Vision and Inertial Sensor Fusion for 3D Self-Localization in Unknown Environment

Céline Teulièrè(1), Lucile Martin(1), Christophe Leroux(1), Pr. Edwige Pissaloux(2)

(1) CEA-LIST DTSI/SRI/LTC
18 route du panorama BP 6
92265 Fontenay-aux-roses Cedex
France
Email: celine.teuliere@cea.fr, lucile.martin@cea.fr, christophe.leroux@cea.fr

(2) Université Paris 6, Laboratoire de Robotique de Paris, CNRS/FRE 2507,
18 route du panorama BP 61
92 256 Fontenay-aux-Roses
France
Email: pissaloux@robot.jussieu.fr

INTRODUCTION

Estimating the ego-motion of an autonomous system is required in many important applications such as navigation, surveillance, etc. In visual trajectory tracking the relative motion between objects in the scene and cameras is determined through the apparent movement of the objects in a sequence of images. Feature based approach recognize objects (features/landmarks) and extract their positions in successive frames. Inertial rate sensors are a good complement to vision since they are very robust, requiring no external measurements. They provide motion information, which is difficult to extract from vision; and they provide information for all six degrees of freedom using a single sensor package. Inertial rate sensors are classically composed of accelerometers, which measure specific force (linear acceleration plus the apparent acceleration due to gravity) and rate gyros, which measure angular velocity.

On the one hand, the inertial sensors have large measurement uncertainty at slow motion and lower relative uncertainty at high velocities. Inertial sensors can measure very high velocities and accelerations. On the other hand, the cameras can track features very accurately at low velocities. With increasing velocity tracking is less accurate since the resolution must be reduced to obtain a larger tracking window with same pixel size and, hence, a higher tracking velocity.

Data fusion may be integrated at different levels of a self-localization application. It might be used to track features (objects) in successive frames [5, 7, 12, 15] and it may break in for motion tracking application [1, 2, 4]. Peter Corke [5] uses data fusion to reduce the search window for a correlation tracking method applied to trajectory tracking. The motion estimation of the robot in the previous image frame initialises the first guess for the feature point to-be-tracked in the actual image, and the estimation of this feature point is updated using the correlation similarity measure. Inertial measurements are then used to refine this estimation and to reduce the size of the correlation search window.

S. Graovac [7] uses a similar method combined with the use of landmarks. Data fusion is done through 3 steps: estimation of the robot’s rates using only inertial measurements, from this estimation the positions of the landmarks in the next image are predicted, data fusion is then used to get a more accurate estimation of the positions of the landmarks in the image.

Finally, inertial data measurements can be used in a predictive state of stochastic filters like particle filters as described in [1, 2, 4].

This paper presents a vision-based application for 3D localization of a mobile robot in natural indoor environments. The vision system includes a stereo head that can be installed on a robot. An inertial measurement unit is added to this system in order to achieve a better accuracy of the 3D self-localization application. The proposed system is evaluated in terms of effectiveness, robustness and computational speed. Evaluation was performed on several pairs of real stereo images of natural scenes and the corresponding recorded inertial data taken onboard an unmanned aerial vehicle (UAV). Inertial rate sensors are composed of accelerometers and rate gyros. Inertial rate sensors provide motion information, which is difficult to extract from vision; and they provide information for all six degrees-of-freedom.

Subsequent sections outline the process of depth recovery used in the presented application for a stereo pair of images (section 2), an overview of the feature point tracking strategies that have been implemented and tested (section 3 part 1), the theoretical description of the data fusion based tracking method that was elaborated for our application (section 3 part 2), a comparative study of the presented methods (section 4), and a conclusion and a presentation of our future work (section 5).
**3D WORLD RECONSTRUCTION**

This process consists of 4 steps: camera calibration, feature extraction, feature matching, and depth reconstruction.

**Camera calibration**

The camera model consists of intrinsic and extrinsic parameters which are estimated in the initial calibration phase of the whole application. This camera model transforms real-world coordinates in image coordinates. In this application, we are using a pinhole camera model [8] as described below and the camera calibration is done using Zhang’s method [17].

*Intrinsic parameters estimation.*

The stereo rig calibration gives the intrinsic parameters (1):

\[
\begin{bmatrix}
\alpha_x & 0 & u_0 & 0 \\
0 & \alpha_y & v_0 & 0 \\
0 & 0 & 1 & 0
\end{bmatrix}
\]

where: \( \alpha_x = -k_x f \), \( \alpha_y = k_y f \) are the scale factors (in pixels), \( f \) is the focal length, and \( v_0, u_0 \) are the principal point coordinates (in pixels) in the image frame.

The matrix \( I_c \) parameters express the transformation between the image frame and the camera frame (Figure 1). The same transformation, in homologous coordinates takes the form of expression (2):

\[
\begin{bmatrix}
s u \\
s v \\
s
\end{bmatrix} = I_c \begin{bmatrix}
x \\
y \\
z
\end{bmatrix}
\]

where \( \begin{bmatrix} u \ v \end{bmatrix} \) are the 2D image point coordinates (in pixels) of the 3D point \( \begin{bmatrix} x \ y \ z \end{bmatrix} \), both in the camera frame.

*Extrinsic parameters estimation.*

The extrinsic parameters of the stereo rig can be estimated from a couple of images. As the orientation of a pair of images of the calibration pattern is known, the translation and rotation expressing the camera frame position with respect to the pattern frame have to be determined with the equation (3)

\[
A = \begin{bmatrix}
R \\
0 & 1
\end{bmatrix}
\]

where \( \begin{bmatrix} t_x \ t_y \ t_z \end{bmatrix} \) is the translation vector between these two frames, and \( R \) is the rotation.

Therefore, if \( \begin{bmatrix} X \ Y \ Z \end{bmatrix} \) are the coordinates of a pattern point in 3D real world its corresponding coordinates in the camera frame are given by equation (4).

\[
\begin{bmatrix}
x \\
y \\
z
\end{bmatrix} = \begin{bmatrix}
r_{11} & r_{12} & r_{13} & t_x \\
r_{21} & r_{22} & r_{23} & t_y \\
r_{31} & r_{32} & r_{33} & t_z
\end{bmatrix} \begin{bmatrix}
X \\
Y \\
Z
\end{bmatrix}
\]

**Feature points extraction**

Processing all the pixels of an image is time consuming. Therefore the application only deals with features with high information content: Harris and Stephen’s corners [9].

**Feature point matching**

Matching interest points is the process of identifying the 2D image points corresponding to a same 3D scene point in a pair of stereo images representing that scene. Therefore, the performance of a visual self-localization application depends on the matching process accuracy and reliability.

Much work on matching points’ methods has been done, starting from simple correlation methods [3] up to more sophisticated method such as the MLESAC iterative process [16].

In the present application, we are using KLT feature point tracker [15] to find pairs of homologous points between the left and right images. As mismatches remain, this tracker is coupled with a filter based on the orientation of the rig (see [11] for more details).
3D points (depth) reconstruction.

Equation (5) links the 3D coordinates of a point with its 2D projection in the camera reference frame:

\[
\begin{pmatrix}
    su \\
    sv \\
    s
\end{pmatrix}
= 
\begin{pmatrix}
    X \\
    Y \\
    Z \\
    1
\end{pmatrix}
\]

where \( M = I \cdot A \).

The corners’ estimated 3D coordinates in a rectified stereo rig involve the rig’s cameras’ parameters (obtained during the calibration) and the image coordinates of the corners in each image of the scene. The depth can be computed (triangulation) only if all image points coordinates (from right and left images) are expressed in the same frame.

Let \( A_s \) be left-to-right image transfer matrix; \( A_s \) can be computed from equation (6).

\[
A_s = A' \cdot A^{-1}
\]

Consequently, left image points’ coordinates in the right camera frame are expressed by the equation (7):

\[
\begin{pmatrix}
    X' \\
    Y' \\
    Z' \\
    1
\end{pmatrix} = 
\begin{pmatrix}
    X \\
    Y \\
    Z \\
    1
\end{pmatrix}
\]

with \( A' \), the right camera’s extrinsic and intrinsic parameters matrices and \( A \), the left camera’s ones.

**FEATURE TRACKING**

This step consists in tracking the projection of preliminary features of a scene in sequences of images. Considering a self-localization application, it allows estimating the trajectory and motion of a robot by tracking and finding relative changes in the position of features.

Classical tracking methods using only vision information have been implemented: i.e. correlation similarity measures [3], the original KLT [15], or the OPENCV’s [12] pyramidal KLT, as explained in the “feature tracking using vision” paragraph.

The fusion of inertial data to the stereovision information as detailed in the “tracking using data fusion” paragraph can fasten and improve the reliability of the tracking process.

Quantitative and qualitative comparisons of these methods are presented in the “experimental results” section.

**Feature tracking using vision**

Before testing filtering and predictive method, we first implemented a method tracking points in a window centred on the coordinates of the point to-be-tracked using a correlation similarity measure.

For each feature point \( X_i \) from the instant time \( k \) image (\( I_k \)), the window centred on \( X_i \) coordinates in next image (\( I_{k+1} \)) is covered. For each pixel of this window, a correlation similarity measure with \( X_i \) is computed. The pixel presenting the highest correlation score is kept as homologous of \( X_i \).
Three different correlation criteria have been tested:
- The Sum of Square Differences (SSD)
  \[ SSD(s,s') = \sum_{p \in N} [I(s+p) - I'(s'+p)]^2 \]  
- The Zero mean Normalized Cross Correlation (ZNCC)
  \[ ZNCC(s,s') = \frac{\sum_{p \in N} [I(s+p) - I(s)] [I'(s'+p) - I'(s)]}{\sqrt{\sum_{p \in N} [I(s+p) - I(s)]^2 [I'(s'+p) - I'(s)]^2}} \]  
- And the Cross correlation described by Saeedi in [13]
  \[ C(s,s') = \frac{\sum_{p \in N} [I(s+p) - I(s)] [I'(s'+p) - I'(s)]^2}{\sqrt{\sum_{p \in N} [I(s+p) - I(s)]^2 [I'(s'+p) - I'(s)]^2}} \]  

The second visual tracking method that was implemented was a multi-scale approach of the KLT feature point tracker [12, 15]. This algorithm aims to find the optimized displacement regarding SSD correlation similarity measure, i.e. minimizing \( \varepsilon \):  
\[ \varepsilon = \sum_{w} [I_{w+1}(s+d) - I_{w}(s)]^2 \]  

The multi-scale approach consists in constructing representations of \( I_{k+1} \) and \( I_k \) at lower resolutions such as: 1/2, 1/4, 1/8, 1/16, called levels 1, 2, 3 and 4; level 0 corresponding to the original image. \( d \) is found for the level 4 first (denoted \( d^4 \)). Then the result is propagated to the next level as first guess for \( d_{est}^3 \). This estimation is used to compute \( d^3 \) giving then a first guess for \( d_{est}^2 \), and so on to the original image. The multi-scale approach allows to deal with reasonable (~10cm) shifting between \( I_{k+1} \) and \( I_k \) and reduces consequently computing time. For more details on this method please refer to [12, 15].

In order to improve the performances of the tracking application in terms of partial occlusions and shifting limits, we developed a tracking algorithm using accelerometers measurements and keeping information from the previous images to ensure a coherent motion of the camera over time. This algorithm is described in the following section.

**Feature tracking using data fusion**

This section describes how to predict the coordinates of the feature point to-be-tracked fusing inertial sensors rate measurement and time corresponding image information. A recursive state estimator is required to handle the fusion of the relative position information from vision with the rate information from the inertial sensors. In this sensor fusion algorithm, the measurements from the accelerometers have to be integrated twice to compute velocity and position.

Figures 3 left and 3 right illustrate the frames that are used for the images and the inertial measurement unit.

![Image frame](image.png)

**Fig.3:** Left: Image plane frame. Right: Inertial measurement unit frame. (u,v) and (y,z) are supposed collinear planes.

Let’s define:
- \( f \): Focal length of the camera
- \( d_k \): Real depth (on the x axis) of the real 3D points with respect to the stereo rig at grabbing instant \( k \).
Relations between image frame and inertial sensors frame are given by (Fig. 3, 4):

\[ \Delta t = \frac{f_y u x}{|x| + \Delta x}, \quad \Delta v = \frac{-f_z v x}{|x| + \Delta x} \]  

(12)

Where \( d_{k+1} = |x_k| + \Delta x \) is estimated from stereo-vision properties.

For each feature point there are 3 steps to consider:

- After computing the motion between the image taken at k-1 instant and the one taken at k instant \( d_k \) is kept in memory. As: \( \Delta x_k = x_k - x_{k-1} = d_k - d_{k-1} \), we can express the rate as follows:
  \[ \dot{x}_k = \frac{x_k - x_{k-1}}{T_e} \]  
  (13)

- We are looking for \((u_{k+1}, v_{k+1})\) coordinates in image k+1 of the feature point corresponding to \((u_k, v_k)\) coordinates in image k. Previous rates are computed as in (14):
  \[ \dot{y}_k = \frac{y_k - y_{k-1}}{T_e} = \frac{1}{fT_e} \left( d_k \Delta u_k + u_k \Delta v_k \right) \cdot \dot{z}_k = \frac{z_k - z_{k-1}}{T_e} = \frac{-1}{fT_e} \left( d_k \Delta v_k + v_k \Delta u_k \right) \]  
  (14)

\( \bar{x}_{k+1}, \bar{y}_{k+1}, \bar{z}_{k+1} \) are measured from the inertial measurement unit. It is then possible to estimate the 3D translation of the camera between images k and k+1:

\[ \Delta x_{k+1} = \left( \dot{x}_k + \bar{x}_{k+1} T_e \right) T_e, \quad \Delta y_{k+1} = \left( \dot{y}_k + \bar{y}_{k+1} T_e \right) T_e, \quad \Delta z_{k+1} = \left( \dot{z}_k + \bar{z}_{k+1} T_e \right) T_e \]  

(15)

As measurement from the accelerometers can be done at a higher frequency than the image grabbing, a mean of these accelerations measurements is done on the lap of time \( T_e \).

From (12) and (15), the new coordinates of the point to be tracked in the image k+1 are then:

\[ u_{k+1} = u_k + \frac{f \Delta y_{k+1} - u_k \Delta v_{k+1}}{d_k + \Delta x_{k+1}}, \quad v_{k+1} = v_k - \frac{-f \Delta z_{k+1} + v_k \Delta y_{k+1}}{d_k + \Delta x_{k+1}} \]  

(16)

- Using a correlation similarity measure around the point defined by (16), the new coordinates of \((u_k, v_k)\) in image k are found. \( d_{k+1}, \Delta x_{k+1} = d_{k+1} - d_k \), \( \Delta u_{k+1}, \Delta v_{k+1} \) are recomputed. The coordinates of the point \((u_k, v_k)\) in the next images can be estimated by iterating the whole process.

**EXPERIMENTAL RESULTS**

To evaluate the performance of a tracking algorithm, 4 criteria should be considered: computing time, detection accuracy, percentage of lost points, capacity to deal with occlusions and noise. Unfortunately these criteria can be difficult to quantify independently from the characteristics of the testing images. For instance, texture, depth of the objects with respect to cameras, velocity and stability of the camera’s motion have a strong impact on the quantity of feature points to-be-tracked and the quality and accuracy of the tracking results.

Even though several types of images have been used to perform our tests, the evaluation of the accuracy of the tracking methods is done qualitatively more than quantitatively. In fact, quantitative results vary too much with texture, depth of the objects, lighting conditions etc… and an average value would not be pertinent.

**Correlation based tracking.**

Three correlation similarity measures have been tested:
- Sum of Square Differences (SSD)
- Zero mean Normalized Cross Correlation (ZNCC)
- Cross Correlation as defined by Saeedi’s in [13]

The results obtained with this method, whatever the correlation is, are not satisfying (Fig 5, 6).

Figure 5 shows that the curved tube located in the front of the image is lost since the second image. Moreover the computing time of this method is not compatible with real time application. The SSD needs an average time of computation per point of 37.7ms/pt, the ZNCC: 116ms/pt and the CC: 117ms/pt. CC and ZNCC still produce numerous tracking errors (Fig 6), even if they give globally better points than the SSD (Fig 5), but the complexity of the calculus implies they are highly time consuming.

![Fig. 5: Feature points tracked over 4 successive frames using SSD correlation similarity measure.](image)

![Fig. 6: Left image: Shifting between images 1 & 2 from fig. 5 using SSD. Centred image: Tracking using ZNCC correlation similarity measure. Right image: Tracking using Saeedi’s correlation similarity measure [13]. Numerous features are mis-tracked in both cases.](image)

KLT tracker tests and results.

Fig. 7 left shows results obtained with KLT method from the same images than in the case of the correlation. Fig. 7 right gives the example of a forward translation. This tracking method gives better results than the correlation based tracking. Computing time is significantly reduced: 0.128s in average (0.25ms/pt).

![Fig. 7: Multi-scale KLT feature point tracking. Left image: vertical translation. Right image: forward translation.](image)

The KLT tracker obviously gives better results than the methods based on a correlation similarity measure. Nevertheless, tracking errors remain, and this algorithm does not deal with partial occlusions and can’t bear translations of the camera of more than 10 cm between images. The velocity of the robot is limited in such a way that there is some overlap between each consecutive frame.

Data fusion based tracker tests and results.

Fig 8 Left presents results obtained for images acquired at constant velocity rate. The variance of the errors of the obtained positions (red) is closer to zero than the one of the previous positions (blue). Such results tend to prove better results could be achieved using a smaller search window. Moreover, there is no constraint on the movement of the camera between to images contrary to KLT tracker, and this algorithm can deal with partial occlusions.
Yet, other series of images tend to shade these results. More specifically, images corresponding to non-zero acceleration give less reliable and inaccurate results (Fig. 8 right). In the particular test of Fig. 8 right, the obtained positions (red) are not better than the previous positions (blue). The mean inaccuracy is close to the window size previously used which means the data fusion here is not more performing than the correlation similarity measure based tracking. These unsatisfying results are certainly due to the accelerometers. Unfortunately, inertial rate sensors are subject to time varying biases and measurement noise. These errors result in growing drift errors in the integrated camera motion. This is especially important for low-cost inertial rate sensors, which are subject to significant drift errors (Fig. 9).

**CONCLUSION AND PERSPECTIVES**

We introduced an application using vision and inertial measurements to self-localize a UAV while moving in its environment. The key issue of this application is the feature point tracking. Even though much work on tracking points has been done, it is still complex to implement in a real application suffering constraints of time and equipment. This paper presented several approaches starting from a simple correlation similarity measure up to a multi-scale approach of the KLT tracker [15]. This last method gives interesting results in terms of computing time and accuracy. Unfortunately, it does not bear shifting of more than 10 cm between images and it loses a consistent quantity of points over a long lap of time. For this reason, we have chosen to try fusing information extracted from images with information obtained from the inertial measurement unit existing onboard the UAV. Using accelerometers to predict the position of a feature point in the next image as described in this paper gives results equivalent to the multi-scale approach due to the noise and biases of the accelerometers. One should notice that using data fusion to track feature points allows dealing with occlusions and shifting of more than 10 cm. For this reason, this approach can be considered more efficient than the KLT tracker, the main constraint being the need of high quality accelerometers.

The next issue to be treated is the one of the trajectory tracking. The application described above allows us to track 3D points over time and estimate their position (depth) with respect to the UAV’s frame. In order to track the UAV’s trajectory over time accurately, its orientation between two instants still has to be estimated (Fig. 10).
Once again, a MLESAC [16] approach could be used but it is time consuming and errors in the estimation of the fundamental matrices can occur. For this reason, a particle filter is being implemented fusing accelerometers and gyro-meters measurements with images information.

ACKNOWLEDGEMENT

We are most grateful for the support from the DGA. The support of the CEA-LIST is gratefully acknowledged especially M. Laurent Eck and M. Yvan Measson.

REFERENCES