Detection, Estimation and Avoidance of Mobile Objects Using Stereo-Vision and Model Predictive Control

Hélène Roggeman, Julien Marzat, Maxime Derome,
Martial Sanfourche, Alexandre Eudes, Guy Le Besnerais
ONERA – The French Aerospace Lab, F-91123 Palaiseau, France
{firstname.lastname}@onera.fr

Abstract

We propose a complete loop (detection, estimation, avoidance) for the safe navigation of an autonomous vehicle in presence of dynamical obstacles. For detecting moving objects from stereo images and estimating their positions, two algorithms are proposed. The first one is dense and has a high computational load but is designed to fully exploit GPU processing. The second one is lighter and can run on a standard embedded processor. After a step of filtering, the estimated mobile objects are exploited in a model predictive control scheme for collision avoidance while tracking a reference trajectory. Experimental results with the complete loop are reported for a micro-air vehicle and a mobile robot in realistic situations, with everything computed on board.

1. Introduction

The safe navigation of terrestrial and aerial autonomous robots requires the development of new methods for event detection and trajectory definition in uncertain environments. For vehicles with a limited payload and in situations where usual localization means (GPS/IMU) are unavailable or not accurate enough, a current trend in robotics is to rely on embedded vision data for simultaneous localization, mapping and obstacle detection [6].

This paper considers the definition of a generic strategy relying mainly on embedded stereo-vision for detecting mobile objects with no specific prior information, estimating their motion and defining safe trajectory for autonomous navigation. The target applications are micro-air vehicles (MAV) in indoor cluttered environments and mobile robots or passenger vehicles in urban environments.

This work builds upon existing results in embedded vision-based localization and mapping (eVO algorithm [26]), mobile object detection and tracking using stereo-vision [9,10] and model predictive control (MPC) for safe autonomous navigation [5,20,23,24]. The contribution here is a fully integrated perception and control loop for waypoint navigation with avoidance of unknown dynamical obstacles. Experimental results in realistic conditions are provided for a mobile robot in urban environment and a micro-aerial vehicle in flying arena. Parallel computing can be available on mobile robots, thus a more computationally expensive method has been designed to fully exploit GPU processing. On the other hand, a lighter algorithm is fully functional and compatible with the embedded processing capabilities of aerial vehicles. Specific modeling and avoidance strategies are defined within the MPC framework to take into account the mobile obstacles.

Some previous contributions have addressed similar issues. In [16], objects are detected by feature tracking and clustering and avoidance is achieved using a fuzzy controller for a mobile robot. In [14], a 2D dynamic occupancy grid is associated with a simple open-loop avoidance strategy. In [15], experimental results on terrestrial and aerial robots have been reported using a monocular detection method, which did not take into account 3D information. A camera and a range sensor were combined in [8] to track mobile obstacles and a potential field algorithm for avoidance, with experimental validation on a mobile robot. In [11], a methodology has been proposed to detect mobile obstacles using occupancy grids, predict their motion and avoid them via a motion planning strategy. This was successfully applied on an autonomous passenger vehicle equipped with several heavy sensors dedicated to navigation in urban environment. In [31], stereo-vision and a laser range sensor were used to detect mobile objects based on a prior on color and shape. An heuristic avoidance strategy was defined and tested on a mobile robot. Randomised MPC for obstacle avoidance has been reported in [7], with experimental results on terrestrial vehicles. In summary, most previous references considered combinations of sensors and computationally demanding control strategies, while we focus here on exploiting a single stereo sensor inside an integrated vision and control loop whose limited computational cost makes it compatible with most robotic platforms.

Section 2 provides an overview of the complete perception and control loop, Section 3 describes the two vision-based algorithms for detection and tracking of mobile objects. Section 4 details the filtering and motion prediction step, which is exploited by a MPC controller (Section 5) for safe autonomous navigation. Section 6 reports experimental results on two robotic platforms.

2. Overview of perception-control loop

Experiments have been carried out on two target platforms: a mobile robot and a micro-air vehicle (MAV).

The mobile robot is a four-wheel Robotnik Summit XL (Fig. 1a) equipped with a stereo rig made of two electronically synchronized USB cameras (5.5 mm S-mount lens) separated by a baseline of 18 cm. The cameras are configured to capture VGA frames at 20 Hz. The robot embeds a PC with an Intel quad-core i7 processor and a Nvidia GPU (GT640).

The MAV is an Asctec Neo hexarotor (Fig. 1b) made available in the context of the FP7 EuRoC project [28], which embeds a NUC dual-core i7 CPU, a triple-redundant IMU and the VI-sensor [21] which includes up to four cameras at 20 Hz (only the front rigid stereo-rig was used in this research).

(a) Robotnik Summit XL

(b) Asctec Firefly Neo with VI sensor

Figure 1: Robotic platforms used for experiments

The same integrated perception and control loop (Fig. 2) was deployed on the two platforms, with only a few adaptations to take into account the sensors and state-space dimensions of each robot. The main localization algorithm for the two platforms is the vision-based odometry eVO algorithm [26], which is used alone for the mobile robot and fused via Kalman filtering with the IMU for the MAV [18]. The robot low-level controller is used to make the motors track the commanded linear and angular speeds, while the low-level controller of the MAV is used to transform thrust, roll, pitch and yaw rate orders into motor speeds.

3. Mobile object detection and pose estimation

Two methods for mobile object detection and tracking using only embedded stereo-vision information have been defined. One is based on dense (i.e. pixel-wise) processing, the other relies on tracking and clustering of sparse features.

3.1. Detection from dense stereo-vision and ego-motion estimation

Mobile object detection based on dense stereo-vision algorithms has been studied in several works. They essentially rely on stereo-vision reconstruction and ego-motion estimation to predict a quantity from time k to $k+1$ under a rigid scene hypothesis. This quantity is then compared to the observation to form a residual image which is thresholded to provide candidate mobile objects. The prediction can be performed on some geometrical quantity expressed in the image space such as the Optical Flow (OF) and/or the disparity [25, 30], or in the 3D space for instance the Scene Flow (SF) as in [1, 32]. In contrast, the proposed method uses stereo-vision and ego-motion estimation to predict an image at time $k+1$ which is compared to the recorded image so as to detect moving objects. The algorithms proposed by [2, 3] were also in this line. Our contribution is to better account for uncertainty propagation in the prediction process and to propose a GPU implementation able to run in real time thanks to a careful selection of the algorithmic components.

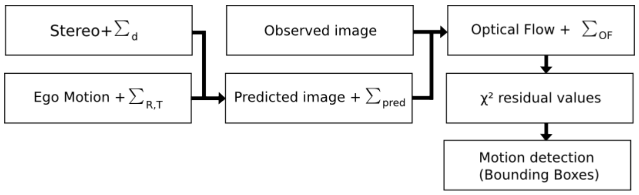

Stereo+Σ_d
Observed image
Optical Flow + Σ_{OF}
Ego Motion + $\Sigma_{R,T}$
Predicted image + Σ_{pred}
χ^2 residual values
Motion detection
(Bounding Boxes)

Figure 3: Flowchart of the dense detection method. Uncertainty sources are coded as covariance matrices Σ.

The flowchart of the algorithm based on image prediction is presented in Fig. 3. Stereo images are used for computing a depth map and also for estimating ego-motion parameters (rotation matrix R and translation vector T). All further computations are conducted on the images issued from the left camera: we denote I_k the left image at instant k. Image prediction uses the current left image combined with the estimated depth map and ego-motion parameters. The predicted image and the observed image are fed into an optical flow algorithm. The estimated optical flow is used as the residual, from which a χ^2 statistic is computed and thresholded for detection purpose. Further processing steps clean the small isolated false alarms and fit bounding boxes to the confirmed detection areas.

Rectified pairs of stereo images are considered, so that the stereo configuration is characterized by intrinsic parameters $\{f, x_0, y_0\}$ (identical for both cameras) and a baseline b along the row (X) axis. At each observation time, a dense depth map d_k in the geometry of the left image is computed

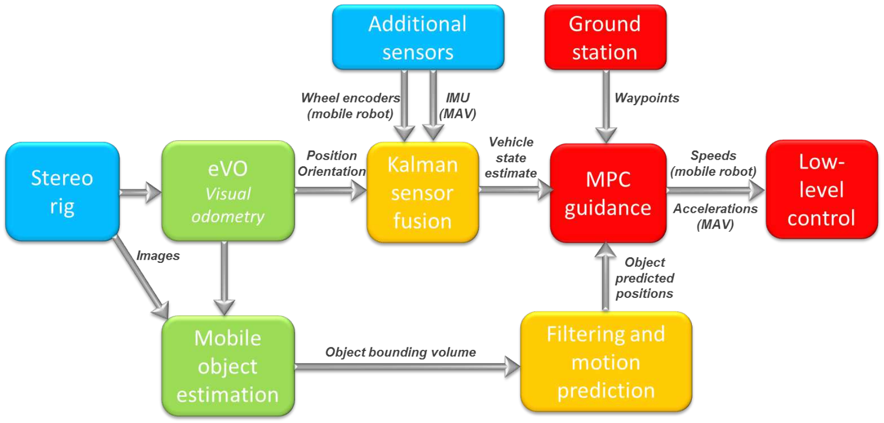


Figure 2: Perception and control loop for autonomous navigation in presence of mobile obstacles

by the fast "Accurate Coarse To Fine" (ACTF) method described in [29]. Image points of pixel coordinates (x, y) and disparity $d_k(x, y)$, are then triangulated as:

$$\hat{X}_k(x, y, d_k) = -\frac{b}{d_k(x, y)} \begin{pmatrix} x - x_0 \\ y - y_0 \\ f \end{pmatrix} \qquad (1)$$

Compensating ego-motion and going back in time, the pixel coordinates of these points in the previous frame are predicted under the assumption of a static scene, as:

$$\begin{bmatrix} x_{k-1}^{pred} \\ y_{k-1}^{pred} \end{bmatrix} = \Pi_{k-1}\left(R^{-1}\left(\hat{X}_k - T\right)\right) \qquad (2)$$

where Π refers to the projection onto the image plane at time $k-1$. The backward predicted image I_k^{pred} is obtained by interpolating image intensity I_{k-1} at these predicted positions. Occlusions are filled with image intensity picked up from frame I_k. Finally, from I_k and I_k^{pred}, we compute the residual OF $\phi_\mathrm{r}(x, y)$ for each pixel position. This pixel-wise OF estimation is the most computationally demanding step of moving object detection, for which a very fast GPU implementation of the highly parallel local OF algorithm eFolki [22] is employed. Detection is then achieved by thresholding the χ^2 map derived from the residual flow

$$\Lambda^2(x, y) = \phi_\mathrm{r}(x, y)^T \Sigma_{\phi_\mathrm{r}}^{-1} \phi_\mathrm{r}(x, y). \qquad (3)$$

The main issue here is to propagate the uncertainty of each component so as to correctly estimate the covariance matrix of the residual OF Σ_{ϕ_r}. In particular, previous references [2,3] have neglected the uncertainty about ego-motion parameters (R, T). This term is accounted for in the proposed model as detailed in [9].

From the raw detection image, several operations are conducted to filter potential false alarms and provide position and dynamic attributes for each selected detection. Connected components (CC) are extracted. For each CC, a depth is computed from the median value of the disparities. 3D neighboring CC are merged (eg. if their 3D distance is less than a threshold, e.g. 0.1 m) and median depth is recomputed. Finally, CC which are too small (e.g. areas less than 0.4 $\times$ 0.4 m^2) are rejected. The 3D position of the object is then easily derived from CC image position and median depth. Velocity is computed from the median disparity and OF over the CC. Note that here the "full OF" between I_{k-1} and I_k is required, not the residual OF. We compute the full OF from the residual one and a predicted static motion derived from depth, ego-motion and residual motion. This estimate, denoted by PCOF (Prediction-Correction Optical Flow) in [10], is obtained at a very low computational cost here, as all required components are already available from the detection step. Finally, a simple consistency check between one frame and the previous one is added to filter spurious detections: a detection in one frame at some 3D position X_d is confirmed if there is a detection in the previous frame whose position and velocity are consistent with X_d, according to the χ^2 norm of the 3D distance.

The computation time of the overall process of residual computation, detection and bounding box construction ranges between 50 ms (no obstacle) and 250 ms (several detections) on the mobile robot.

3.2. Detection from 3D feature clustering

In order to deal with limited computing power and lack of GPU, we also propose a generic sparse feature-based detection algorithm using stereo images, assuming that several image features are located on the mobile objects. Considering two successive stereo-images, mobile objects are detected in a two-step process (Fig. 4):

1. Triangulated image features are classified into static or moving particles by analysing the two-view and the

three-view geometric consistencies of the stereo and temporal features associations.

2. Moving particles are clustered into objects according to a rough geometric prior of metric dimensions.

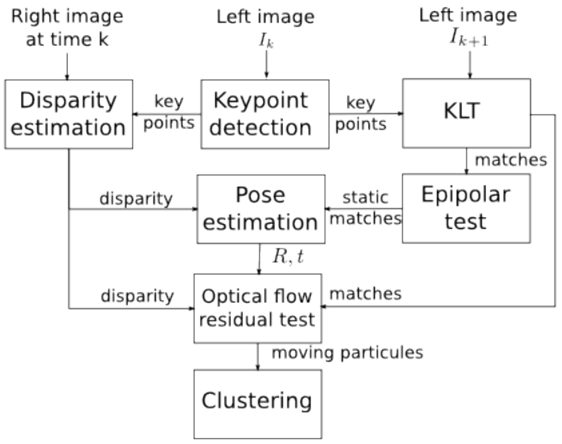


Figure 4: Summary of the sparse mobile object detection algorithm.

Harris corner are detected in the left image I_k at pixel positions (x_k^n, y_k^n). An homogeneous spatial distribution of the features is enforced thanks to a bucketing strategy. A sparse disparity calculation described in [26] makes it possible to triangulate the 3D points $\hat{X}_k^n$ associated to each feature using Eq. (1). Then, the sparse OF ϕ_k^n between time k and $k+1$ is computed on each Harris feature by the KLT algorithm [27]. A first classification into moving and static features is obtained by using the epipolar constraint on temporal associations. More precisely, a robust RANSAC-based Fundamental matrix estimation process is run on the OF vectors to classify the outliers features as moving particles. The remaining associations provide 2D-3D matches which are used to estimate the 6D camera motion (R, T) in a robust manner by the gold standard RANSAC-P3P algorithm [13], followed by a non-linear optimization on the inlier matches. Then the final detection of moving particles is achieved by thresholding the residual sparse OF $(\phi_k^n - \phi(\hat{X}_k^n, R, T))^2$, where the second term is derived from the 3D position at time $k-1$ and the ego-motion assuming that the point belongs to the static background scene. This approach is the simplest way to check the three-view geometric consistency.

A Delaunay triangulation of the moving particles then gives access to a rough 3D structure. This structure is computed in inertial frame, thanks to the knowledge of the estimated camera pose. Triangles are pruned according to their size and orientation (only the most vertical ones are retained). The remaining connected triangles form candidate objects. We finally compute 3D bounding cylinders and select those whose width is inside a specified range.

3.3. Discussion

The feature-based method can be successfully embedded on the MAV's on-board computer, since it runs in less than 100 ms. It is very efficient but ultimately relies on feature detection, and may fail if there is not enough features detected on the mobile object. Risk of non detection is lower with the first method based on dense (pixel-wise) processing, but it comes at a higher computational cost. Yet if the payload of the platform allows it, the use of a GPU makes it possible to use it in real time in a safer navigation loop.

4. Filtering and motion prediction

4.1. Modeling

The mobile object detected is modeled as a cylinder of radius r and height h, restricted to a circle in 2D. At each time step k, the output of the image processing is a measurement vector

$$Z_k = \left[X_k^T, r_k, h_k\right]^T, \tag{4}$$

where X_k is the estimated position of the center of gravity of the cylinder in inertial frame (in 2D or 3D), r_k and h_k the measured radius and height.

Since these measurements can be very noisy due to geometrical ambiguities, a step of Kalman filtering is performed. This is also a mean to provide an estimate of the object velocity, which is required for predicting its future motion. The state vector of the Kalman filter at time k is denoted by $\mathcal{X}_k = \left[X_k^T, V_k^T, r_k, h_k\right]^T$ and its associated covariance matrix is P_k. Z_k is the measurement vector from equation (4) and R_k the covariance of observation noise (provided by the perception module). Q_k is the covariance of process noise and t_e the sampling period.
As in [14], a constant velocity motion is assumed for the mobile object, which yields the dynamical model

$$\begin{cases} \mathcal{X}_k = A\mathcal{X}_{k-1} \\ Z_k = C\mathcal{X}_k \end{cases} \tag{5}$$

with

$$A = \begin{bmatrix} I & t_e.I & 0 & 0 \\ 0 & I & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \ C = \begin{bmatrix} I & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{6}$$

where I is the identity matrix of appropriate dimension (2 or 3). Since this model is linear, a classical Kalman filter can be applied. The process noise covariance matrix, for estimating the velocity and shape parameters, has the form $Q_k = \text{diag}\left(0, \sigma_v^2.I, \sigma_s^2, \sigma_s^2\right)$. The measurement covariance matrix has the form $R_k = \text{diag}\left(\sigma_x^2.I, \sigma_r^2, \sigma_h^2\right)$.

When no measurement of the mobile object is received from the perception module, only the state prediction is executed. Therefore the mobile object position is extrapolated with the last estimated velocity. When new measurements are received, the update step resumes. In the case of multiple objects being tracked, an index is associated to each object. In case of loss of measurements, the prediction is performed as described previously and when new measurements are received, the tracking is simply based on the shortest distance between predicted positions and newly measured ones.

4.2. Delays

The first method introduces a small computation delay between the time step at which the images are collected and the time step at which the cylinder measurement vector is actually received by the prediction algorithm. This delay, denoted by Δt, is estimated using time stamps and a correction is applied to the obstacle measured position as

$$X = X_r + \Delta t.I.\hat{V} \tag{7}$$

where X_r is the received position, $\hat{V}$ the current estimated velocity from the Kalman filter and X the corrected position which is then included in (4).

4.3. Trajectory prediction

Since the model predictive controller of the autonomous vehicle operates on a finite prediction horizon of H_p time steps (see Section 5), the future positions $\tilde{X}_i$ $(i \in [k+1, k+H_p])$ of the mobile object are predicted on the same horizon using the filtered position and velocity,

$$\tilde{X}_k = \widehat{X}_k \tag{8}$$

$$\forall i \in [k+1; k+H_p]\,,\ \tilde{X}_{i+1} = \tilde{X}_i + t_e.\widehat{V}_i \tag{9}$$

$\Xi_k = \left\{\tilde{X}_k^{(j)}, j = 1, \ldots, N_{\text{obs}}\right\}$ is defined as the set containing the predicted positions of all the N_{obs} detected mobile obstacles at time k, in the case of several obstacles.

4.4. Obstacle mapping

Collision avoidance is classically addressed with a distance map, $p \mapsto d_{\text{obs}}(p)$ which returns at any position p in space the distance to the closest obstacle [17, 20]. This strategy can also be applied to mobile obstacles, with an additional time dependence to the mobile object predicted position: the collision checking function becomes dynamic. For each time step k in the prediction horizon, the arguments of the distance map at time k are thus the MAV predicted position p_k and the set of predicted obstacle positions Ξ_k: $d_{\text{obs}}(p_k, \Xi_k)$. This function is computed along two different strategies, depending on the dimension of the search space.

4.4.1 Two-dimensional case

As in [24], 2D collision distance maps are created by applying a morphological Euclidean distance transform (EDT) on a binary occupancy grid. This procedure has been adapted to the case of dynamical obstacles by managing in parallel a binary occupancy grid and a 2D collision map for each time step of the prediction horizon (H_p steps). For each predicted position of a mobile object, the square containing the 2D projection of its bounding cylinder is included in the binary map corresponding to the time step. When all active mobile objects have been processed, an Euclidian Distance Transform map is computed from each binary grid to return the distance to the nearest obstacle at any position and a given future time step. Static obstacles can be easily taken into account by incorporating them as an initialization of the binary grids.

4.4.2 Three-dimensional case

Since the generation of a complete distance map is more computationally demanding in three dimensions, the exact distance to the closest point of the cylinder with respect to the predicted MAV position p is instead computed on request. Following the method described in [4], this geometrical computation consists in projecting the test point p on the cylinder unit axis u, which yields one along-axis component $x_u = (c-p)\cdot u$ and one perpendicular component $y_u = \sqrt{\|c-p\|^2 - x_u^2}$, where c is the cylinder center position. The collision distance can then be computed by checking in which Voronoi intersection region the test point is located (inside, on the side, above, below). Static obstacles can be taken into account separately by using an Octomap model and an associated EDT that provides the distance to the closest static obstacle [17]. The collision distance function $d_{\text{obs}}(\cdot,\cdot)$ should then return the closest obstacle between the mobile and static ones, at each prediction step.

5. Model Predictive control

Model Predictive Control (MPC) is a widely-used control approach for waypoint rallying and trajectory tracking in constrained environments [12], which is why it has been selected here for the safe autonomous navigation mission in presence of mobile obstacles.

A dynamical model of the vehicle $\xi_k = f(\xi_{k-1}, U_{k-1})$ is exploited to build future state trajectories $\mathbf{X}_k$ on a finite time horizon H_p. Based on this prediction, a control input sequence $\mathbf{U}_k$ on a control horizon H_c (with $H_c \le H_p$) is computed as the optimal solution to the minimization of a multi-criterion cost function $J(\mathbf{U}_k, \mathbf{X}_k)$. The first element of this input sequence is applied on the vehicle, and the procedure is repeated at each time step to take into account changes in the environment. This is formalized as

$$\mathbf{U}_k = \{U_k, U_{k+1}, \ldots, U_{k+H_c-1}\} \quad (10)$$
$$\mathbf{X}_k = \{\xi_{k+1}, \xi_{k+2}, \ldots, \xi_{k+H_p}\} \quad (11)$$

$$\begin{gathered}\mathbf{U}_k^* = \arg\min_{\mathbf{U}_k \in \mathbb{U}^{H_c}} J(\mathbf{U}_k, \mathbf{X}_k) \\ \text{subject to } \xi_i = f(\xi_{i-1}, U_{i-1}), \\ \forall i \in [k+1; k+H_p]\end{gathered} \quad (12)$$

This strategy is applied on the two target platforms with specific adaptations of the cost functions.

5.1. Mobile robot

It is driven by the kinematic model

$$\begin{cases} p_k = p_{k-1} + t_e v_{k-1} \begin{bmatrix} \cos\theta_{k-1} \\ \sin\theta_{k-1} \end{bmatrix} \\ \theta_k = \theta_{k-1} + t_e \omega_{k-1} \end{cases} \quad (13)$$

where p is the 2D robot position and θ its heading angle. The control inputs are the linear speed v and the angular speed ω. The cost function in (12) is defined as the weighted sum

$$J = w_{\text{wp}} J_{\text{wp}} + w_{\text{obs}} J_{\text{obs}} \quad (14)$$

of a navigation term J_{wp} and an obstacle avoidance term J_{obs}, with associated weights $w_{(\cdot)}$ [5].
The navigation cost is the sum of the distances between the waypoint p_w to be reached and the predicted robot positions on H_p.

$$J_{\text{wp}}^{(\text{rob})} = \frac{1}{H_p v_{\max} t_e} \sum_{i=k+1}^{k+H_p} \|p_w - p_i\|^2 \quad (15)$$

The maximum velocity $v_{\max}$ serves as a normalization factor. The avoidance term J_{obs} penalizes the trajectories in which the predicted robot positions are in collision with the future positions of a mobile obstacle.

$$J_{\text{obs}} = \frac{1}{H_p} \sum_{i=k+1}^{k+H_p} f_{obs}\left(d_{\text{obs}}(p_i, \Xi_i)\right). \quad (16)$$

Construction of the distance function has been described in Section 4.4. The function f_{obs} normalizes this distance into a cost between 0 and 1, such that $f_{obs}(d)$ is equal to 1 if $d < d_{\text{safe}}$ and to zero if $d > d_{\text{des}}$ with a smooth decrease in-between, as depicted in Fig. 5. The relative values of the weights w_{wp} and w_{obs} define the importance of each criterion one with respect to the other. The obstacle cost is equal to zero if there is no obstacle in collision on the predicted trajectory, therefore it is fixed to a large value so that it dominates the navigation cost in case of emergency.

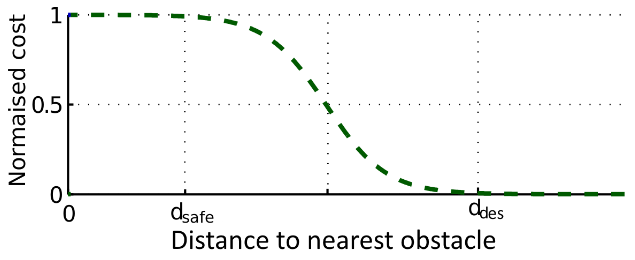


Figure 5: Penalty function for obstacle avoidance [5]

5.2. Micro-Air Vehicle

The MAV is assumed to be controlled directly in acceleration, which is afterwards translated into a thrust value and associated roll and pitch angles transmitted to the low-level Asctec controller. The yaw was kept constant in these experiments, using a simple proportional controller. The discretized MAV guidance model is thus directly

$$\begin{cases} p_k = p_{k-1} + t_e \mathcal{V}_{k-1} + \frac{t_e^2}{2} U_{k-1} \\ \mathcal{V}_k = \mathcal{V}_{k-1} + t_e U_{k-1} \end{cases} \quad (17)$$

where p and $\mathcal{V}$ are the 3D MAV position and velocity in inertial frame, and U is the linear acceleration which serves as control input.

In the case of the MAV, since both position and velocity are controlled, a reference trajectory $\mathcal{T} = \{\mathcal{P}^{\text{ref}}, \mathcal{V}^{\text{ref}}\}$ is defined in the inertial frame as a straight line from the starting point to the waypoint p_w with a constant reference velocity v_{nom} (acceleration ramps are added at the start and at the end). It is discretized with the same time step t_e as the control loops. The MAV control input is defined as the sum

$$U_k = U_k^{\text{lin}} + \mathbf{1}_{\text{obs}} U_k^{\text{avoid}} \quad (18)$$

where U_k^{lin} is the optimal solution for trajectory tracking only, and U_k^{avoid} is an additional component for collision avoidance which is computed only if a collision is detected along the predicted MAV trajectory (case $\mathbf{1}_{\text{obs}} = 1$). A linear unconstrained MPC formulation makes it possible to compute the control input U_k^{lin} as the first element of the optimal sequence minimizing

$$\mathbf{U}_k^{\text{lin}} = \arg\min_{\mathbf{U}_k} J_{\text{lin}}(\mathbf{U}_k, \mathbf{X}_k, \mathcal{T}) \quad (19)$$

$$J_{\text{lin}} = \sum_{i=k}^{k+H_p} \left\| [p_i, V_i] - [\mathcal{P}_i^{\text{ref}}, \mathcal{V}_i^{\text{ref}}] \right\|_Q^2 + \|U_i - U_{i-1}\|_R^2 \quad (20)$$

It can be shown that this tracking problem can be solved using a LQ controller (see [19] for more details).

If a moving obstacle has been detected and its predicted position intersects the trajectory of the MAV controlled by applying $\mathbf{U}_k^{\text{lin}}$, the second control input contribution U_k^{avoid} is then obtained by minimizing a cost function with the same formulation as (14). The navigation cost is very similar to (15), defined as

$$J_{\text{wp}}^{(\text{mav})} = \frac{1}{H_p v_{\text{nom}} t_e} \sum_{i=k+1}^{k+H_p} \|\mathcal{P}_i^{\text{ref}} - p_i\|^2 \qquad (21)$$

The only difference here is that the position error is computed with respect to the reference position at time k. The collision avoidance cost is exactly the one from equation (16), with the collision distance computed as described in Section 4.4.2.

5.3. Computation of optimal cost

For computational tractability and since no analytical solution is available, a systematic search approach on a discrete set is used to find a suboptimal solution to the optimization problem (12). The same control input is also applied on the entire control horizon, to reduce complexity. As in [23], the sets of candidate control inputs are defined to contain the bounds of the input space as well as the null input, and distribute the other values with an increased density around the null input. The number of candidates can be adapted to be compatible with the time step t_e governing the control loops.

6. Experiments

Experiments were conducted with the mobile robot and the MAV presented in Section 2 in realistic conditions, with no specific prior information available on the mobile obstacle characteristics or motion.

6.1. Mobile Robot

These experiments are representative of a urban navigation scenario. The robot should reach a waypoint located 10 meters in front of the starting position. A pedestrian crosses the path of the robot from the right side with respect to the robot general direction. The speed of the pedestrian is about $1\ \text{m/s}$ and that of the robot $0.4\ \text{m/s}$. The time step is equal to $t_e = 0.25$ s. Results on a typical run are presented in Fig. 6, with a focus on different moments of the experiment. It can be seen that this obstacle is correctly detected by the dense detection algorithm (the few outliers are filtered by the motion prediction module) and that the MPC algorithm achieves a safe avoidance (within a parameterized safety distance of 2 m) before eventually reaching the waypoint.

6.2. Micro-air vehicle

The MAV mission is to rally a point 5 meters ahead at a speed of 0.3 m/s, while a mobile obstacle crosses the reference trajectory from the left. This obstacle is a foam pendulum hung to a rope and moved using pulleys by a human operator (Fig. 7). Ground truth localization from a motion capture system are available for both the MAV and the mobile object, which makes it possible to evaluate the performance of the vision algorithm as well as the safety level of the control algorithm (target separation distance). Four successful experiments have been performed. Evaluation is based on the ground truth distance d between the MAV and the mobile obstacle, whenever it is below an activation distance (fixed here to 2.5 m). More precisely, the score is the sum of weights computed over all time steps and defined as:

- 0 for emergency ($d < 0.8\ \text{m}$),
- 0.3 for risky ($0.8\ \text{m} < d < 1.2\ \text{m}$),
- 1 for optimal ($1.2\ \text{m} < d < 1.6\ \text{m}$),
- 0.5 for conservative ($d > 1.6\ \text{m}$).

The avoidance scores on the four experiments led to an average of 0.739 (std ± 0.083), in-between conservative and optimal. As concerns the tracking accuracy of the mobile object, the sparse method running on-board of the MAV obtained a distance RMS (averaged on the four tries) of 0.12 m (std ± 0.01 m), computed between the embedded estimate and the ground truth trajectory provided by the motion capture system.

7. Conclusions

This work has defined an integrated vision-based estimation and control loop to achieve safe autonomous navigation in presence of unknown mobile objects. Two methods relying on stereo-vision data have been proposed for detecting and estimating the position of obstacles. One of the vision-based estimation methods is dense and thus requires more computational power, which can be obtained through the assistance of a GPU board, while the second method relies on sparse features and therefore consumes fewer resources, compatible with a typical embedded PC of a MAV. They are associated with a motion prediction module and a MPC algorithm to define safe trajectories online. Experimental results have shown that, by selecting the vision algorithm adapted to the on-board computing resource, the methodology can be successfully embedded on a mobile robot and on a micro-air vehicle.

Acknowledgements

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

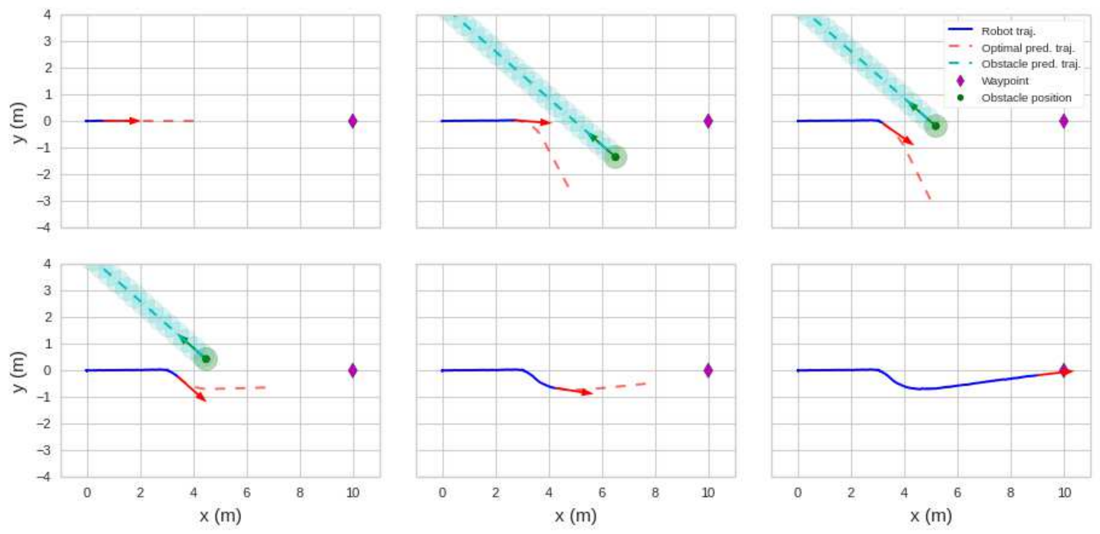
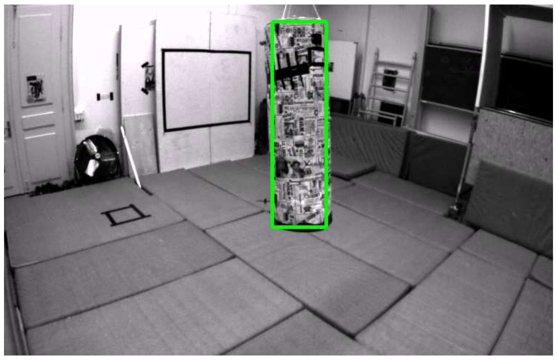
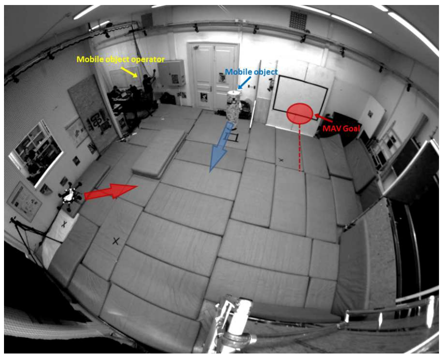


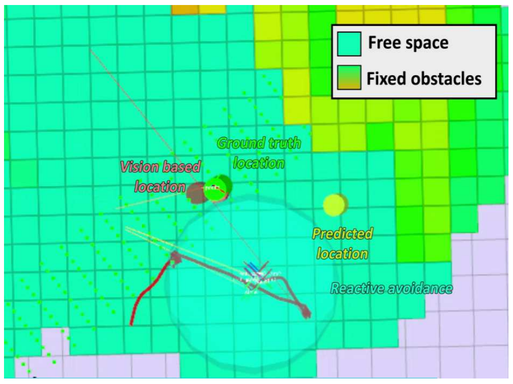
Figure 6: Experimental results with the mobile robot. Purple diamond: waypoint, Blue line: robot trajectory, Dashed red line: predicted optimal trajectory, Green point: obstacle center, Blue dashed line: predicted trajectory of the obstacle



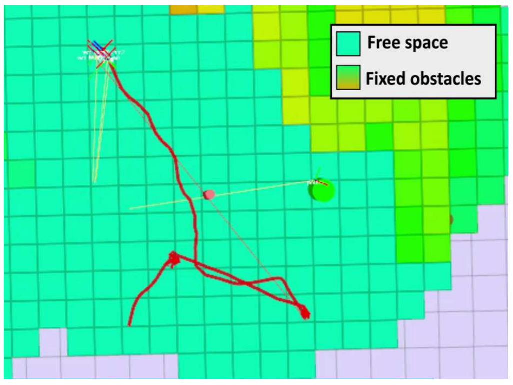
(a) Embedded mobile object detection in image (green rectangle)



(b) Experimental setup. Top left: human operator, Top middle: mobile object, Bottom left: MAV, Top right in red: waypoint.



(c) Rviz view (early detection). Green cylinder: current estimated object, Yellow cylinder: last object position on prediction horizon. Light blue circle: collision detection.



(d) Rviz view (end of mission). Light red line: reference trajectory, Thick red line: MAV trajectory with avoidance maneuver. Green cylinder: object estimated position.

Figure 7: MAV experiment: trajectory tracking with mobile object detection and avoidance

References

[1] P. F. Alcantarilla, J. J. Yebes, J. Almazán, and L. M. Bergasa. On combining visual slam and dense scene flow to increase the robustness of localization and mapping in dynamic environments. In *IEEE International Conference on Robotics and Automation*, pages 1290–1297, 2012. 2

[2] A. Bak, S. Bouchafa, and D. Aubert. Detection of independently moving objects through stereo vision and ego-motion extraction. In *IEEE Intelligent Vehicles Symposium*, pages 863–870, 2010. 2, 3

[3] A. Bak, S. Bouchafa, and D. Aubert. Dynamic objects detection through visual odometry and stereo-vision: a study of inaccuracy and improvement sources. *Machine vision and applications*, 25(3):681–697, 2014. 2, 3

[4] A. Barbier and E. Galin. Fast distance computation between a point and cylinders, cones, line-swept spheres and cone-spheres. *Journal of Graphics tools*, 9(2):11–19, 2004. 5

[5] S. Bertrand, J. Marzat, H. Piet-Lahanier, A. Kahn, and Y. Rochefort. MPC strategies for cooperative guidance of autonomous vehicles. *AerospaceLab Journal*, (8):1–18, 2014. 1, 6

[6] F. Bonin-Font, A. Ortiz, and G. Oliver. Visual navigation for mobile robots: A survey. *Journal of intelligent and robotic systems*, 53(3):263–296, 2008. 1

[7] A. Brooks, T. Kaupp, and A. Makarenko. Randomised MPC-based motion-planning for mobile robot obstacle avoidance. In *IEEE International Conference on Robotics and Automation, Kobe, Japan*, pages 3962–3967, 2009. 1

[8] C.-H. Chen, C. Cheng, D. Page, A. Koschan, and M. Abidi. A moving object tracked by a mobile robot with real-time obstacles avoidance capacity. In *18th International Conference on Pattern Recognition*, volume 3, pages 1091–1094, 2006. 1

[9] M. Derome, A. Plyer, M. Sanfourche, and G. Le Besnerais. Moving object detection in real-time using stereo from a mobile platform. *Unmanned Systems*, 3(4):253–266, 2015. 1, 3

[10] M. Derome, A. Plyer, M. Sanfourche, and G. Le Besnerais. A prediction-correction approach for real-time optical flow computation using stereo. In *German Conference on Pattern Recognition, Hannover, Germany*, pages 365–376, 2016. 1, 3

[11] D. Ferguson, M. Darms, C. Urmson, and S. Kolski. Detection, prediction, and avoidance of dynamic obstacles in urban environments. In *IEEE Intelligent Vehicles Symposium*, pages 1149–1154, 2008. 1

[12] R. Findeisen, L. Imsland, F. Allgower, and B. A. Foss. State and output feedback nonlinear model predictive control: An overview. *European Journal of Control*, 9(2-3):190–206, 2003. 5

[13] M. A. Fischler and R. C. Bolles. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM*, 24(6):381–395, 1981. 4

[14] C. Fulgenzi, A. Spalanzani, and C. Laugier. Dynamic obstacle avoidance in uncertain environment combining PVOs and occupancy grid. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 1610–1616, 2007. 1, 4

[15] B. Jung and G. S. Sukhatme. Detecting moving objects using a single camera on a mobile robot in an outdoor environment. In *International Conference on Intelligent Autonomous Systems*, pages 980–987, 2004. 1

[16] K. M. Krishna and P. K. Kalra. Detection, tracking and avoidance of multiple dynamic objects. *Journal of Intelligent and Robotic Systems*, 33(4):371–408, 2002. 1

[17] B. Lau, C. Sprunk, and W. Burgard. Efficient grid-based spatial representations for robot navigation in dynamic environments. *Robotics and Autonomous Systems*, 61(10):1116–1130, 2013. 5

[18] S. Lynen, M. W. Achtelik, S. Weiss, M. Chli, and R. Siegwart. 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*, pages 3923–3929, 2013. 2

[19] J. Marzat, S. Bertrand, A. Eudes, M. Sanfourche, and J. Moras. Reactive MPC for autonomous MAV navigation in indoor cluttered environments: Flight experiments. In *Proceedings of the 20th IFAC World Congress, Toulouse, France*, pages 16566–16572, 2017. 6

[20] J. Marzat, J. Moras, A. Plyer, A. Eudes, and P. Morin. Vision-based localization, mapping and control for autonomous MAV: EuRoC challenge results. In *15th ONERA-DLR Aerospace Symposium (ODAS), Toulouse, France*, 2015. 1, 5

[21] J. Nikolic, J. Rehder, M. Burri, P. Gohl, S. Leutenegger, P. T. Furgale, and R. Siegwart. A synchronized visual-inertial sensor system with FPGA pre-processing for accurate real-time SLAM. In *IEEE International Conference on Robotics and Automation, Hong Kong, China*, pages 431–437, 2014. 2

[22] A. Plyer, G. L. Besnerais, and F. Champagnat. Real-time Lucas-Kanade optical flow estimation for real-world applications. *Journal of Real Time Image Processing*, 2014. 3

[23] Y. Rochefort, H. Piet-Lahanier, S. Bertrand, D. Beauvois, and D. Dumur. Model predictive control of cooperative vehicles using systematic search approach. *Control Engineering Practice*, 32:204–217, 2014. 1, 7

[24] H. Roggeman, J. Marzat, M. Sanfourche, and A. Plyer. Embedded vision-based localization and model predictive control for autonomous exploration. In *IROS Workshop on Visual Control of Mobile Robots (ViCoMoR), Chicago IL, USA*, pages 13–20, 2014. 1, 5

[25] V. Romero-Cano and J. I. Nieto. Stereo-based motion detection and tracking from a moving platform. In *IEEE Intelligent Vehicles Symposium*, pages 499–504. IEEE, 2013. 2

[26] M. Sanfourche, V. Vittori, and G. Le Besnerais. 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, November 3-8*, pages 2107–2114, 2013. 1, 2, 4

[27] J. Shi and C. Tomasi. Good features to track. In *IEEE Conference on Computer Vision and Pattern Recognition, Seattle, WA, USA*, pages 593–600, 1994. 4

[28] B. Siciliano, F. Caccavale, E. Zwicker, M. Achtelik, N. Mansard, C. Borst, M. Achtelik, N. O. Jepsen, R. Awad, and R. Bischoff. EuRoC - the challenge initiative for european robotics. In *Proceedings of the 41st International Symposium on Robotics, Munich, Germany*, pages 1–7, 2014. 2

[29] M. Sizintsev and R. P. Wildes. Coarse-to-fine stereo vision with accurate 3d boundaries. *Image and Vision Computing*, 28(3):352–366, 2010. 3

[30] A. Talukder and L. Matthies. Real-time detection of moving objects from moving vehicles using dense stereo and optical flow. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, volume 4, pages 3718–3725, 2004. 2

[31] A. Tsalatsanis, K. Valavanis, and A. Yalcin. Vision based target tracking and collision avoidance for mobile robots. *Journal of Intelligent and Robotic Systems*, 48(2):285–304, 2007. 1

[32] A. Wedel, T. Brox, T. Vaudrey, C. Rabe, U. Franke, and D. Cremers. Stereoscopic scene flow computation for 3d motion understanding. *International Journal of Computer Vision*, 95(1):29–51, 2011. 2