Prediction of the scene quality for stereo vision-based autonomous navigation

Hélène Roggeman * **Julien Marzat** *
Anthelme Bernard-Brunel * **Guy Le Besnerais** *

* *ONERA – The French Aerospace Lab, F-91123, Palaiseau, France.*
e-mail: firstname.lastname@onera.fr.

Abstract: This paper presents an autonomous navigation architecture for a robot using stereo vision-based localisation. The main contribution is the prediction of the quality of future localisation of the system in order to detect and avoid areas where vision-based localisation may fail, due to lack of texture in the scene. A criterion based on the estimation of future visible landmarks, considering uncertainties on landmarks and camera positions, is integrated in a Model Predictive Control loop to compute safe trajectories with respect to the visual localisation. The system was tested on a mobile robot and the obtained results demonstrate the effectiveness of our method.

Keywords:
Autonomous vehicles, Robot navigation, Stereo vision, Model Predictive Control

1. INTRODUCTION

This work takes place in the context of autonomous navigation in unknown indoor environments with Micro Air Vehicles or mobile robots. In this kind of environment, there are no global localisation systems available such as GPS. Vision sensors are then a usual solution for estimating the localisation of a vehicle. Using visual landmarks in the images, a visual odometry algorithm can be used to compute the position and orientation of the camera and thus of the system. However, if the scene contains too few landmarks, for instance when the robot faces a white wall, the algorithm cannot extract enough landmarks to compute the localisation accurately. In this case, the mission is likely to fail. In this work, we aim at ensuring that the system remains well-localised during the mission time. To reach this objective, we have developed a criterion that predicts the quality of the scene for localisation, using already known landmarks. We have then integrated this criterion into a Model Predictive Control (MPC) loop so as to demonstrate its efficiency on a real robot.

1.1 Related work

In the literature, some authors were interested in the ability to predict the quality of future measurements to improve navigation algorithms, (see Makarenko and et al. (2002), Bourgaul et al. (2002), Vidal-Calleja et al. (2006), Sim and Roy (2005)) or environment reconstruction algorithms (Forster et al. (2014), Dunn et al. (2009)). Some authors propose a criterion based on the Shannon entropy (see Bourgaul et al. (2002), Bachrach et al. (2012), Sim and Roy (2005)). Vidal-Calleja et al. (2006) choose the optimal control to reduce uncertainties on the camera and landmark positions using a criterion based on the mutual information. Forster et al. (2014) use the information gain to choose the trajectory that maximizes the precision of the environment reconstruction. Other approaches rely on a criterion based on image or scene geometry. Dunn et al. (2009) look for a Next-Best-View, considering uncertainties on measurements and scene appearance. Sadat et al. (2014) compute a texture criterion from the local density of triangles in the 3D mesh used for environment reconstruction. This criterion is used for planning trajectories with RRT* in real time towards a desired goal. Mostegel et al. (2014) propose a criterion which accounts for the geometric quality of landmarks (triangulation angle, proof of existence) and for the ability of recognizing each point in order to improve waypoint navigation. In all these references, the authors use a monocular camera, whereas, in our case, we use a stereo rig. It allows us to directly access the depth information for each point. Moreover, unlike most of these references, we take into account uncertainties on the landmark positions and the camera position.

1.2 Overview

The basic mission considered in this work is to perform autonomous navigation between waypoints with a mobile robot. The environment can have relatively textureless areas that are a problem for the visual localisation. This is why the robot has to detect these zones in order to choose safe trajectories to navigate. To locate itself, the robot is equipped with a stereo rig, composed of two fixed cameras with known calibration. The stereo images received from the rig are used to compute the pose with a visual odometry algorithm. The main steps of such an algorithm are:

- Interest points, like Harris (Harris and Stephens (1988)), are extracted from both images and matched.
- The 3D positions of the corresponding points in the scene are computed by triangulation.

- The motion of the left camera is estimated from the apparent displacement of the projections of the 3D points in the images.

In this work, we have choosen to use eVO, a visual odometry algorithm described in Sanfourche et al. (2013). Other stereo algorithms could be considered as well, e.g. Klein and Murray (2007). A criterion is proposed to evaluate the quality of the scene that the robot will encounter in a future position. It uses the 3D points sent by the visual odometry algorithm to predict the visible points in the images in the future, considering the uncertainties on the landmark positions and on the movement of the system. The criterion is explained in Section 2. We have set up experiments to demonstrate the relevance of this criterion, which are described in Section 2.5. A navigation strategy based on MPC is presented in Section 3. It integrates our localisation quality criterion with waypoint navigation and an obstacle avoidance criterion. Experimental results of navigation are shown in Section 4.

2. LOCALISATION QUALITY CRITERION

Let us first introduce some notations. $Y = (x, y, z)^T$ is a 3D point, expressed in a global frame $(\mathcal{W})$, unless otherwise stated, defined by the camera position and orientation at the beginning of the mission. The camera frame $(\mathcal{C}_n)$ is the frame defined by the current camera position and orientation. At t_0, $(\mathcal{W})$ and $(\mathcal{C}_n)$ are coincident. For a vector v, $\tilde{v} = (v, 1)^T$ is the augmented vector.

2.1 Projection of a 3D point

Given a 3D point computed by the visual odometry algorithm and a desired camera position, the 2D projection of the 3D point in the corresponding image is computed. First, the 3D point is expressed in the camera frame, then, the projection is computed.

Change of basis T is the translation vector and R is the rotation matrix from camera frame to global frame. The new coordinates of the 3D point Y, denoted Y', are

$$\tilde{Y'} = P^{-1}\tilde{Y} = \begin{pmatrix} R & T \\ 0 & 1 \end{pmatrix}^{-1} \tilde{Y} \tag{1}$$

Camera model Let $p = (u, v)^T$ denote the image projection point of a 3D point $Y = (x, y, z)^T$, expressed in the camera frame. α is the focal length and (u_0, v_0) are the optical center coordinates.

$$\tilde{p} = z^{-1}KY = z^{-1} \begin{pmatrix} \alpha & 0 & u_0 \\ 0 & \alpha & v_0 \\ 0 & 0 & 1 \end{pmatrix} Y \tag{2}$$

2.2 Uncertainty computation

The goal is to estimate the uncertainty on the position of a projected point on the image after a camera displacement, denoted with rotation and translation (R, T). The rotation matrix is written as

$$R = R_z(\theta_z)R_y(\theta_y)R_x(\theta_x) \tag{3}$$

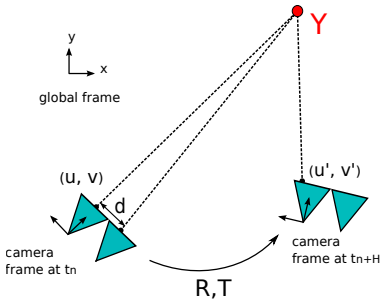

Fig. 1. Computing the 2D projection of a point in a future camera image: (u, v, d) and (R, T) are known. Y is triangulated using (u, v, d). (u', v') is deduced from Y and (R, T).

where $R_\bullet$ is the rotation about the $\bullet$ axis by an angle $\theta_\bullet$, and the translation vector is written

$$T = (t_x, t_y, t_z)^T \tag{4}$$

$\Theta = (\theta_x, \theta_y, \theta_z, t_x, t_y, t_z)$ is the vector of the displacement parameters and (u, v, d), the projected point position in the left camera and disparity. The uncertainties on the triangulation of the 3D point and on the estimation of the camera position after the displacement are taken into account. We model the uncertainties on Θ and (u, v, d) by a normal distribution with zero mean. The covariance matrices are denoted by Σ_Θ and $\Sigma_{u,v,d}$. Moreover, the uncertainties between Θ and (u, v, d) are assumed to be independent. As shown in Figure 1, $p = (u, v)$ is the projected point in the image in the first position and p' the projected point in the image after the displacement of the camera.

Expression of p' To express p' as a function f of (Θ, u, v, d), we use a triangulation function denoted by Π^{-1}, the change of basis (R, T) and a projection function denoted by Π:

$$p' = f(\Theta, u, v, d)$$
$$p' = \Pi(Y'(\Theta, u, v, d)) = \Pi\left(R(\Theta) \cdot \Pi^{-1}(u, v, d) + T(\Theta)\right) \tag{5}$$

Covariance of p' As the uncertainties between Θ and (u, v, d) are independent, the covariance on the position of p' can be written as

$$\Sigma_{p'} = J_{f_\Theta} \cdot \Sigma_\Theta \cdot J_{f_\Theta}^T + J_{f_{u,v,d}} \cdot \Sigma_{u,v,d} \cdot J_{f_{u,v,d}}^T \tag{6}$$

with J_{f_Θ} and $J_{f_{u,v,d}}$ the Jacobian matrices of f with respect to Θ and (u, v, d), respectively.

$$J_{f_\Theta} = \frac{\partial f}{\partial \Theta} = J_\Pi(Y') \cdot J_{Y'}(Y) \tag{7}$$

$$J_{f_{u,v,d}} = \frac{\partial f}{\partial u, v, d} = J_\Pi(Y') \cdot R \cdot J_{\Pi^{-1}}(u, v, d) \tag{8}$$

J_Π, $J_{\Pi^{-1}}$ and $J_{Y'}$ are the Jacobian matrices of the projection function, the triangulation function and the change of basis function, described in the following paragraphs.

Jacobian matrix of the triangulation function The 3D position of a point is computed with the coordinates of the projected point in the image and the disparity.

$$Y = \Pi^{-1}(u, v, d) = \frac{-b}{d} \cdot \begin{pmatrix} u - u_0 \\ v - v_0 \\ \alpha \end{pmatrix} \quad (9)$$

where b denotes the baseline between the left and right camera.

$$J_{\Pi^{-1}}(u, v, d) = \frac{\partial \Pi^{-1}(u, v, d)}{\partial u, v, d} = \frac{b}{d^2} \cdot \begin{pmatrix} -d & 0 & u - u_0 \\ 0 & -d & v - v_0 \\ 0 & 0 & \alpha \end{pmatrix} \quad (10)$$

Jacobian matrix of the projection function The image position of a projected point in the left image is computed using the 3D point coordinates, expressed in the camera frame.

$$p' = \Pi(Y') = \begin{pmatrix} \frac{\alpha}{z'} \cdot x' + u_0 \\ \frac{\alpha}{z'} \cdot y' + v_0 \end{pmatrix} \quad (11)$$

$$J_{\Pi}(Y') = \frac{\partial \Pi(Y')}{\partial Y} = \frac{\alpha}{z'^2} \cdot \begin{pmatrix} z' & 0 & -x' \\ 0 & z' & -y' \end{pmatrix} \quad (12)$$

Jacobian matrix of the change of basis function

$$J_{Y'}(Y) = \left.\frac{\partial Y'}{\partial \Theta}\right|_Y = \left[\frac{\partial R}{\partial \theta_x, \theta_y, \theta_z} \cdot Y \;\middle|\; \frac{\partial T}{\partial t_x, t_y, t_z}\right] \quad (13)$$

The derivative of the rotation matrix (see equation 3) is

$$\frac{\partial R}{\partial \theta_x, \theta_y, \theta_z} = \left[R_z R_y \frac{\partial R_x}{\partial \theta_x} \;\middle|\; R_z \frac{\partial R_y}{\partial \theta_y} R_x \;\middle|\; \frac{\partial R_z}{\partial \theta_z} R_y R_x\right] \quad (14)$$

2.3 Estimating the probability

The aim is to estimate the probability that a projected point p' lies inside the image after a displacement of the system. From the point's covariance Σ_p (obtained from (6)), a confidence ellipse is deduced and the area of its intersection with the image support is computed. In this Section, $X = (x, y)^T$ is an image point, $(\mathcal{M})$ is the image frame and $(\mathcal{M}')$ is the ellipse frame (whose origin is the point p' and frame axes are the ellipse axes). The ellipse equation in frame $(\mathcal{M})$ is

$$(X - p')^T \Sigma_p^{-1} (X - p') = s \quad (15)$$

$s = 4.605$ for a 90% confidence ellipse, this value can be found in a table of χ^2 distribution with 2 degrees of freedom. To simplify the computation of the area, it is done in the ellipse frame $(\mathcal{M}')$. The ellipse equation in the ellipse frame is then

$$X'^T \Sigma_D^{-1} X' = s \quad (16)$$

where $\Sigma_D = P^T \Sigma_p P$ is a diagonal matrix and $X' = P^T (X - p')$. λ_1 and λ_2 are the eigenvalues of Σ_p. The ellipse semi-axes lengths are $a = \sqrt{s\lambda_1}$ and $b = \sqrt{s\lambda_2}$ and are directed by the eigenvectors of Σ. The total area of the ellipse is $A_{ell} = \pi ab$. To compute the area of the intersection, the intersection points between the ellipse and the lines corresponding to the borders of the image are found, expressed in the ellipse frame $(\mathcal{M}')$. The intersection points are found by solving the system composed by the equation (16) and the equation of a line $L^T \tilde{X}' = 0$ with $L = (m, n, q)$. This problem is equivalent to solving a quadratic equation on x or y. If there is no intersection, the area is $A = A_{ell}$ or $A = 0$, depending on whether the point is definitively inside or outside the image. In the other cases, we compute the double integral on the domain delimited by the ellipse and the lines delimiting the image.

$$D = \{X \in \mathbb{R}^2 \mid X^T \Sigma_D^{-1} X \leqslant s \text{ and } L^T \tilde{X} \leqslant 0\} \quad (17)$$

$$A = \iint_D dxdy \quad (18)$$

The probability is estimated by the ratio of this area to the ellipse area.

$$P_r = \frac{A}{A_{ell}} \quad (19)$$

2.4 Algorithm

At time t_n, the current pose P_n and a set of 3D points $\mathcal{Y}_n = (Y_i)_{i \in [1,N]}$, as defined in Section 2, are known, as well as the camera matrix K, thanks to a camera calibration. The objective is to predict the number of points which will be seen at time t_{n+H}, where H is a parameterized prediction horizon (see Section 3). The desired pose P_{n+H} is known. The points are projected onto the corresponding image plane: first, a change of basis is done with equation (1) to express the points in the camera frame. Next, the position of the projection in the image is calculated with (2). Then, the uncertainty of each projected point is computed and the probability of being in the image is evaluated by the ratio (19). Finally, the number of points with a probability higher than a threshold s_{proba} are counted. Algorithm 1 summarizes the procedure.

Algorithm 1 Algorithm at time t_n

$N \leftarrow 0$
Require: $\mathcal{Y}_n$, P_{n+H}, K, Σ_Θ and $\Sigma_{u,v,d}$
  for Y in $\mathcal{Y}_n$ **do**
    Change of basis: $\tilde{Y}' = P_{n+H} \tilde{Y}$
    Projection: $\tilde{p}' = z^{-1} K Y'$
    Jacobian matrices computation J_Π, J_{T_R} et $J_{X'}$
    Covariance computation $\Sigma_{p'}$
    Computation of P_r, the probability of p' to be in the image
    if $P_r > s_{proba}$ **then**
      $N \leftarrow N + 1$
    end if
  end for
  return N

2.5 Criterion validation

We have first conducted open loop experiments in order to validate the proposed criterion on real images. The same trajectory is repeated three times with a mobile robot, shown in Figure 5. At each passage, some markers are added on the walls, to add more texture to the scene. The three trajectories are denoted by: Sequence 1, (without markers), Sequence 2, (with a few markers added) and Sequence 3, (with more markers). In the first sequence, there is one location where the scene has few interest points, highlighted by with a circle in the first trajectory in Figure 4. The images from this location for the three trajectories are shown in Figure 2, with the projected points. For the three trajectories, the criterion is evaluated

in each position with different horizons H. The goal of these experiments is to check that the number of features is well predicted and that the algorithm can detect the relatively textureless area. The parameters are set as follows:

- Time step: $t_e = 0.25$ s
- Probability threshold: $s_{proba} = 0.5$
- Uncertainties $\Sigma_\Theta = \text{diag}(\sigma^2_{\theta_x}, \sigma^2_{\theta_y}, \sigma^2_{\theta_z}, \sigma^2_x, \sigma^2_y, \sigma^2_z)$ and $\Sigma_{u,v,d} = \text{diag}(\sigma^2_u, \sigma^2_v, \sigma^2_d)$ with:
  - position: $\sigma_x = \sigma_y = \sigma_z = 0.005$ m
  - orientation: $\sigma_{\theta_x} = \sigma_{\theta_y} = \sigma_{\theta_z} = 0.001$ rad
  - image point position: $\sigma_u = \sigma_v = 0.2$ pixel
  - disparity: $\sigma_d = 0.4$ pixel

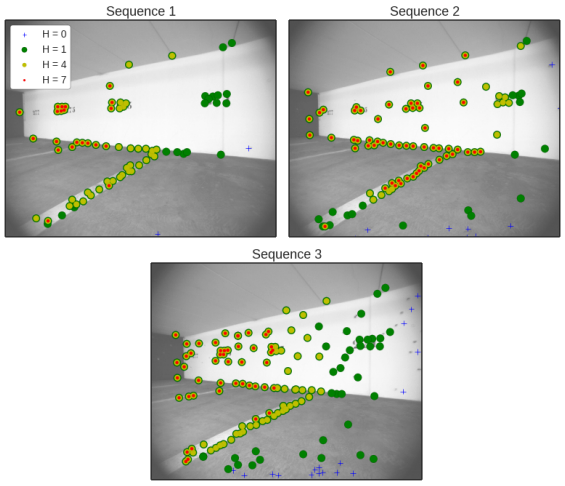

Fig. 2. Images and projected points at time t_n and their predicted visibility in the future for different horizons H: in green, the visible points at time t_{n+1}, in yellow, the visible points at time t_{n+4} and in red, the visible points et time t_{n+7}. The robot follows a trajectory that turns to the left.

In Figure 3, the number of predicted landmarks and the number of actually seen landmarks are compared. It can be noted that the number of predicted landmarks is slightly overvalued but it follows the variations of the real number. For instance, at time $t = 200$, the number of visible landmarks falls, this event was well predicted by the proposed criterion. In Figure 4, each trajectory is drawn with the number of predicted points computed at each time step. At the circled zone, it can be seen that the number of predicted landmarks is very low for sequence 1 and higher for the sequences 2 and 3.

As a conclusion, the experiments demonstrate that the proposed criterion correctly predicts the number of visible features.

3. AUTONOMOUS NAVIGATION

3.1 Robot model

$X = (x, y, \theta)^T$ is the state vector, with (x, y) the robot position and θ the heading angle. $U = (v, \omega)^T$ is the command vector, with v and ω linear speed and angular speed. The kinematical model of the system can be written as

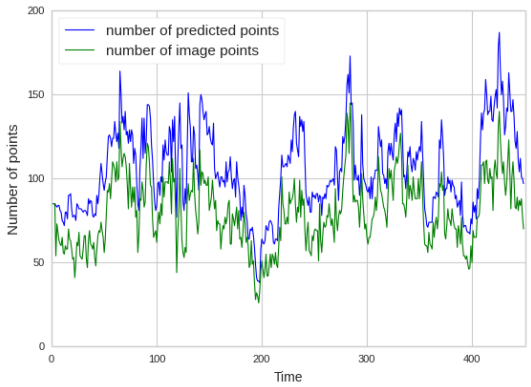

Fig. 3. Number of predicted landmarks and number of landmarks actually seen in the first sequence with $H = 1$.

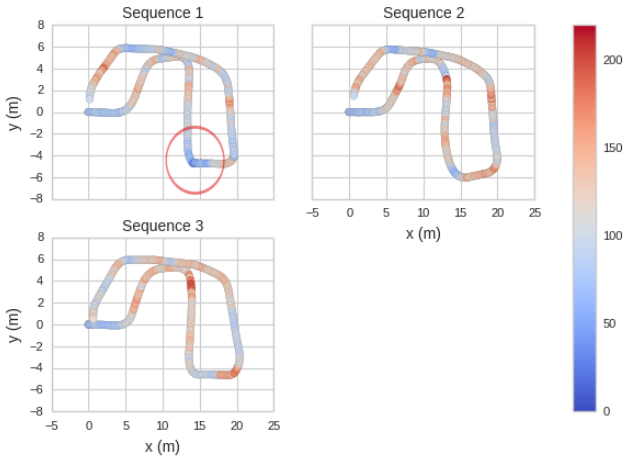

Fig. 4. Number of predicted landmarks for each trajectory with a horizon $H = 1$. The circle shows the place where the number of visible features is very low for sequence 1.

$$X_n = g\left(X_{n-1}, U_{n-1}\right) \tag{20}$$

$$\begin{cases} x_n = x_{n-1} + t_e v_{n-1} \cos\theta_{n-1} \\ y_n = y_{n-1} + t_e v_{n-1} \sin\theta_{n-1} \\ \theta_n = \theta_{n-1} + t_e \omega_{n-1} \end{cases} \tag{21}$$

with t_e the sampling period.

3.2 MPC

Model Predictive Control is a discrete time command strategy. Using the dynamical model of the system, some trajectories are predicted on a finite horizon from a sequence of control inputs. For each trajectory, a cost function which represents the objectives of the mission, is computed. Finally, the first command of the sequence corresponding to the minimal cost is applied to the system and the operation is performed again for the next time step.

The sequences of control inputs, and corresponding states, computed from the model (21) are denoted by

$$\mathcal{U}_n = \{U_n, U_{n+1}, \ldots, U_{n+H_c-1}\} \tag{22}$$

$$\mathcal{X}_n = \{X_{n+1}, X_{n+2}, \ldots, X_{n+H_p}\} \tag{23}$$

H_c is the control horizon and H_p the prediction horizon. After the step H_c, the control $U_i = (v_{H_c}, 0), i \in [H_c, H_p]$ is used. The controls inputs are bounded by $(-v_{\max}, -\omega_{\max})^T$ and $(v_{\max}, \omega_{\max})^T$.

A cost function $J(\mathcal{U}_n, \mathcal{X}_n)$ is defined and optimized to obtain the optimal control $\mathcal{U}_n^*$.

$$\begin{aligned} \mathcal{U}_n^* &= \arg\min_{\mathcal{U}_n} J(\mathcal{U}_n, \mathcal{X}_n) \\ &\text{with } X_k \text{ satisfying (21)}, \\ &\forall k \in [n+1, n+H_p] \end{aligned} \tag{24}$$

The first component U_n^* of the optimal solution is applied to the system.

3.3 Cost function

The cost function is defined as

$$J = w_{\text{loc}} J_{\text{loc}} + w_{\text{wp}} J_{\text{wp}} + w_{\text{obs}} J_{\text{obs}} \tag{25}$$

where

- J_{loc} is the localisation quality cost,
- J_{wp} is the waypoint navigation cost,
- J_{obs} is an obstacle avoidance cost.

Each cost is normalized between 0 and 1. The weights $w_\bullet$ tune the relative importance of each cost. To optimize the cost function, the control input space is discretised and each command is evaluated. The discretization is set so as to ensure that the optimal solution can be computed during a time step of the system. To limit complexity, it was chosen to apply the same linear and angular velocities on the entire control horizon.

Localisation quality cost The localisation quality cost penalizes the trajectories which have few predicted visible landmarks using the estimated number of points defined in Section 2.

$$J_{\text{loc}} = 1 - \frac{N(X_{n+H}, \mathcal{Y}_n)}{N_{\max}} \tag{26}$$

where $N(X_{n+H}, \mathcal{Y}_n)$ is the number of predicted landmarks at $n+H$ and $N_{\max}$ is the total number of 3D points. H is the time horizon where the prediction is computed, it must be fixed beforehand.

Navigation The following cost is built to reach a waypoint X_w, given the predicted positions of the robot $X_k = [x_k, y_k]^{\text{T}}$,

$$J_{\text{wp}} = \frac{1}{H_p v_{\max} t_e} \sum_{k=n+1}^{n+H_p} \|X_w - X_k\|^2 \tag{27}$$

Obstacle avoidance This cost penalizes the intersection of each predicted position in $\mathcal{X}_n$ with existing obstacles in the current occupancy grid. The obstacle avoidance cost is computed as

$$J_{\text{obs}} = \frac{1}{H_p} \sum_{k=n+1}^{n+H_p} f_o(X_k). \tag{28}$$

where f_o is a penalty function defined to give a high cost if there are nearby obstacles and a low cost if the obstacles are far. More details can be found in Roggeman et al. (2014).

4. EXPERIMENTAL RESULTS

4.1 Robotic platform

The experimental platform is a four-wheel Robotnik Summit XL (Figure 5) equipped with a Xtion sensor and a stereorig composed of two electronically synchronized USB cameras equipped with 2.8mm lens and separated by a 32cm long baseline. The point cloud sent by the Xtion sensor is used to compute an occupancy map which can be used to perform obstacle avoidance.

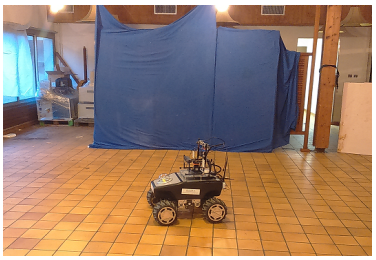

Fig. 5. The mobile robot in the experimental environment, the textureless wall is visible behind the robot

4.2 Experiments

The robot is placed at a starting point and has to reach a waypoint in front of a textureless wall. There are no obstacles in the environment, so w_{obs} is set to 0. The shortest path is to go straight towards the waypoint, but, in this case, the localisation is likely to fail. A better way would be to get around and approach the waypoint with a viewing angle that allows the robot to see more landmarks. The horizons are set at $H_c = 5$ and $H_p = 8$. The others parameters are fixed at the values defined in Section 2.5. We designed a first test to choose the horizon H for the localisation cost. With $H = 2$, the robot is too short-sighted. Preliminary results suggested that $H = 4$ provided satisfactory results, so it was kept for further analysis. Different weights w_{loc} were tested for the localisation quality criterion with a waypoint weight computed as:

$$w_{\text{wp}} = 1 - w_{\text{loc}} \tag{29}$$

In Figure 6, the odometry estimated with eVO is compared with the odometry computed by the wheel encoders, considered as the ground truth in these almost rectilinear trajectories. The localisation diverges when the localisation quality cost is not considered ($w_{\text{loc}} = 0.0$). For a small value of $w_{\text{loc}} = 0.1$, the divergence no longer occurs but the error remains important. When the weight is increased, the trajectories are no longer straight and the robot succeeds in getting closer to the waypoint with a good localisation. The behaviour of the robot when w_{loc} increases is related to the number of visible points predicted by our criterion, see Figure 7.

For each weight, the experiment was repeated ten times. For both localisation methods, the travelled distance was computed. Then, we computed the deviation of the travelled distance between our experimental values (eVO) and the ground truth (wheel encoders).

Figure 8 shows that the deviation consistently decreases when the weight w_{loc} increases until 0.5. $w_{\text{loc}} = 0.5$ seems

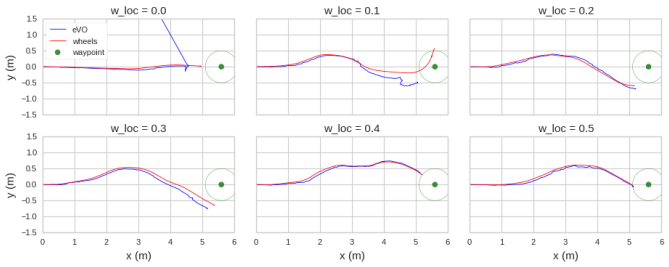

Fig. 6. Trajectories followed by the robot with different weights w_{loc}, given by eVO and wheel encoders. The green circle shows the area in which the waypoint is considered reached.

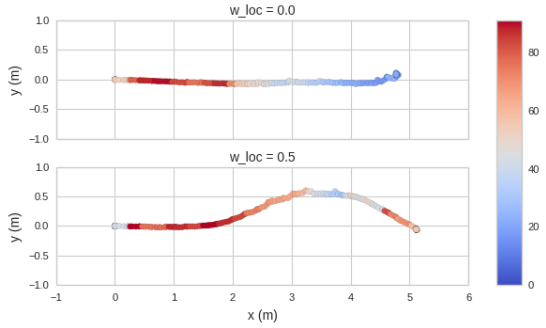

Fig. 7. Number of 3D points for two trajectories

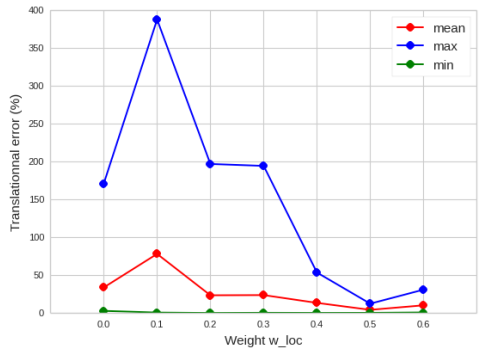

Fig. 8. Mean, maximum and minimum of the deviation between experimental length and the ground truth for each weight w_{loc}

to be a good value as the deviation on the travelled distance is small for all the trajectories.

5. CONCLUSION

In this paper, we have presented a navigation algorithm for robotic platforms which considers the future scene quality on vision-based localisation. It is based on an original criterion to estimate the future localisation quality. This criterion is computed from the 3D points given by a visual odometry algorithm and accounts for the uncertainties that occur in the process. We have integrated this criterion into a control loop designed to perform navigation. The experimental results show that the platform can detect and avoid the textureless areas and improve its localisation performance. Future work will consider more complex missions where obstacle avoidance is involved.

6. ACKNOWLEDGEMENT

The authors would like to thank Maxime Derome (Derome et al. (2014)) for his help with the formulas on uncertainties.

REFERENCES

Bachrach, A. et al. (2012). Estimation, planning, and mapping for autonomous flight using an RGB-D camera in GPS-denied environments. *The International Journal of Robotics Research*, 31(11), 1320–1343.

Bourgaul, F. et al. (2002). Information based adaptive robotic exploration. In *Proceedings of the IEEE/RSJ International Conference on Robots and Systems, Lausanne, Switzerland*, volume 1, 540–545.

Derome, M. et al. (2014). Real-time mobile object detection using stereo. 1021–1026.

Dunn, E. et al. (2009). Developing visual sensing strategies through next best view planning. In *Proceedings of the IEEE/RSJ International Conference on Robots and Systems, St Louis, MO, USA*, 4001–4008.

Forster, C. et al. (2014). Appearance-based active, monocular, dense reconstruction for micro aerial vehicle. In *2014 Robotics: Science and Systems Conference*, EPFL-CONF-203672.

Harris, C. and Stephens, M. (1988). A combined corner and edge detector. In *In Proc. of Fourth Alvey Vision Conference*, 147–151.

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.

Makarenko, A. and et al. (2002). An experiment in integrated exploration. In *Proceedings of the IEEE/RSJ International Conference on Robots and Systems, Lausanne, Switzerland*, volume 1, 534–539.

Mostegel, C. et al. (2014). Active monocular localization: Towards autonomous monocular exploration for multirotor mavs. In *Proceedings of the IEEE International Conference on Robotics and Automation, Hong Kong, China*, 3848–3855.

Roggeman, H. et al. (2014). Embedded vision-based localization and model predictive control for autonomous exploration. In *IROS Workshop on Visual Control of Mobile Robots (ViCoMoR)*, 13–20. Chicago, United States.

Sadat, S.A. et al. (2014). Feature-rich path planning for robust navigation of mavs with mono-slam. In *Proceedings of the IEEE International Conference on Robotics and Automation, Hong Kong, China*, 3870–3875.

Sanfourche, M. et al. (2013). evo: A realtime embedded stereo odometry for mav applications. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems, Tokyo, Japan*, 2107–2114.

Sim, R. and Roy, N. (2005). Global a-optimal robot exploration in slam. In *Proceedings of the IEEE International Conference on Robotics and Automation, Barcelona, Spain*, 661–666.

Vidal-Calleja, T. et al. (2006). Active control for single camera slam. In *Proceedings of the IEEE International Conference on Robotics and Automation, Orlando, FL, USA*, 1930–1936.