Vision-based Detection and Tracking for Space Navigation in a Rendezvous Context

Antoine Petit, Eric Marchand, Keyvan Kanani

VISION-BASED DETECTION AND TRACKING FOR SPACE NAVIGATION

Antoine Petit[1], **Eric Marchand**[2], **and Keyvan Kanani**[3]

[1]*INRIA, Rennes, France*
[2]*IRISA, Rennes, France*
[3]*Astrium, Toulouse, France*

ABSTRACT

This paper focuses on navigation issues for space autonomous, uncooperative rendezvous with targets such as satellites, space vehicles or debris. In order to fully localize, using a vision sensor, a chaser spacecraft with respect to a target spacecraft or debris, a visual model-based detection and tracking technique is proposed. Our tracking approach processes complete 3D models of complex objects, of any shape by taking advantage of GPU acceleration. From the rendered model, correspondences are found with image edges and the pose estimation task is then addressed as a nonlinear minimization. For detection, which initializes the tracking, pose estimation is based on foreground/background segmentation and on an efficient contour matching procedure with synthetic views, over a few initial images. Our methods have been evaluated on both synthetic images and real images.

1. INTRODUCTION

The active removal of heavy space debris (typically larger than 1000kg) has been identified as a key development to control the growth in the debris population and to limit the risk for active satellites. In that context, ESA has recently launched projects such as the Geostationary Servicing Vehicle (GSV) or the RObotic GEostationary orbit Restorer (ROGER), both tasked to capture, inspect, assist or re-orbit satellites in trouble. Astrium has been working on optimization and implementation of sensors and navigation solutions onboard a Debris Removal Vehicle with the main objective to ensure high safety proximity maneuvers. Here we focus on algorithms which achieves 3D visual detection and tracking of a complex target, based on its 3D model and using a monocular camera, and which allows estimation of the chaser state with respect to a target spacecraft or debris.

Our approach has been tested on real images, such as Soyuz-TMA rendezvous with ISS and Atlantis space shuttle pitch maneuver, to demonstrate its robustness to real data. A test campaign on simulated images (Spot family satellites) has also been carried out to quantitatively show the performances and robustness of both tracking and associated navigation.

1.1. 3D model-based tracking

Common model-based approaches use either point [2], edge features [5, 3] or a combination of both [13]. Edge features offer a good invariance to illumination changes or image noise and are particularly suitable with poorly textured scenes. For such class of approaches, the pose computation is achieved by minimizing the distance between the projected edges of the 3D model and the corresponding edge features in the image, extracted thanks to a 1D search for gradient maxima along the model edge normals. Weighted numerical nonlinear optimization techniques are used for the minimization. To reject outliers, methods like RANSAC [2] or M-Estimators [13, 3] are common trends to make the algorithm robust to occlusions and illumination variations.

Most of these approaches process 3D models which are made-up of lines. But achieving the model projection in the image has limitations and some problems appear when dealing with objects made of cylindrical, spherical, curved or complex shapes. Furthermore, complete polygonal models for complex objects can be too heavy and need to be manually redesigned to keep the most relevant edges of the scene and to make the algorithm computationally efficient.

A first challenge of our solution is to process a complete polygonal 3D model. In this sense, the whole information from the geometrical shape of any kind of scene can be used and a heavy phase of a manual redesign of the model is completely avoided. Our method relies on the use of the graphics process units (GPU) and of a 3D rendering engine. This allows to automatically manage the projection of the model and to determine the visible and prominent edge from the rendered scene. Such method has also been considered in [14]. An advantage of these technique is to automatically handled the hidden face removal process and to implicitly handle auto occlusions. A second challenge is to improve the robustness by combining both depth and texture edges and by including multiple hypothesis in the edge matching process.

1.2. 3D model-based detection

As the tracking procedure works frame by frame, an initial estimate of the pose has to be provided. In the literature, many object recognition methods consist in learning and classifying 2D bag of features extracted from various views of the query object ([7, 1]). As they have proven their efficiency for 2D textured objects for many recognition tasks, it is quite unadapted to our case, as we consider poorly textured or untextured 3D objects. Besides, real training images are hardly to be obtained for our applications. For these reasons we propose to rely on the shape and saliency of our object through a set of synthetic views obtained from the 3D model. Our detection method, while being generic and quite unsupervised, can be computationally heavy and too coarse when considering a single query image ([4, 12, 9]), and the tracking could then be lost. Therefore we propose to spread our detection strategy over a sequence of several images. This would enable to keep tracking the object and refine its pose within this sequence. In this way, our approach is similar to [11]. The overall purpose is to robustly match each image with a prototype view of the model, with respect to a position, an orientation in the image and a scale, which are estimated through a particle filtering framework. An initial coarse estimate of these parameters is necessary and is found thanks to a segmentation of the foreground object from the background.

The remainder of the paper is organized as follows. Section 2 gives an overview of the proposed method for tracking and Section 3 for detection. Some experimental results are given in Section 4.

2. 3D MODEL BASED TRACKING

2.1. Classical approaches

Our problem is restricted to model-based tracking, using a 3D model of the target. The purpose is to compute the camera pose which provides the best alignment between edges of the projected model and edges extracted in the image.

Such approaches have proved to be very efficient and various authors have proposed different formulation of the problem (eg, [5, 3]). Although one can find some difference in these various solution, the main idea is the following. Given a new image, the 3D model of the scene or the target is projected in the image according to the estimated previous camera pose $\mathbf{r}$. Each projected line $l_i(\mathbf{r}) = pr(L_i, \mathbf{r})$ of the model is then sampled leading to a set of 2D points $\{\mathbf{x_i}\}$. Then from each sample point $\mathbf{x}_i$ a 1D search along the normal of the projected edge is performed to find a corresponding point $\mathbf{x}'_i$ in the image.

In order to compute the new pose, the distances between points $\mathbf{x}'_i$ and the projected lines l_i are minimized with respect to the following criteria [3] :

$$\Delta = \sum_i \rho(d_\perp(l_i(\mathbf{r}), \mathbf{x}'_i)) \quad (1)$$

where $d_\perp(l_i(\mathbf{r}), \mathbf{x}'_i)$ is the distance between a point $\mathbf{x}'_i$ and the corresponding line $l_i(\mathbf{r})$ projected in the image from a pose $\mathbf{r}$. ρ is a robust estimator, which reduces the sensitivity to outliers. his is a non-linear minimization process with respect to the pose parameters $\mathbf{r}$. In [3], the minimization process follows the Virtual Visual Servoing framework similar to Gauss-Newton approach.

Our purpose to consider complex shape targets, textured or untextured, leads to forget the notion of 3D sharp edges as in [3] or in our previous work [8] and to consider only 3D points that belongs indifferently to sharp edges or to the "occlusion boundaries" or rims, or to texture edges. Two issues have then to be considered: complex model projection and 3D points selection.

The tracking algorithm in our method is structured as follows:

- Projection of the detailed model with respect to the pose $\mathbf{r}_k$ computed for the previous image I_k. To achieve this process we rely on the graphics libraries (openGL) that allows to perform quickly this projection regardless the complexity of the model thanks to the use of the GPU.
- From this projection we generate 3D measurement points $\mathbf{X}_i$ by extracting edges from the depth and texture discontinuities of the rendered model.
- We then search for corresponding edge points $\mathbf{x}'_i$ in image I_{k+1}. To allow a better robustness, we propose a multiple hypothesis version of the algorithm.
- Last step is to estimation of the pose $\mathbf{r}_{k+1}$ which minimizes the errors $d(\mathbf{x}_i, \mathbf{x}'_i)$ between the point extracted from the image $\mathbf{x}'_i$ and the projection of the selected 3D points $\mathbf{x}_i(\mathbf{r}) = pr(\mathbf{X}_i, \mathbf{r})$, with the following criteria:

$$\Delta = \sum_i \rho(d(\mathbf{x}_i(\mathbf{r}), \mathbf{x}'_i)) \quad (2)$$

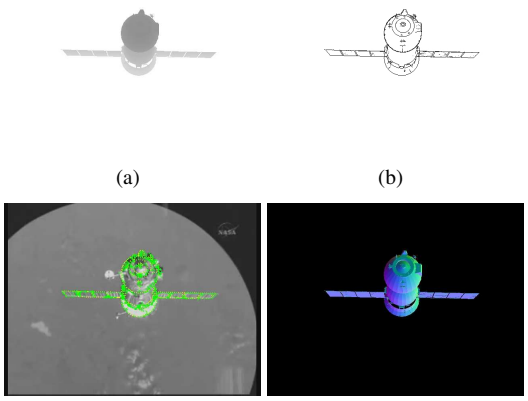
2.2. Generation of 3D measurement points

As in [14], at each acquired image I_{k+1}, the model is rendered and projected using an openGL rendering engine, with respect to the previous pose $\mathbf{r}_k$.

Our goal is to obtain a set of 3D points $\mathbf{X}_i$ that belong to target rims, edges and visible texture from the rendered textured scene and the depth buffer. Our approach follows [14] and is related to the techniques of silhouette generation of polygonal models described in [6].

Edge extraction using depth and texture discontinuities. From the depth or Z-buffer, which corresponds to the depth values of the scene according to the camera location at each pixel point (Figure 1(a)), we can determine the discontinuities which suit the geometrical appearance of the scene. Therefore, we apply a second order differential operator, such as a Laplacian filter, to these computed Z values, resulting in a binary edge map of the visible scene ((Figure 1(b)). In our approach, we have implemented the filtering computations on the GPU through shader programming, resulting in a much lower computational time.

In case of highly textured scenes, geometrical edges are not sufficient and ambiguities with texture edges arise, which results in false matching and thus local minima during the pose estimation process. An improvement of our method is then to combine the depth discontinuities with texture discontinuities. The rendered textures of the 3D model are thus processed by a classical Canny edge algorithm and the obtained edges are added to the ones generated from the depth buffer.



(a) (b) (c) (d)

Figure 1: On (a) is represented the z-buffer of the rendered 3D model using Ogre3D, from which the edge map is computed (b).This edge map is then sampled to extract measurement points, reprojected on the current image and from them a 1D search along the edge normal is performed to find a matching edge point in the acquired image (c). (d) shows the normal map of the scene.

Generation of 3D measurement points. Given the edge map of the complete scene, the 3D coordinates of the edge points in the scene can be computed thanks to the Z-buffer and the pose used to project the model. As dealing with the whole edge map can be computationally intensive, we can sample it along x and y coordinates of the image in order to keep a reasonable number of these edge measurement points.

Besides, the tracking phase (see Section 2.B) requires the orientation of the edge underlying a measurement point $\mathbf{x}_i$. For the texture edges, it is done within the Canny algorithm on the rendered textures. For the depth edges, we compute the Sobel gradients along x and y on a grey level of the normal map of the scene. The normal map of the scene associates the [R,G,B] value of each pixel location to the coordinates in the world frame of the normal to the corresponding surface in the scene (see Figure 1(d)). These basic image processing steps, as well as the retrieval of the normal map, are also processed on the GPU.

2.3. Pose Estimation and low level tracking

Tracking from edge measurement points. The measurement points are then processed to track corresponding edges in the image. In a similar manner to [13, 3, 14], we perform a 1D search along the normal of the underlying edge (Figure 2 and Figure 1(c)) of each $\mathbf{x}_i$. A common approach is to choose the pixel with the maximum gradient as the matching edge point $\mathbf{x}'_i$ in the image.

Minimization of a distance to a line. Once correspondences are established, the goal is then to estimate the new pose $\mathbf{r}_{k+1}$ that realigns the measurement points with their matching observed image points $\mathbf{x}'_i$. This task is addressed by minimizing the errors $d(\mathbf{x}_i(\mathbf{r}), \mathbf{x}'_i)$. As in [5, 3, 14], our approach considers the distance between the projected 3D line $l_i(\mathbf{r})$ underlying the projected 3D measurement point $\mathbf{X}_i(\mathbf{r})$ and the selected matching point $\mathbf{x}'_i$ in the image (see Figure 2). Criteria (2) becomes:

$$\Delta = \sum_i \rho(d_\perp(l_i(\mathbf{r}), \mathbf{x}'_i))) \tag{3}$$

where ρ is a robust estimator used to reject outliers (Tuckey estimator), and $d_\perp(l_i(\mathbf{r}), \mathbf{x}'_i)$ is the distance between a point $\mathbf{x}'_i$ and the corresponding line $l_i(\mathbf{r})$. It has to be noted that for sharp edges the 3D point $\mathbf{X}_i(\mathbf{r})$ is not modified when we modify $\mathbf{r}$. This is no longueur the case for points $\mathbf{X}_i(\mathbf{r})$ that belong to an occlusion rim. Nevertheless, since the camera motion between two successive images is very small, this approximation has no impact on the efficiency of the approach.

The optimization technique is similar to the virtual visual servoing framework described in [3], which relates this optimization problem to a visual servoing issue.

Multiple hypothesis solution. In order to improve the robustness of the pose estimation and to avoid problems due to ambiguities between edges, it is possible to consider and register different hypothesis corresponding to potential edges. They correspond to different local extrema of the gradient along the scan line. As in [13, 10], we choose the hypothesis which has the closest distance to the projected 3D line l_i during the minimization process. The cost function becomes :

$$\Delta = \sum_i \rho(\min_j d_\perp(l_i(\mathbf{r}), \mathbf{x}'_{i,j})) \tag{4}$$

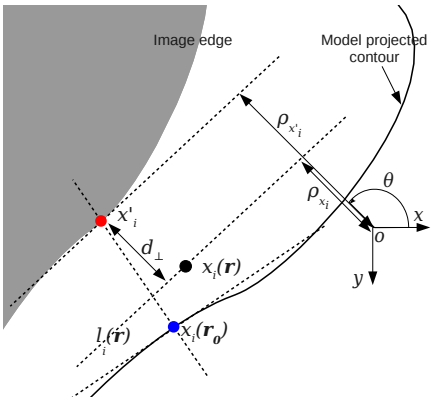


Figure 2: Moving edge principle: from the initial pose $\mathbf{r}_0$, 1D search along the projected contour underlying the measurement point. Distance of a point $\mathbf{x}'_i$ to a corresponding line $l_i(\mathbf{r})$ within the minimization process.

where points $\mathbf{x}'_{i,j}$ are the selected candidates for each measurement point $\mathbf{x}_i$.

3. 3D-MODEL BASED DETECTION AND POSE ESTIMATION

Here is addressed the issue of initializing our tracking algorithm, for a poorly textured object, with a potentially cluttered background or not. The proposed solution only relies on the 3D model of the object. The pose estimation by detection is achieved over a few initial successive images. The overview of the method is described as follows (and see Figure 3) :

- **Offline learning stage** : it aims at building a hierarchical model view graph leading to prototype views V of the model.
- **Online detection stage** :
  - Segmentation of the object using a real-time bilayer segmentation technique. By computing binary moments of the extracted silhouette, an initial estimate of its center of gravity (x_c, y_c), its orientation θ and its area A.
  - For each prototype view, a particle filter with respect to the state $\mathbf{x} = [x_c \quad y_c \quad \theta \quad A]^T$ is set.
  - Through a Bayesian framework, at each frame the most likely prototypical view V and its corresponding estimate $\hat{\mathbf{x}}_\mathbf{v}$ are determined, providing the complete pose.
  - When a view reaches a sufficient fitting probability with respect to the others, the process is stopped. The pose is thenrefined by traversing through the hierarchical view graph.

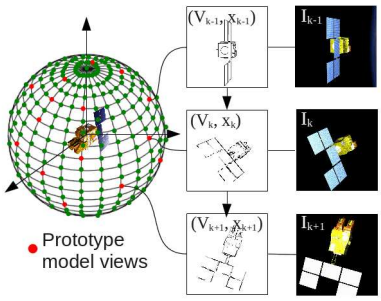


Figure 3: Framework of the detection process. Synthetic views of the model are generated on a view sphere. Transitions between model views follow a Markov jump model.

3.1. Hierarchical model view graph

Generation of synthetic views. The purpose of the method is to match an input image with a synthetic view generated from the 3D model with respect to a similarity metric. These synthetic views thus have to cover the whole 6D search space. We propose, as in [4, 12, 9, 11], to generate these views on a view sphere centered on the 3D model, parametrized by two DoF (latitude and longitude). These is performed by setting virtual cameras at uniformly spaced viewpoints (Figure 3 left). From this rendered views, we extract their contours by processing the corresponding depth buffer of the scene through a Laplacian filter. Thus, in a different manner to [11], we do not only consider the silhouette contours but also the crease contours of the object.

Building a hierarchical view graph. Since this process can be computationally challenging when considering the whole database, we iteratively cluster the views into a hierarchical view graph. At the first level of hierarchy, we build clusters within disjoint neighborhoods in the spherical space, in order to cope with memory requirements. This is done by comparing the views with each other in each neighborhood with respect to an edge-based similarity metric. The result is a set of clusters, each represented by a prototype model view. It defines the first level of our hierarchy. We proceed in the same way with these views. Thus we can iteratively build successive hierarchical levels with this method until a reasonable number (around 30) of prototype model views is reached.

3.2. Segmentation and silhouette extraction

As matching the model views with the images gives two DoF (lattitude and longitude around the object), a solution is proposed to estimate the four other parameters, which are the position and orientation of the object in the image, and its scale or area in the image. It relies on

the segmentation of the silhouette of the object, which is supposed to be moving in the image. However these parameters are often coarsely computed due for example to some cluttered background, occlusions so it is necessary to adapt and refine them in order to find a consistent best match among the few prototype model views.

3.3. A Bayesian Framework for matching input images to prototype model views

Our problem consists in aligning the prototype views to the input images and finding the most likely one. It has appeared relevant to reason on several input images considering that the results provided by segmentation can be too coarse. Besides, as this detection method can be computationally challenging, the process is spread over a few frames instead of focusing only on the first frame with more exhaustive and costlier process and risking to loose track of the moving object in the next frames. In order to assure a smooth transitions between the prototype model views throughout the image sequence, a bayesian framework is proposed to determine the most likely view V_k^j, along with the parameters $\hat{\mathbf{x}} = \begin{bmatrix} \hat{x_c} & \hat{y_c} & \hat{\theta} & \hat{A} \end{bmatrix}^T$, estimated through particle filtering, at each frame I_k (see Figure 3).

4. EXPERIMENTAL RESULTS

4.1. Implementation

The rendering process of the 3D polygonal and textured model relies on OGRE (Object-oriented Graphics Rendering Engine). The library avoids to use explicitly the underlying system libraries (Direct3D or OpenGL). We have considered shader programming for some image processing steps during the rendering and edge generation phases. This is done using OpenGL Shading Language (GLSL), supported by OGRE. Regarding hardware, an NVIDIA NVS 3100M graphic card has been used, along with a 2.8GHz Intel Core i7 CPU.

For tests procedures, we have performed experiments on both real and synthetic image sequences. They consist in evaluations of the algorithm described in the previous sections, for the texture and untextured cases and for the single and multiple hypothesis solutions.

4.2. Tests on real images

The first example deals with the tracking of the Soyuz TMA-12 spacecraft during its rendezvous phase with the International Space Station (ISS). With its three different modules, the main body of the spacecraft is of complex shape, with curved and fuzzy edges, but the solar arrays tend to facilitate the tracking. Here the tracking procedure consists in only using the depth discontinuities in the rendered scene to generate the visible and prominent edges (see Figure 1(d)), and the single hypothesis solution has shown to be sufficient for this sequence. Despite uncertainties over the camera internal parameters, the 3D model consistency, the cluttered background and the low quality of the video, the tracking phase is successfully achieved (see Figures 5(a)- 5(d)). For detection, we observe that as the segmentation is not very precise (Figure 4(a)), it takes five input images (three are shown here, Figures 4(b)-4(d)) to match with a sufficiently likely model view along with acceptable state parameters to initialize the tracking.

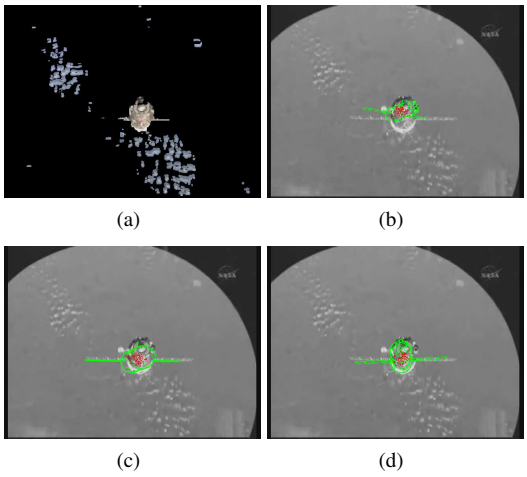


Figure 4: Detection for the Soyuz sequence. From the coarse segmentation phase (a) and an unfitted model view (b) the process converges to a consistent one (d).

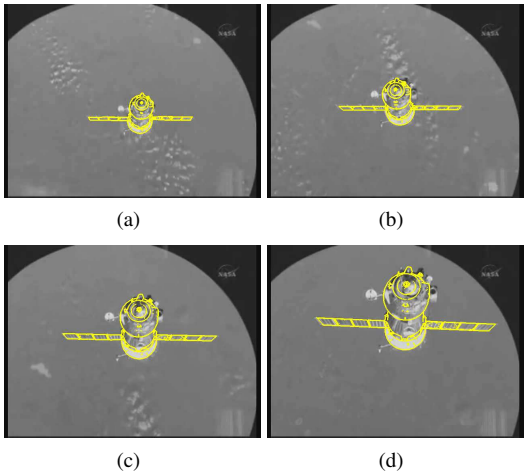


Figure 5: Tracking relying on depth edges for the Soyuz sequence.

The second example concerns the Atlantis Space shuttle performing a pitch maneuver for its rendezvous with the ISS. An untextured 3D model of the spacecraft has been processed for the tracking, together with the multiple hypothesis registration process and a Kalman filter

with a constant velocity model on the pose parameters. Figure 6 shows the results of the detection procedure. The segmentation phase (Figure 6(a)) quite finely extracts the shape of the shuttle and then the matching prototype and the corresponding state parameters can then be quite immediatly determined (after 3 input images, (Figures 6(b), 6(c), 6(d))) to initialize the tracking phase. Figures 7(a)- 7(f) shows the tracking then properly performed over the sequence, with a robustness to some illumination changes. The multiple hypothesis solution and the Kalman filter are necessary to handle the shuttle flip in the image (Figure 7(d)).

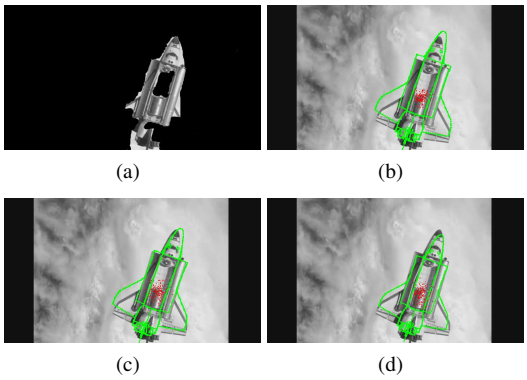


Figure 6: Detection for the Atlantis sequence

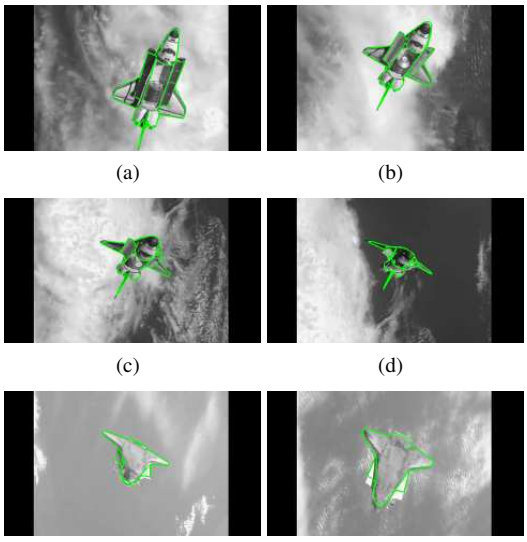


Figure 7: Tracking relying on depth edges for the Atlantis sequence, along with a multiple hypothesis registration process and a Kalman filter.

4.3. Tests on synthetic images

The evaluation has been performed using a ray-tracing simulator developed by Astrium for space environments. We focus here on the case of the Spot satellite family. For space debris removal concerns, we consider an arbitrary rotation for the target attitude and a chaser spacecraft is supposed to be located on a similar orbit, with a slightly different eccentricity in order to make the chaser fly around the target (Figure 8(a)), in the $x_{Orb} - z_{Orb}$ plane of the orbital frame (see Figure 8(b)). The chaser is also equipped with a camera filming the target and a spot light to lighten the target.

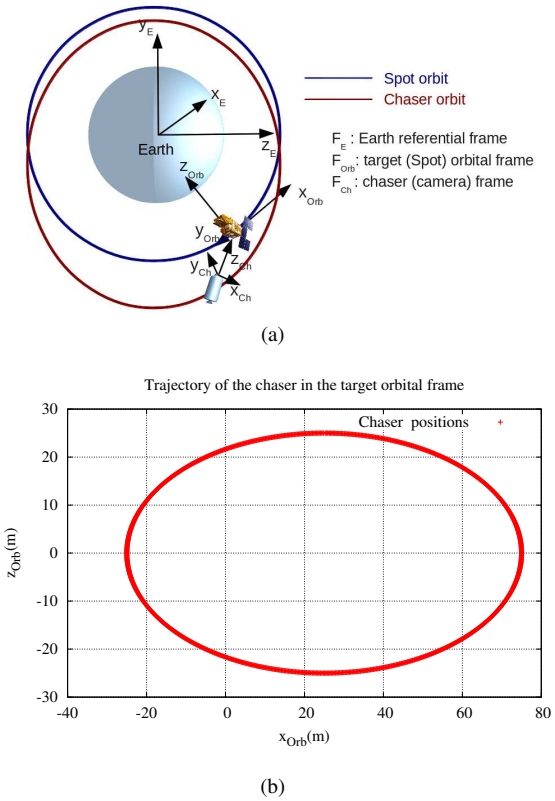


Figure 8: Chaser and target (Spot) orbits in the Earth reference frame

As it can be observed on Figures 9(f)- 9(m), the tracking presents good performances throughout the whole sequence. In order to quantify the different results, we separately evaluate the accuracy of rotation and translation components of an estimated camera pose $\hat{\mathbf{r}}$ with respects to the true pose $\mathbf{r}^*$, as we can be provided with Ground Truth. The results for the single (SH) and multiple hypothesis (MH) approaches, relying on both depth and texture edges, and with or without Kalman filtering are represented on Figure 10. They show that when using multiple hypothesis along with the Kalman filter, translation and rotation errors can be kept small, especially when the target is far from the chaser with low luminosity (see Figure 9(l)), and when the solar panels flip in the image (see Figure 9(j)), leading to some local minima for the single hypothesis solution. Figures 9(b)-9(e) show the detection process that correctly initializes the tracking, despite the coarse segmentation phase (Figure 9(a)).

4.4. Computational costs

Thanks to the implementation of several image processing phases on the GPU, the execution time could be considerably reduced. The whole algorithm can be processed at around 15 fps when relying on depth edges. The multiple hypothesis solution does not affect much computations. Including texture information makes the process costlier, since more points are considered, with a 7 fps framerate for the Spot sequence. The detection technique runs at 0.5 fps.

5. CONCLUSION

This paper presents detection and tracking methods suited for complex, textured or untextured objects in deep space environments, for space rendezvous and space debris removal purposes. The tracking relies on the generation of visible and salient edges from the 3D complete polygonal model, projected using a rendering engine. Information from both geometrical and texture discontinuities are used. This technique avoids the heavy and restrictive processing of a 3D line model. The method is similar to the classical approaches as it consists in the realignment of the generated model edges with edges detected in the image. A model-based detection framework has also been designed to initialize the tracking, through a Bayesian edge matching procedure over a few initial images. From the tests carried out on real images, our approach qualitatively presents satisfactory results, for both tracking and detection phases. Thanks to the realistic image simulator, performances can be measured, showing the relevance of the approach. The implementation proves to be computationally efficient.

REFERENCES

[1] Serge Belongie, Jitendra Malik, and Jan Puzicha. Shape matching and object recognition using shape contexts. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(4):509–522, April 2002.

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

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

[4] Christopher M. Cyr and Benjamin B. Kimia. A similarity-based aspect-graph approach to 3d object recognition. *International Journal of Computer Vision*, 57:5–22, 2004.

[5] T. Drummond and R. Cipolla. Real-time visual tracking of complex structures. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 24(7):932–946, July 2002.

[6] T. Isenberg, B Freudenberg, S. Schlechtweg, and T. Strothotte. A developer guide to silhouette algorithms for polygonal models. *IEEE Comput. Graph. Appl.*, 23(4):28–37, 2003.

[7] V. Lepetit and P. Fua. Keypoint recognition using randomized trees. *IEEE Trans. on PAMI*, 28(9):1465–1479, September 2006.

[8] A. Petit, E. Marchand, and K. Kanani. Vision-based space autonomous rendezvous : A case study. In *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems, IROS'11*, pages 619–624, San Francisco, USA, September 2011.

[9] Christian Reinbacher, Matthias Ruether, and Horst Bischof. Pose estimation of known objects by efficient silhouette matching. In *20th International Conference on Pattern Recognition (ICPR)*, 2010.

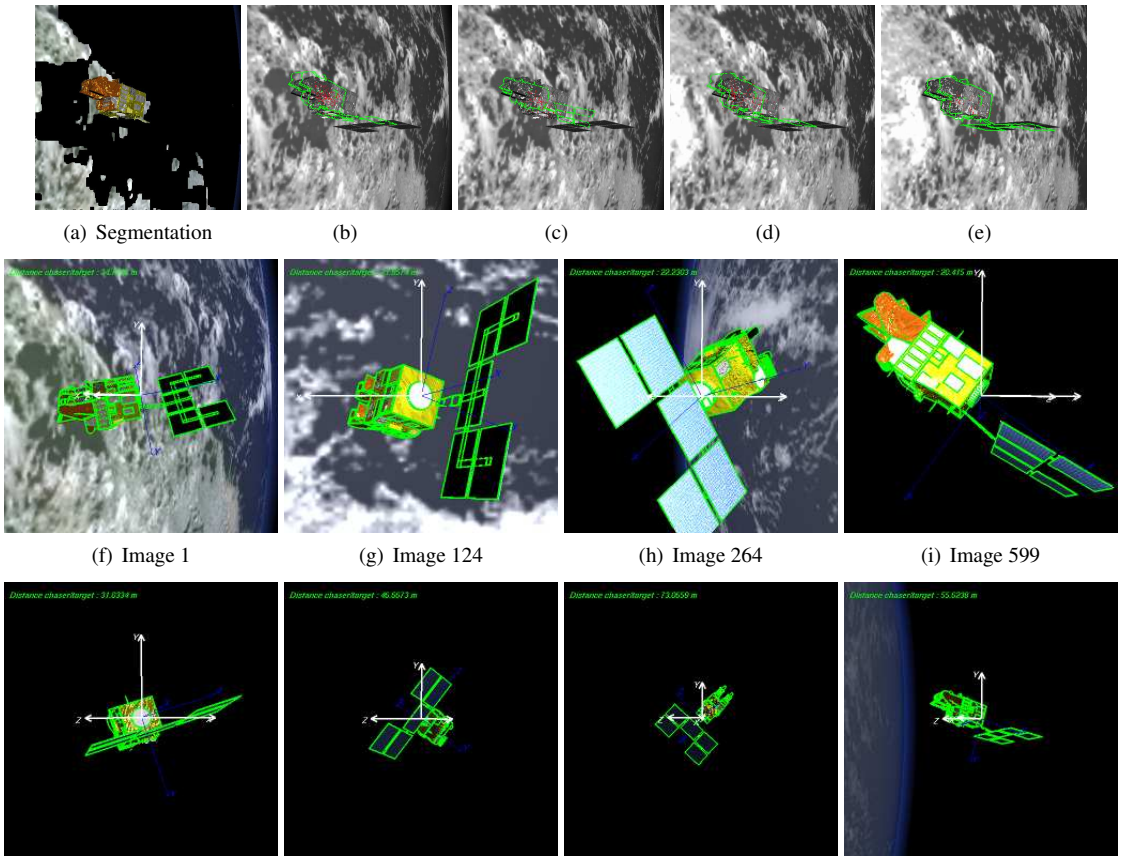
[10] C. Teulière, E. Marchand, and L. Eck. Using multiple hypothesis in model-based tracking. In *IEEE Int. Conf. on Robotics and Automation, ICRA'10*, pages 4559–4565, Anchorage, Alaska, May 2010.

[11] A. Toshev, A. Makadia, and K. Daniilidis. Shape-based object recognition in videos using 3d synthetic object models. *Computer Vision and Pattern Recognition, IEEE Computer Society Conference on*, 0:288–295, 2009.

[12] Markus Ulrich, Christian Wiedemann, and Carsten Steger. Cad-based recognition of 3d objects in monocular images. In *Proceedings of the 2009 IEEE international conference on Robotics and Automation*, ICRA'09, pages 2090–2097, Piscataway, NJ, USA, 2009. IEEE Press.

[13] L. Vacchetti, V. Lepetit, and P. Fua. Combining edge and texture information for real-time accurate 3d camera tracking. In *ACM/IEEE Int. Symp. on Mixed and Augmented Reality, ISMAR'04*, pages 48–57, Arlington, VA, November 2004.

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



(a) Segmentation (b) (c) (d) (e)

(f) Image 1 (g) Image 124 (h) Image 264 (i) Image 599

(j) Image 754 (k) Image 864 (l) Image 1099 (m) Image 1439

Figure 9: The segmentation of the target (a) is quite coarse due to the cluttered background. We observe the detection, which takes 8 images to converge (4 are shown here, (b)-(e)) and the tracking ((f)-(m)), relying on depth and texture edges, with multiple hypothesis and Kalman filtering.

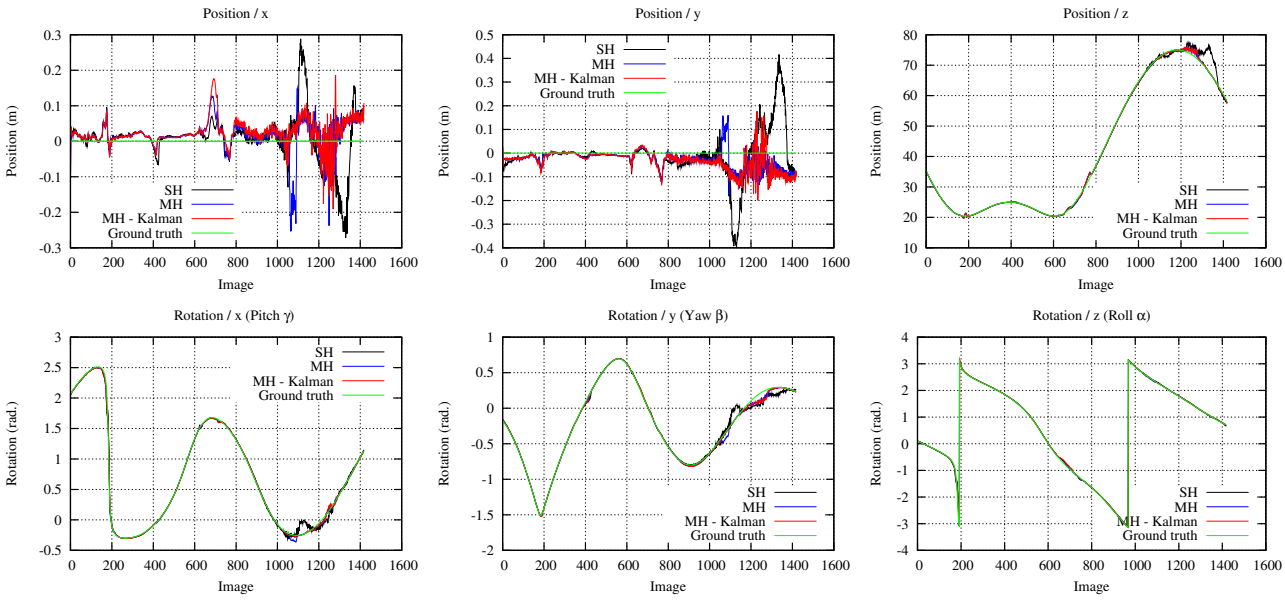


Figure 10: Estimated camera pose parameters of the target over all the sequence, along with the ground truth, for the single and multiple hypothesis solutions, together with a Kalman filter or not.