Combining complementary edge, point and color cues in model-based tracking for highly dynamic scenes

Antoine Petit, Eric Marchand, Keyvan Kanani

Combining complementary edge, point and color cues in model-based tracking for highly dynamic scenes

Antoine Petit, Eric Marchand, Keyvan Kanani

***Abstract*— This paper focuses on the issue of estimating the complete 3D pose of the camera with respect to a complex object, in a potentially highly dynamic scene, through model-based tracking. We propose to robustly combine complementary geometrical edge and point features with color based features in the minimization process. A Kalman filtering and pose prediction process is also suggested to handle potential large inter-frame motions. In order to deal with complex 3D models, our method takes advantage of hardware acceleration. Promising results, outperforming classical state-of-art approaches, have been obtained on various real and synthetic image sequences, with a focus on space robotics applications.**

I. Introduction

Determining the complete 3D pose of the camera with respect to the object is a key requirement in many robotic applications involving 3D objects, for instance in the case of autonomous, vision-based and uncooperative space rendezvous with space targets or debris [13]. Based on the knowledge of the 3D model of the target, common approaches address this problem by using either texture [1], edge features [6, 5, 13] or color or intensity features [11, 17]. In order to cope with advantages and drawbacks of these different types of cues, some researches have focused on combining them. Different ways of handling the integration have been considered.

Some studies propose a sequential integration of edge and texture cues [10, 3], where the computed dominant motion [10] provides a prediction of the projected edges in the image, improving the initialization of an edge-based minimization process. [3] uses optical flow of pixels lying on the projected object to initialize a maximization process of the separation between object and background along the object contours using color or luminance probabilities.

Other methods simultaneously merge features of different types within probabilistic or filtering techniques such as Kalman filters w.r.t. the pose parameters [8, 7], by integrating interest points matching in an Extended Kalman Filter for [8], and integrating optical flow estimation for [7]. [18] elaborated a similar hybrid solution within a probabilistic Expectation Maximization (EM) framework which aims at optimizing the posterior probabilities of both edge and FAST point features.

Works of [21, 15, 16, 12] rely on a deterministic iterative minimization of a global error function combining the different features. [21] proposes for instance to integrate in a classical edge-based error function [6, 5] geometrical distances between keypoint features detected and matched in two consecutive frames and the projection of their 3D positions. [16] instead relies on optical flow estimation of some regularly spread pixels lying on planar patches on the object, providing point correspondences between successive images and thus point to point distances to be minimized w.r.t. the camera displacement through a homography. In contrast to [21, 16], [15] proposes a texture-based error function which is directly based on the difference between intensities of some pixels selected on reference planar patches of the object. [12, 14] combine in the objective function to be optimized, w.r.t. the camera pose, a classical geometrical information provided by the distances between model and image edges with color information through object/background color separation along the model edges.

Following ideas suggested by [12, 14], and by [21, 20], which propose hybrid methods incorporating edge and keypoint features, we propose to integrate within a deterministic pose estimation process the different considered visual features, in order to take advantage of their complementarity and to overcome the limitations of classical edge-based approaches.

A low-level multiple hypotheses edge matching process is also embedded in our framework. Like in our previous works [14], the model projection and edge generation phase relies on the graphics process units (GPU) in order to handle complex 3D models, and to be reasonably time-consuming.

Since the application context of this work mainly deals with space robotics applications (space rendezvous and proximity operations) for which the targeted object generally involves constant velocity motions, a Kalman filtering framework on the pose parameters is designed, enabling a prediction step for the pose, providing a finer initialization, intended to cope with potential large inter-frame motions and to avoid local minima.

II. Combining keypoint, edge and color features in a model-based tracking framework

Our problem is restricted to model-based tracking, using a 3D model of the target. The goal is to estimate the camera pose $\mathbf{r}$ by minimizing, with respect to $\mathbf{r}$, the function Δ accounting for errors $e_i(\mathbf{r})$ between a set of visual features extracted from the image and the forward projection of their 3D homologues in the image plane:

$$\Delta(\mathbf{r}) = \sum_i \rho(e_i(\mathbf{r})) \tag{1}$$

where ρ is a robust estimator, which reduces the sensitivity to outliers. This is a non-linear minimization problem with

A. Petit is with INRIA Rennes - Bretagne Atlantique, Lagadic Team, France, Antoine.Guillaume.Petit@inria.fr. E. Marchand is with Université de Rennes 1, IRISA, Lagadic Team, France, marchand@irisa.fr. K. Kanani is with Astrium, Toulouse, France

respect to the pose parameters $\mathbf{r}$, and we follow a Gauss-Newton minimization framework to handle it, by iteratively updating the pose $\mathbf{r}$:

$$\mathbf{r}_{k+1} = \mathbf{r}_k \oplus \delta\mathbf{r} \quad (2)$$

$$\delta\mathbf{r} = -\lambda(\mathbf{DJ})^{+}\mathbf{De} \quad (3)$$

where $(\mathbf{DJ})^{+}$ is the pseudo inverse of $\mathbf{DJ}$, with $\mathbf{J}$ the Jacobian matrix of the error vector $\mathbf{e}(\mathbf{r})$. λ is a proportional gain and $\mathbf{D}$ is a weighting matrix associated to the Tukey robust estimator. With the homogeneous matrix $\mathbf{M}_{k+1}$, the new pose $\mathbf{r}_{k+1}$ can be updated using the exponential map [9]:

$$\mathbf{M}_{k+1} = exp([\delta\mathbf{r}])\ \mathbf{M}_k \quad (4)$$

Our challenge is to integrate, within this framework, geometrical edge-based features with color features along silhouette edges and geometrical features based on Harris corner points extracted in the image, tracked through the KLT algorithm [19]. The objective is to fuse in the criterion Δ to be optimized a geometrical information provided by distances between edges, with distances between keypoints and with a denser color information through object/background color separation along the silhouette model edges. The goal is to benefit from the complementarity of these features and to overcome the limitations of classical single cue approaches.

Geometrical features based on edges or keypoints such as Harris corners rely on line-to-point or point-to-point correspondences and on geometrical distances accounting for these correspondences. On one hand, edges have the advantage of being robust to illumination conditions but suffer from having similar appearances, resulting in ambiguities between different edges and potential local minima. On the other hand, point features can be described more specifically, imposing a better spatio-temporal constraint. Though begin locally performed, the KLT algorithm allows a larger convergence radius. However, these keypoints are more sensitive to illumination conditions, for both extraction and tracking steps. Besides, pose errors resulting from the tracking phase at the previous frame are integrated in the back-projection process of the keypoints, leading to potential *drift* problems across the sequence. Whereas for edges, the edge matching process in the image instead rely on the absolute reference of the projection of the 3D model.

With color or intensity edge-based feature, the idea is to avoid any image extraction or segmentation that could lead to outliers and mismatches. By processing a dense information along the silhouette of the projected 3D model by modeling the color (or luminance) appearance on both sides of the edges, using simple statistics, and optimizing their separation, a better accuracy can be achieved. A main advantage is a better robustness to image or motion blur, background clutter or noise. However, among drawbacks these features need color contrast to perform efficiently and are limited by their computational costs.

By combining these features, Δ can be rewritten as:

$$\Delta = w^g\Delta^g + w^c\Delta^c + w^p\Delta^p \quad (5)$$

Δ^g refers to the geometrical edge-based error function, Δ^c stands for the color-based one and Δ^p corresponds to the keypoint features. w^g, w^c and w^p are the respective weighting parameters. Let us note that the involved visual features rely on the projection of the 3D model.

III. Visual features

A. Geometrical edge features

1) Model projection and generation of model edge points: As in our previous works [13, 14], we propose to automatically manage the projection of the model and to determine the visible and prominent edges from the rendered scene, by considering the direct use of a complete model, which can be textured or not. By using the graphics process units (GPU) and a 3D rendering engine (OpenSceneGraph here), we avoid any manual pre-processing.

For each acquired image $\mathbf{I}_{k+1}$, the model is rendered with respect to the previous pose $\mathbf{r}_k$. Processing the depth map, we can obtain a set of N_g 2D control points $\{\mathbf{x}_i\}_{i=1}^{N_g}$ which belong to target rims, edges and visible textures from the rendered scene. The corresponding 3D points $\mathbf{X}_i$ can then easily be reconstructed.

2) Error computation and Jacobian matrix: the edge-based function Δ^g is computed in a similar way to [14]. From the model edge points we perform a 1D search along the normal of the underlying edge of each $\mathbf{x}_i(\mathbf{r}_k)$. A common approach is to choose on the scan line the pixel with the maximum gradient as the matching edge point $\mathbf{x}'_i$ in the new image. The considered approach is based on the *ECM* algorithm [2].

Once correspondences between the set of control points $\{\mathbf{x}_i\}_{i=1}^{N_g}$ and the set of image edge points $\{\mathbf{x}'_i\}_{i=1}^{N_g}$ are established, our approach considers the distance between the projected 3D line $l_i(\mathbf{r})$ underlying the projected control point $\mathbf{x}_i(\mathbf{r})$ (projected from the 3D point $\mathbf{X}_i$) and the selected matching point $\mathbf{x}'_i$ in the image. The error function $\Delta^g(\mathbf{r})$ to be minimized with respect to the pose $\mathbf{r}$ can be written as:

$$\Delta^g(\mathbf{r}) = \frac{1}{N_g}\sum_i \rho^g(e_i^g(\mathbf{r})) \quad (6)$$

where $e_i^g(\mathbf{r}) = \sigma_g^{-1} d_\perp(l_i(\mathbf{r}), \mathbf{x}'_i))$ and $d_\perp(l_i(\mathbf{r}), \mathbf{x}'_i))$ is the distance between the point $\mathbf{x}'_i$ and the corresponding line $l_i(\mathbf{r})$. ρ^g is a Tukey robust estimator and σ_g is a normalization factor, the estimator of the standard deviation of the error e_i^g. $\widehat{\sigma_g}^2 = \frac{1}{N_g}\sum_i \rho^g(d_\perp(l_i(\mathbf{r}_f), \mathbf{x}'_i))$ with $\mathbf{r}_f$ the pose computed at the end of the previous minimization process.

The criterion Δ^g improves other approaches [22, 12, 4] which consider the distance between $\mathbf{x}_i(\mathbf{r})$ and $\mathbf{x}'_i$ along the 2D normal vector $\mathbf{n}_i$ to the edge underlying $\mathbf{x}_i(\mathbf{r})$, but neglecting the dependence of $\mathbf{n}_i$ w.r.t. to the pose $\mathbf{r}$, what is taking into in our computation of the Jacobian matrix of e_i^g.

A key requirement is to compute the 3D equation in the scene or object frame of the line L_i such that $l_i(\mathbf{r}) = pr(L_i, \mathbf{r})$. This is necessary in order to compute the error e_i^g and the corresponding Jacobian matrix $\mathbf{J}_{e_i^g}$. For more details and the complete computation of $\mathbf{J}_{e_i^g}$, see [14, 5].

As in [13], multiple hypotheses can be considered in the process. These hypotheses correspond to potential edges, which are different local extrema $\mathbf{x}'_{i,j}$ of the gradient along the scan line. And Δ^g becomes:

$$\Delta^g = \sum_i \frac{1}{N_g} \rho^g(\sigma_g^{-1} \min_j d_\perp(l_i(\mathbf{r}), \mathbf{x}'_{i,j})) \quad (7)$$

where points $\mathbf{x}'_{i,j}$ are the selected candidates for each control point $\mathbf{x}_i$. For σ_g, we use the estimate: $\widehat{\sigma_g}^2 = \frac{1}{N_g}\sum_i \rho^g(\min_j d_\perp(l_i(\mathbf{r}), \mathbf{x}'_{i,j}))$.

B. Color features

The color-based function Δ^c is elaborated to characterize the separation between both sides of projected model edges belonging to the silhouette, by relying on color information. Δ^c is computed the same way as in [14], and the principle steps are briefly recalled hereafter.

The goal is to compute local color statistics (means and covariances) along the normal to the projected silhouette model edges, on both sides. Then for each pixel along the normal, we determine a residual representing the consistency of the pixel with these statistics, according to a fuzzy membership rule to each side.

More formally, given the set of N_s projected silhouette model edge points $\{\mathbf{x}_i(\mathbf{r})\}_{i=1}^{N_s}$, we compute color statistics, so to say RGB means $\overline{\mathbf{I}}_i^O$ and $\overline{\mathbf{I}}_i^B$ and covariances $\overline{\mathbf{R}}_i^O$ and $\overline{\mathbf{R}}_i^B$, on both side of the edge (object O and background B) using $2D+1$ pixels along the edge normal $\mathbf{n}_i$, up to a distance L.

These statistics are then mixed according to a fuzzy membership rule, giving means $\hat{\mathbf{I}}_{i,j}(\mathbf{r})$ and covariances $\hat{\mathbf{R}}_{i,j}(\mathbf{r})$ for the pixels $\mathbf{y}_{i,j}$ on the normal $\mathbf{n}_i$, wether they are on the object or background side. $\hat{\mathbf{I}}_{i,j}(\mathbf{r})$ can be seen as a desired color value for $\mathbf{y}_{i,j}$. An error $e^c_{i,j}(\mathbf{r})$ can then be defined as:

$$e^c_{i,j}(\mathbf{r}) = \sqrt{(\hat{\mathbf{I}}_{i,j}(\mathbf{r}) - \mathbf{I}(\mathbf{y}_{i,j}))^T \hat{\mathbf{R}}^{-1}_{i,j} (\hat{\mathbf{I}}_{i,j}(\mathbf{r}) - \mathbf{I}(\mathbf{y}_{i,j})))} \quad (8)$$

Using a M-estimator (Tukey) to cope with outliers, Δ^c can be written as:

$$\Delta^c = \frac{1}{N_c} \sum_i \rho^c(\sum_j e^c_{i,j}(\mathbf{r})) \quad (9)$$

with $N_c = 2DN_s$ accounting for the number of color features. For more accuracy and temporal smoothness, we propose to introduce temporal consistency, by integrating the color statistics computed on the previous frame ${}^P\mathbf{I}$ for the silhouette edge points $\mathbf{x}_i(\mathbf{r}_k)$ at the first iteration of the minimization process. With α a weighting factor ($0 < \alpha < 1$), it gives $e^c_{i,j}(\mathbf{r}) = \alpha\hat{\mathbf{I}}_{i,j}(\mathbf{r}) + (1-\alpha)({}^P\hat{\mathbf{I}}_{i,j}(\mathbf{r})) - \mathbf{I}(\mathbf{y}_{i,j})$. For the computation of the corresponding Jacobian matrix $\mathbf{J}_{e^c_{i,j}}$, see [14].

C. Geometrical keypoint based features

Another class of visual features which can be used are keypoints tracked across the image sequence. As previously suggested by [3] or by [21, 16] within there hybrid approaches, the idea is to design a texture-based objective function Δ^p accounting for geometrical distances between keypoints extracted and tracked over successive images. But in contrast to [16], which process 2D-2D point correspondences to estimate the 2D transformation from $\mathbf{I}_k$ to $\mathbf{I}_{k+1}$ of planar local regions underlying the points, we use 2D-3D correspondences to directly minimize Δ^p w.r.t. the pose $\mathbf{r}$.

More specifically, let us denote $\{\mathbf{x}_i\}_{i=1}^{N_p}$ a set of detected interests points in frame $\mathbf{I}_k$. Assuming the pose $\mathbf{r}_k$ has been properly estimated, we can restrict these points to be lying on the projected 3D model with respect to $\mathbf{r}_k$. Since we rely on a complete 3D model, the depth of the points in the scene can be accurately retrieved, and using $\mathbf{r}_k$, we can back-project these points on the 3D model, giving a set $\{\mathbf{X}_i\}_{i=1}^{N_p}$ of 3D points of the 3D model. This is a major difference with respect to [21] which aims at simultaneously optimizing the camera poses and projections of the matched points in two successive frames, relaxing the assumption of having accurate previous pose estimate and 3D model, but increasing computations. Our knowledge of a complete 3D model, along with the use of convenient rendering techniques allows us the keep this assumption valid.

As keypoints, we employ the Harris corners detector inside the silhouette of the projected model in the image to extract the set $\{\mathbf{x}_i\}_{i=1}^{N_p}$ in $\mathbf{I}_k$. Then, the KLT tracking algorithm enables to track this set of points in frame $\mathbf{I}_{k+1}$, resulting in a corresponding set $\{\mathbf{x}'_i\}_{i=1}^{N_p}$.

From the correspondences between $\{\mathbf{X}_i\}_{i=1}^{N_p}$ and $\{\mathbf{x}'_i\}_{i=1}^{N_p}$, Δ^p can be computed as follows:

$$\Delta^p(\mathbf{r}) = \frac{1}{N_p} \sum_i^{N_p} \rho^p(e^p_i) \quad (10)$$

with $e^p_i = {\sigma_p}^{-1}(\mathbf{x}_i(\mathbf{r}) - \mathbf{x}'_i)$ and $\mathbf{x}_i(\mathbf{r}) = pr(\mathbf{X}_i, \mathbf{r}))$. ρ^p is the Tukey robust estimator associated to these errors. σ_p accounts for the standard deviation of errors e^p_i. Similarly to σ_g, we use the estimate $\widehat{\sigma_p}^2 = \frac{1}{N_p}\sum_i \rho^p(\mathbf{x}_i(\mathbf{r}_f) - \mathbf{x}'_i)$. The Jacobian matrix $\mathbf{J}_{e^p_i}$ is computed as follows:

$$\mathbf{J}_{e^p_i} = {\sigma_p}^{-1} \frac{\partial \mathbf{x}_i}{\partial \mathbf{r}} \quad (11)$$

$$= \frac{\mathbf{K}}{\sigma_p} \begin{bmatrix} -1/Z & 0 & x/Z & xy & -(1+x^2) & y \\ 0 & -1/Z & y/Z & (1+y^2) & xy & -x \end{bmatrix}$$

with (x, y) the meter coordinates of the image point $\mathbf{x}_i(\mathbf{r}) = pr(\mathbf{X}_i, \mathbf{r}))$, and Z the depth of the corresponding 3D point. $\mathbf{K}$ is the calibration matrix of the camera.

D. Combination of the visual features

The combination of the three types of features and their respective errors $e^g_i(\mathbf{r})$, $e^c_{i,j}(\mathbf{r})$ and $e^p_i(\mathbf{r})$ in the minimization framework is achieved by stacking the error vectors $\mathbf{e}^g_i$, $\mathbf{e}^c_{i,j}$ and $\mathbf{e}^p_i$ into a global error vector $\mathbf{e}$ and their corresponding Jacobian matrices $\mathbf{J}_{\mathbf{e}^g}$, $\mathbf{J}_{\mathbf{e}^c}$ and $\mathbf{J}_{\mathbf{e}^p}$ into a global Jacobian matrix $\mathbf{J}$:

$$\mathbf{e} = \begin{bmatrix} \sqrt{\lambda^g}\mathbf{e}^{gT} & \sqrt{\lambda^c}\mathbf{e}^{cT} & \sqrt{\lambda^p}\mathbf{e}^{pT} \end{bmatrix}^T \quad (12)$$

$$\mathbf{J} = \begin{bmatrix} \sqrt{\lambda^g}\mathbf{J}_{\mathbf{e}^g}{}^T & \sqrt{\lambda^c}\mathbf{J}_{\mathbf{e}^c}{}^T & \sqrt{\lambda^p}\mathbf{J}_{\mathbf{e}^p}{}^T \end{bmatrix}^T \quad (13)$$

where $\lambda^g = w^g/N_g$, $\lambda^c = w^c/N_c$ and $\lambda^p = w^p/N_p$, $\mathbf{e}$ is a $N_g + N_c + N_p$ vector and $\mathbf{J}$ is a $(N_g + N_c + N_p) \times 6$

matrix. Regarding the weighting matrix $\mathbf{D}$, it is written as $\mathbf{D} = blockdiag(\mathbf{D^g}, \mathbf{D^c}, \mathbf{D}^p)$, where $\mathbf{D}^g$, $\mathbf{D^c}$ and $\mathbf{D^p}$ are the respective weighting matrices associated to the robust estimators ρ^g, ρ^c and ρ^p.

IV. Filtering and pose prediction

A. Pose uncertainty

An important tool to set up is the measurement of the quality and reliability of the tracking process, based on the errors provided by the different cues integrated in the objective function. For this purpose, we can compute the covariance matrix $\mathbf{\Sigma}_{\delta\mathbf{r}}$ on the parameters of the pose error $\delta\mathbf{r}$, which results from the errors $\mathbf{e}$, based on equation (4). We can indeed assume that the pose error $\delta\mathbf{r}$ follows a Gaussian distribution $\delta\mathbf{r} \sim \mathcal{N}(\mathbf{0}, \mathbf{\Sigma}_{\delta\mathbf{r}})$. We also assume that error $\mathbf{e}$ follows a Gaussian distribution $\mathbf{e} \sim \mathcal{N}(\mathbf{0}, \mathbf{\Sigma_e})$, with $\mathbf{\Sigma_e} = blockdiag(\lambda^g \mathbf{I}_{N_g \times N_g}, \lambda^c \mathbf{I}_{N_c \times N_c}, \lambda^p \mathbf{I}_{N_p \times N_p})$, since the errors are normalized. From equation (3) giving the value of $\delta\mathbf{r}$, $\mathbf{\Sigma}_{\delta\mathbf{r}}$ can be written as:

$$\mathbf{\Sigma}_{\delta\mathbf{r}} = E\left[\delta\mathbf{r}\delta\mathbf{r}^T\right] = E\left[(\mathbf{DJ})^+ \mathbf{De}\ \mathbf{e}^T\mathbf{D}^T((\mathbf{DJ})^+)^T\right]$$

Since the uncertainty lies on $\mathbf{e}$ we have:

$$\mathbf{\Sigma}_{\delta\mathbf{r}} = (\mathbf{DJ})^+\mathbf{D}\mathbf{\Sigma_e}\mathbf{D}^T((\mathbf{DJ})^+)^T \tag{14}$$

We can actually determine a covariance matrix for the whole set of the visual features or one for each type, giving three covariance matrices, $\mathbf{\Sigma}^g_{\delta\mathbf{r}}$, $\mathbf{\Sigma}^c_{\delta\mathbf{r}}$, and $\mathbf{\Sigma}^p_{\delta\mathbf{r}}$.

B. Kalman filtering and pose prediction

Kalman filtering: in order to smooth pose estimates, we propose to incorporate a filtering process, relying on the Kalman filtering theory. To achieve this, we employ an linear Kalman filter on the parameters of the camera velocity $\mathbf{v}$, which is integrated to determine the pose so that: $\mathbf{M}_{k+1} = exp([\mathbf{v}_{k+1}\delta t])\ \mathbf{M}_k$, at each time step k. We assume a constant velocity dynamic model. This model is particularly suitable in our context of space rendezvous since constant velocity motions are generally involved. As the state of the system $(\widehat{\mathbf{x}}_k, \mathbf{\Sigma}_k)$ we simply choose the velocity parameters, so that $\mathbf{x}_k = \mathbf{v}_k$ is the actual system state at time step k, with $\widehat{\mathbf{x}}_k = \widehat{\mathbf{v}}_k$ the a posteriori estimate of $\mathbf{x}_k$, and $\mathbf{\Sigma}_k$ the corresponding covariance matrix:

$$\widehat{\mathbf{x}}_k = \mathbf{x}_k + \eta_k \quad \text{and} \quad \widehat{\mathbf{M}}_k = exp([\eta_k\delta t])\mathbf{M}_k \tag{15}$$

with $\eta_k \sim \mathcal{N}(\mathbf{0}, \mathbf{\Sigma}_k)$, and $\widehat{\mathbf{M}}_k$ the posterior estimate of the actual pose $\mathbf{M}_k$. With a constant velocity model, the true state at time step $k+1$ is evolved from the state at k according to:

$$\mathbf{x}_{k+1} = \mathbf{x}_k + \mathbf{n}_{k+1} \tag{16}$$

with $\mathbf{n}_k \sim \mathcal{N}(\mathbf{0}, \mathbf{Q}_k)$ the state noise, and $\mathbf{Q}_k = diagblock(\mathbf{\Sigma}_{\boldsymbol{v}}, \mathbf{\Sigma}_{\boldsymbol{\omega}})$ the state noise covariance matrix, with $\mathbf{\Sigma}_{\boldsymbol{v}} = \sigma^2_{\boldsymbol{v}}\mathbf{I}_{3\times3}$ and $\mathbf{\Sigma}_{\boldsymbol{\omega}} = \sigma^2_{\boldsymbol{\omega}}\mathbf{I}_{3\times3}$, $\sigma_{\boldsymbol{v}}$ and $\sigma_{\boldsymbol{\omega}}$ being state noise standard deviations respectively on the translation and rotation parameters of $\mathbf{v}$. As an observation $\mathbf{z}_{k+1}$, the minimization process described in section II provides us with a pose measure $\mathbf{M}^m_{k+1}$, and with a measure of the velocity $\mathbf{v}^m_{k+1}$:

$$\mathbf{v}^m_{k+1} = exp^{-1}(\widehat{\mathbf{M}}_k\ \mathbf{M}^{m\ -1}_{k+1}) \tag{17}$$

$$\mathbf{z}_{k+1} = \mathbf{v}^m_{k+1} = \mathbf{x}_{k+1} + \mathbf{w}_{k+1} \tag{18}$$

with $\mathbf{w}_k \sim \mathcal{N}(\mathbf{0}, \mathbf{R}_k)$, with $\mathbf{R}_k$ the observation noise covariance matrix. The noise $\mathbf{w}_k$ can actually be interpreted as equal to $\delta\mathbf{r}_k$, which is the pose error at the end of the pose estimation process, so that $\mathbf{R}_k = \mathbf{\Sigma}_{\delta\mathbf{r}}$ (equation 14). The prediction step can then be achieved, giving prior estimate of the future state and the pose:

$$\widehat{\mathbf{x}}_{k+1|k} = \widehat{\mathbf{x}}_k, \mathbf{\Sigma}_{k+1|k} = \mathbf{\Sigma}_k + \mathbf{Q}_{k+1} \tag{19}$$

$$\text{and } \widehat{\mathbf{M}}_{k+1|k} = exp(\left[\widehat{\mathbf{x}}_{k+1|k}\right])\widehat{\mathbf{M}}_k \tag{20}$$

The update step is classically performed, resulting in the Kalman gain $\boldsymbol{K}_{k+1}$ and in posterior state and pose estimates $\widehat{\mathbf{x}}_{k+1}$, $\mathbf{\Sigma}_{k+1}$ and $\widehat{\mathbf{M}}_{k+1}$:

$$\boldsymbol{K}_{k+1} = \mathbf{\Sigma}_{k+1|k}(\mathbf{\Sigma}_{k+1|k} + \mathbf{R}_k)^{-1} \tag{21}$$

$$\widehat{\mathbf{x}}_{k+1} = \widehat{\mathbf{x}}_{k+1|k} + \boldsymbol{K}_{k+1}(\mathbf{z}_{k+1} - \widehat{\mathbf{x}}_{k+1|k}) \tag{22}$$

$$\mathbf{\Sigma}_{k+1} = (\mathbf{I} - \boldsymbol{K}_{k+1})\mathbf{\Sigma}_{k+1|k} \tag{23}$$

$$\widehat{\mathbf{M}}_{k+1} = exp(\left[\boldsymbol{K}_{k+1}(\mathbf{z}_{k+1} - \widehat{\mathbf{x}}_{k+1|k})\right])\widehat{\mathbf{M}}_{k+1|k} \tag{24}$$

Pose prediction: from the estimate pose $\widehat{\mathbf{M}}_{k+1}$, we suggest to use it to provide a predicted pose $\widehat{\mathbf{M}}^p_{k+1}$ which is intended to initialize the pose estimation phase for the next time step. A natural idea is to choose the prior estimate at $k+1$ so that $\widehat{\mathbf{M}}^{pred}_{k+1} = exp([\widehat{\mathbf{x}}_{k+1}])\widehat{\mathbf{M}}_{k+1}$.

This prediction step is particularly useful when large inter-frame motions are observed in the image, since the model projection with respect to $\widehat{\mathbf{M}}^{pred}_{k+1}$ can bring the error function Δ closer to its actual minimum, avoiding local ones.

V. Experimental results

We validate the proposed tracking method, both qualitatively on real images and quantitatively on synthetic images, and its benefits are verified. Let us first classify the different contributions proposed in this paper:

- Efficient model projection and model edge control point generation, and pose estimation based on geometrical edge features: $C0$.
- Integration of a multiple-hypothesis framework in the edge registration process: $C1$.
- Integration of the color-based visual features: $C2$.
- Temporal consistency for the color-based objective function: $C3$.
- Integration of the geometrical keypoint features: $C4$.
- Kalman filtering and pose prediction step: $C5$.

The solution combining $(C0, C1, C2, C3)$ is actually very close to the one proposed in [14] except the incorporation of a multiple-hypotheses scheme taking advantage of line clustering. We refer it as NM (for Nominal Mode). We propose to evaluate the main contributions of this paper ($C4$ and $C5$) and to compare them with (NM) which has proven in [14] to have superior performances with respect to [5, 12, 13].

In terms of implementation, a standard laptop (2.8GHz Intel Core i7 CPU) has used and for all the following tests.

A. Results on synthetic images

We consider the synthetic sequence, proposed in [14], featuring a Spot satellite and which is provided with ground-truth on the pose. The results of the tracking of the target can be seen on Figures 1 and 2 where the errors on the pose parameters are plotted (translation and rotation, represented by Euler angles). For our new solutions $(NM, C4)$ and $(NM, C5)$, and for our previous one (NM), the tracking is properly performed, as depicted on the image sequence on Figure 1 for $(NM, C4, C5)$. In terms of pose errors, though the improvement is quite slight on this sequence, the incorporation of keypoints enables to avoid some peaks on the pose parameters errors which are encountered by (NM), especially when the panels of satellite tends to flip in the image (around frames 750, see second image in Figure 1) or when the target tends to get far, with low luminosity (frames 900-1350). Due to the temporal constraint imposed by the tracking of the keypoints, these points can prevent from some jitter effect on the pose.

By employing the Kalman filter and the pose prediction step, the pose get smoothed. The advantage of the prediction step can be enhanced by temporally down-sampling this sequence, since tracking can be correctly handled with a down-sampling factor f up to $f = 7$ (see provided video), and enables to deal with very large inter-frame motions (up to 20 pixels).

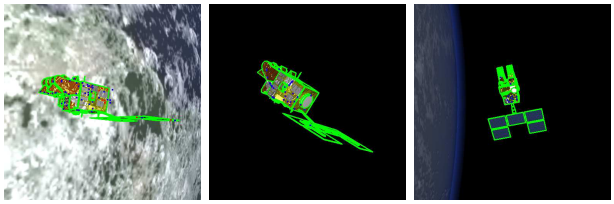


Fig. 1: Tracking for the Spot sequence with the proposed method $(NM, C4, C5)$ for frames 20, 750 and 1440. Tracked Harris corners with the KLT algorithm are represented in with blue dots. Red dots are their reprojection at the end of the pose estimation process.

B. Results on real images

A first sequence shows the Soyuz TMA-03M undocking from the International Space Station (ISS) We have also run the Nominal Mode NM, and the proposed solution $(NM, C4)$. Different temporal down-sampling factors have been tried out. With $f = 1$, we observe that $(NM, C4)$ is able to track the target on a longer sequence than (NM) (Figure 3(e)). With $f = 5$, the advantage of introducing keypoints is obvious since tracking for $(NM, C4)$ is not affected by the down-sampling (Figures 3(a)- 3(d)), in contrast to (NM) which rapidly fails (Figures 3(f)). The ability of keypoints of being properly tracked with large inter-frame motions (up to 25 pixels) is here stressed out. The uncertainty of the pose for both methods is represented on Figure 4 by the global covariance matrix $\mathbf{\Sigma}_{\delta \mathbf{r}}$, showing, for $f = 5$, larger uncertainty when both solutions tends to fail (around frame 20 for NM and around frame 420 for $(NM, C4)$).

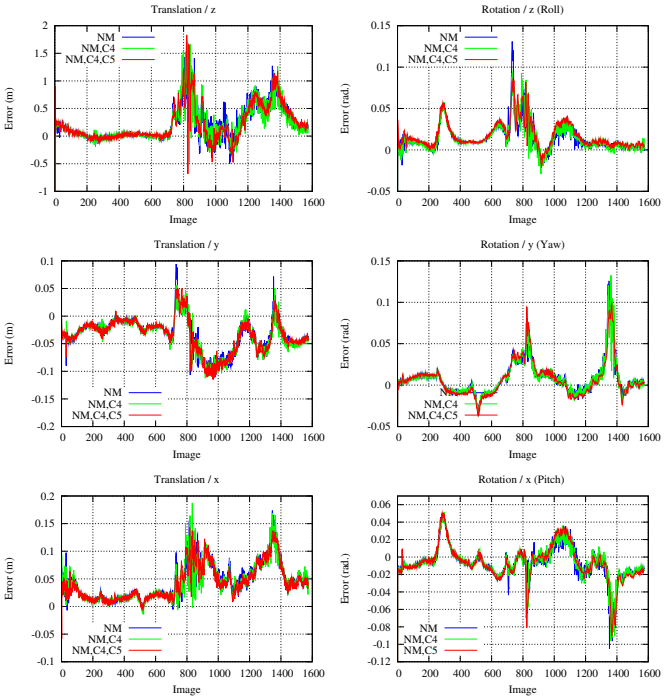
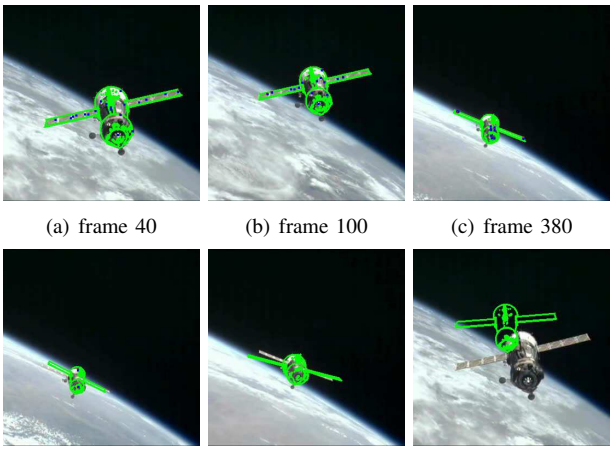


Fig. 2: Errors on the estimated camera pose over the whole sequence, (NM), $(NM, C4)$ and $(NM, C4, C5)$ with $f = 1$.



(a) frame 40 (b) frame 100 (c) frame 380

(d) frame 420 (e) frame 1670 (f) frame 40

Fig. 3: Tracking for the Soyuz sequence using $(NM, C4)$ with $f = 5$ (a-d), using NM with $f = 1$ (e) and $f = 5$ (f).

The second sequence features a figurine of Mario, which is 35cm high, and made of curved and complex shapes. The 3D model was obtained using a Kinect sensor and the ReconstructMe software. Despite rough modelization of some parts, this model, which is made of 15000 vertices, for a 5.5MB size, has been directly used in our tracking algorithm, showing the convenience of the proposed method and the efficiency of the model projection system. With (NM), tracking gets quickly lost (Figure 5(f)) due to the large inter-frame motion, where $(NM, C4)$ achieves it successfully throughout the sequence, even in the case of very large motions in the image (up to 40 pixels) and important motion blur (see Figure 5(a) and provided video).

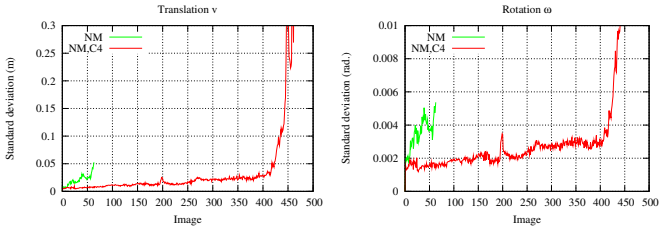
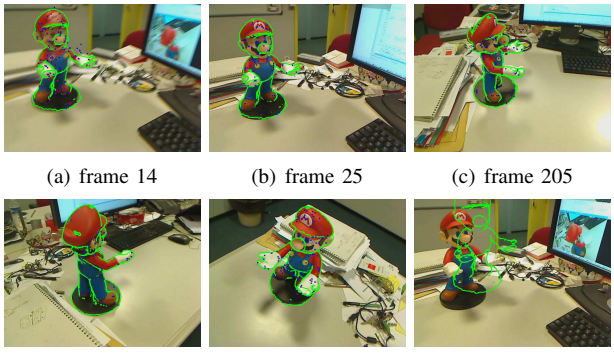


Fig. 4: Covariances on the pose displacement for (NM) and $(NM, C4)$. Square root of traces on translation and rotation parameters of $\mathbf{\Sigma}_{\delta\mathbf{r}}$ are represented.

The uncertainty of the pose for both methods is also represented (Figure 6), for the different visual features, with $\mathbf{\Sigma}^{g}_{\delta\mathbf{r}}$, $\mathbf{\Sigma}^{c}_{\delta\mathbf{r}}$, and $\mathbf{\Sigma}^{p}_{\delta\mathbf{r}}$, and for the whole set with $\mathbf{\Sigma}_{\delta\mathbf{r}}$. We can observe that the effective fail of the tracking for NM around frame 5 does not have much effect on the global covariance, however, the covariances generated by the keypoints and the geometrical edge features take much larger values.

Let us finally note that the proposed algorithm $(NM, C4, C5)$ runs at around 8 fps in the presented cases.



(a) frame 14 (b) frame 25 (c) frame 205

(d) frame 260 (e) frame 495 (f) frame 15 (NM)

Fig. 5: Tracking for the Mario sequence with $(NM, C4)$ (a-e) and NM (f).

VI. Conclusion

In this paper we have presented a robust and hybrid approach of 3D visual model-based object tracking.Our approach has been tested via various experiments, on both synthetic and real images, in which the integration of keypoints has proven to be particularly efficient in the case of highly dynamic scenes with large inter-frame motions. Our method has also shown to be promising in terms of robustness with respect to illumination conditions, motion blur and noise, outperforming state-of-the-art approaches, for instance in the case of space rendezvous.

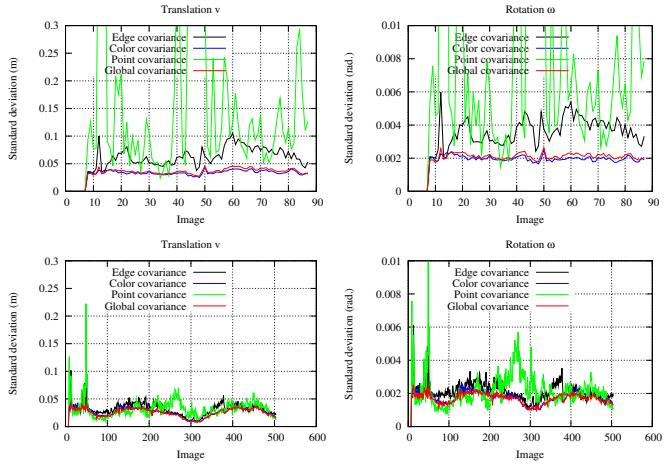


Fig. 6: Covariances on the pose displacement for (NM)(top) and $(NM, C4)$(bottom). Sqare root of traces on translation and rotation parameters of $\mathbf{\Sigma}^{g}_{\delta\mathbf{r}}$ (Edge), $\mathbf{\Sigma}^{c}_{\delta\mathbf{r}}$ (Color), $\mathbf{\Sigma}^{p}_{\delta\mathbf{r}}$ (Point) and $\mathbf{\Sigma}_{\delta\mathbf{r}}$ (Global) are represented.

References

[1] G. Bleser, Y. Pastarmov, and D. Stricker. Real-time 3d camera tracking for industrial augmented reality applications. *Journal of WSCG*, pp. 47–54, 2005.

[2] P. Bouthemy. A maximum likelihood framework for determining moving edges. *IEEE T. on PAMI*, 11(5):499–511, May 1989.

[3] T. Brox et al. High accuracy optical flow serves 3-D pose tracking: exploiting contour and flow based constraints. In *ECCV'06*, pp. 98–111, Graz, May 2006.

[4] C. Choi and H. I. Christensen. Robust 3d visual tracking using particle filtering on the special euclidean group: A combined approach of keypoint and edge features. *IJRR*, 31(4):498–519, April 2012.

[5] A.I. Comport, E. Marchand, M. Pressigout, and F. Chaumette. Real-time markerless tracking for augmented reality: the virtual visual servoing framework. *IEEE T-VCG*, 12(4):615–628, July 2006.

[6] T. Drummond and R. Cipolla. Real-time visual tracking of complex structures. *IEEE PAMI*, 24(7):932–946, July 2002.

[7] M. Haag and H.H. Nagel. Combination of edge element and optical flow estimates for 3D-model-based vehicle tracking in traffic image sequences. *IJCV*, 35(3):295–319, December 1999.

[8] V. Kyrki and D. Kragic. Integration of model-based and model-free cues for visual object tracking in 3D. In *IEEE ICRA'05*, pp. 1566–1572, Barcelona, 2005.

[9] Y. Ma, S. Soatto, J. Košecká, and S. Sastry. *An invitation to 3-D vision*. Springer, 2004.

[10] E. Marchand, P. Bouthemy, F. Chaumette, and V. Moreau. Robust real-time visual tracking using a 2D-3D model-based approach. In *IEEE ICCV'99*, pp. 262–268, Kerkira, 1999.

[11] G. Panin, Al. Ladikos, and A. Knoll. An efficient and robust real-time contour tracking system. In *ICVS*, pp. 44–44, 2006.

[12] G. Panin, E. Roth, and A. Knoll. Robust contour-based object tracking integrating color and edge likelihoods. In *Proc. of VMV 2008*, pp. 227–234, Konstanz, 2008.

[13] A. Petit, E. Marchand, and K. Kanani. Tracking complex targets for space rendezvous and debris removal applications. In *IEEE/RSJ IROS'12*, pp. 4483–4488, Vilamoura, 2012.

[14] A. Petit, E. Marchand, and K. Kanani. A robust model-based tracker for space applications: combining edge and color information. In *IEEE/RSJ IROS'2013*, Tokyo, 2013.

[15] M. Pressigout and E. Marchand. Real-Time Hybrid Tracking using Edge and Texture Information *IJRR*, 26(7):689-713, 2007.

[16] M. Pressigout, E. Marchand, and E. Mémin. Hybrid tracking approach using optical flow and pose estimation. In *IEEE ICIP'08*, pp. 2720–2723, San Diego 2008.

[17] V. Prisacariu and I. Reid. Pwp3d: Real-time segmentation and tracking of 3d objects. In *British Machine Vision Conf.*, September 2009.

[18] E. Rosten and T. Drummond. Fusing points and lines for high performance tracking. In *IEEE ICCV*, pp. 1508–1515, Beijing, 2005.

[19] J. Shi and C. Tomasi. Good features to track. In *IEEE CVPR*, pp. 593–600, Seattle, 1994.

[20] M. Tamaazousti et al. Nonlinear refinement of structure from motion reconstruction by taking advantage of a partial knowledge of the environment. In *IEEE CVPR*, pp. 3073–3080, 2011.

[21] L. Vacchetti, V. Lepetit, and P. Fua. Combining edge and texture information for real-time accurate 3d camera tracking. In *ACM/IEEE ISMAR'2004*, pp. 48–57, Arlington, 2004.

[22] H. Wuest and D. Stricker. Tracking of industrial objects by using cad models. *Journal of Virtual Reality and Broadcasting*, 4(1), 2007.