic Sparse Bundle Adjustment. *ACM Transactions on Mathematical Software*, 36(1), 1–30.

Lynen, S., Achtelik, M.W., Weiss, S., Chli, M., and Siegwart, R. (2013). A robust and modular multi-sensor fusion approach applied to MAV navigation. In *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems, Tokyo, Japan*, 3923–3929.

Matthews, B.W. (1975). Comparison of the predicted and observed secondary structure of T4 phage lysozyme. *Biochimica et Biophysica Acta (BBA) - Protein Structure*, 405(2), 442–451.

Mellinger, D., Michael, N., and Kumar, V. (2012). Trajectory generation and control for precise aggressive maneuvers with quadrotors. *The International Journal of Robotics Research*, 5, 664–674.

Mouragnon, E., Lhuillier, M., Dhome, M., Dekeyser, F., and Sayd, P. (2006). Real time localization and 3d reconstruction. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR), New York, NY, USA*, volume 1, 363–370.

Neunert, M., de Crousaz, C., Furrer, F., Kamel, M., Farshidian, F., Siegwart, R., and Buchli, J. (2016). Fast nonlinear model predictive control for unified trajectory optimization and tracking. In *IEEE International Conference on Robotics and Automation, Stockholm, Sweden*, 1398–1404.

Oleynikova, H., Burri, M., Taylor, Z., Nieto, J., R., S., and Galceran, E. (2016). Continuous-time trajectory optimization for online UAV replanning. In *IEEE/RSJ International Conference on Intelligent Robots and Systems, Daejeon, Korea.*

Park, J.M., Kim, D.W., Yoon, Y.S., Kim, H.J., and Yi, K.S. (2009). Obstacle avoidance of autonomous vehicles based on model predictive control. *Proceedings of the Institution of Mechanical Engineers, Part D: Journal of Automobile Engineering*, 223(12), 1499–1516.

Rochefort, Y., Piet-Lahanier, H., Bertrand, S., Beauvois, D., and Dumur, D. (2014). Model predictive control of cooperative vehicles using systematic search approach. *Control Engineering Practice*, 32, 204–217.

Roggeman, H., Marzat, J., Sanfourche, M., and Plyer, A. (2014). Embedded vision-based localization and model predictive control for autonomous exploration. In *IROS Workshop on Visual Control of Mobile Robots (ViCoMoR), Chicago IL, USA*, 13–20.

Sanfourche, M., Plyer, A., Bernard-Brunel, A., and Le Besnerais, G. (2014). 3DSCAN: Online ego-localization and environment mapping for micro aerial vehicles. *AerospaceLab Journal*, 8, 1–17.

Sanfourche, M., Vittori, V., and Le Besnerais, G. (2013). eVO: A realtime embedded stereo odometry for MAV applications. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Tokyo, Japan*, 2107–2114.

Scaramuzza, D. and Fraundorfer, F. (2011). Visual odometry: Part I - the first 30 years and fundamentals. *IEEE Robotics and Automation Magazine*, 18(4), 80–92.

Shi, J. and Tomasi, C. (1994). Good features to track. In *1994 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Seattle, WA, USA*, 593 – 600.

ACKNOWLEDGEMENTS

The authors thank SNCF for their support to this work, partially funded by the research partnership between ONERA and SNCF.

Some of the experiments were conducted within the European Community's Seventh Framework Programme (FP7) under grant agreement n.608849 (EuRoC).