Multi-View Parameters Tracking of Real Objects

Yannick Perret
yperret@ligim.univ-lyon1.fr

Thierry Excoffier
exco@ligim.univ-lyon1.fr

Saïda Bouakaz
bouakaz@ligim.univ-lyon1.fr

L.I.G.I.M., Université Claude Bernard Lyon I,
43, Boulevard du 11 novembre 1918,
69622 Villeurbanne Cedex, FRANCE

Abstract

A parameters tracking method of moving objects is proposed. This method uses multiple views of the tracked subject. It also uses a model of the tracked subject, which includes geometrical information and parameters, seen as liberty degrees of the object. The problem is defined in term of adequation between images. The method searches the parameters which generate the closed synthetic views of the subject regards to the real images. This is done by using a similarity function between images based on dense comparisons. It allows reliability and occlusions robustness. Parameters tracking is achieved using an adapted simplex-based algorithm, giving fast and stable results. The system requires an initial parameters vector and then tracks parameters variations of the subject during the sequences. Some particular features are presented, such as textures auto-extraction, which permit to find the subject textures, and model adjustment which allow auto-refinement of the geometric model. A set of experiments illustrates the efficiency and the robustness of the approach. Both artificial and real video sequences are used, illustrating various situations.

Keywords: Parameters tracking, multi-view, dense comparison, geometrical model.

1 Introduction

Three dimensional parameters tracking is an important task for several application is the computer vision domain. We can roughly consider three main categories of application for that:

- movements tracking and analysis of real subjects;
- coherent immersion in virtual realities, which requires the knowledge of subjects movements;
- spatial information extraction for augmented reality, in order to make synthetics elements interact in a credible way with reality.

In all these cases, ones needs to know the values and the evolution of liberty degrees – called parameters – of real objects.
There are several kinds of captors (magnetic sensor, data gloves, *etc*.) worn by subjects which allow to track spatial movements, as in movie industry or in [12, 13]. We choose to use only video streams as input because captors have limitations, such as cost, installation time, inadequacy (i.e. for unreachable subjects) or physical constrains (i.e. water, interferences and so on). There are various applications for parameters tracking, including video surveillance, sportsmen movements tracking, gesture recognition, *etc*.
The first part of this paper presents existing works in the parameters tracking area. The second part describes our approach, which can be divided in two parts: similarity measure between images, and parameters optimization. The third part describes the determination of initial parameters vector, and finally the last part presents several experiments for various situations, showing the validity of our approach.

2 Related Work

It exists several approaches to track objects, humans or part of them in video sequences.

A classical approach consist in tracking particular elements putted on subjects, such as colored spots, balls or textured patches. These elements can easily be tracked in 2D images, and their spatial position retrieved using cameras calibration information. In [5] curves drawn on faces are used to track facial attitudes. Such approach is often uses in movies industry, and gives good results in controlled situations. But it cannot be applied if subjects are unreachable or if spots are incompatible with the subject or its environment (i.e. for swimmers, many sportsmen).
Other approaches use features extraction in images. These features can be particular points, areas or contours[1], extracted from the subject. These features are associated with parts of the subject model, and their spatial characteristics can be estimated using camera calibration information. This approach requires to know which features are expected on the subject and to be able to recognize them without ambiguity. Moreover, hi-level features can be altered by occlusions or extraction errors.
Existing approaches also use movements estimation in image, using optical flow[2], textures variations[8], moving contours tracking or other estimators to compute global movements or deformations of the subject. It is efficient for target tracking, but is more delicate to applicate to multi-body articulated subjects, and is sensible to occlusions.
Other researchers have used image analysis/synthesis collaboration to perform parameters tracking[11]. Synthetic images are then used to confirm or to guide parameters validity. In Ref. [4] this approach have been used to track 2D deformations in images.

3 Approach

We present in this section the approach we have developed for parameters tracking problem. Several calibrated cameras are filming the tracked subject. Parameters tracking problem is defined as finding the parameters vector which generate the closest synthetic views with respect to the real ones. This implies to define a similarity function between images, in order to measure how ”close” a set of synthetic views is regards to a set of real images. The resulting parameters vector is the one which optimize this similarity function.
Our method is based on a prediction / verification / correction approach. A parameters vector is proposed, its adequation is measured using the similarity function, and a new one is generated in order to improve the similarity (see Fig 1).
This implies to define a similarity measure between images, and to search the optimum of this function.

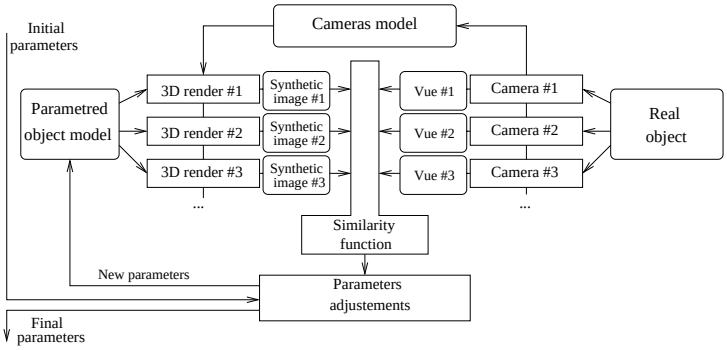

Figure 1: Description of the approach

3.1 Similarity measure

In this section, we present a similarity measure between images. We want to obtain a measure of the "adequation" of the synthetic views regard to the real ones. Human criterion[10] are based on high level features comparisons, useful to do recognition, but not for precise adequation. In our case we know precisely what we are comparing, so we directly use the image content.
We define images as pixels matrix and pixels are referenced by their coordinates: $(x, y) \rightarrow I[x, y]$ is the pixel at the row x and the column y of image I.
As we use colored images, pixels are vectors in the color space used. We have choose to express colors in the YC_rC_b color space, which allow separate representation of luminance (Y) and chrominances (C_rC_b) (see [9] for more details on color spaces and their properties). To compare two images I_1 and I_2, we use dense comparison. It consist to compute distances between every pixels of the two images. The average value is then retained:

$$D_{ist}(I_1, I_2) = \frac{1}{H * W} \sum_{x,y} d_{pix}(I_1[x, y], I_2[x, y]) \qquad (1)$$

with H and W respectively the width and the height of images, and d_{pix} a distance function between pixels.
This pixel distance is defined regarding that images to be compared are not exactly the same. In particular, errors in modelisation of surfaces properties and lights positions generate variations in pixels colors. Moreover unknown objects generates differences in shadows rendering. For monochromatic lights, these errors generate luminance variations in pixel color, and let chrominance roughly unchanged. We take that point in account by separating pixel distance into luminance and chrominance sub-distances. Euclidean distance in luminance space d_{lum} and in chrominance space d_{chrom} are combined in order to give more importance to the less-sensitive information, the chrominance distance:

$$d_{pix}(p_1, p_2) = \alpha * d_{chrom}(p_1, p_2) + (1 - \alpha) * d_{lum}(p_1, p_2) \quad (2)$$

with p_1 and p_2 two pixels and α the relative weight of the two distances.
The distance between images D_{ist} is applied to real images R_i from cameras, and corresponding synthetic views $S_i(p)$, with i a given view and p a parameters vector.
As we got several pair of real / synthetic views, each corresponding to a camera, we extend the measure as follow:

$$D(p) = \sum_i \alpha_i D_{ist}(R_i, S_i(p)), \sum \alpha_i = 1 \qquad (3)$$

with the α_i allowing to control the relative contribution of each view to the global measure. Criterion for the α_i may be for example image resolution or camera quality.

3.2 Optimization

We need to find the parameters vector which generates the closed synthetic images regards to the real ones, that is to say the parameters vector that maximize the similarity function. We have choose a robust methods, based on **simplex** approach (see Ref. [6] for more details). The simplex approach lead several advantages:

- partial robustness to local minima;
- fast and reliable convergence, even with large number of parameters;
- no use of partial derivative of the similarity function.

This method gives good results, and appear to be robust, in particular to little perturbations in the similarity function.
This method have some disadvantage, however. The Simplex tend to "contract" itself when approaching the solution, making the progression slower. To prevent this, we have include randomize perturbation in parameters values, which is based on annealing principle or mutations in genetic algorithms[3]. This allow to reduce excessive contractions and increase convergence speedup in final steps.

3.3 Initial parameters vector

An important hypothesis in our system is that we own an initial parameters vector near the theoritical values for the beginning of the sequences. Finding this initial vector "by the hand" is not always easy.
That's why we have included in our 3D description language **adjustment** points. These points are 3D entities, linked to particular parts of the subject. The render module can generates their 2D positions in projected images.
We have created a new similarity function for this case, based on distances between 2D points. For each parameters vector Pv, adjustment points a_i are projected in each synthetic views V_j. They compared using an Euclidean distance d to corresponding 2D coordinates u_i given by the user.
Let call $P_{V_j}(x)$ the projection of the 3D point x in the view V_j. Similarity S can they be written:

$$S(Pv) = \sum_j (\sum_i d(u_i, P_{V_j}(a_i))) \qquad (4)$$

The user specify the 2D positions u_i of the adjustment points in the reference images for each view. Our program uses a this similarity function and search the parameters vector which minimize the new distance.
With this method, it is possible to find an approximate parameters vector, even with articulated subject, if adjustment points are given at strategic positions such as articulations.
This technique is used for all the real examples given in the next section.

4 Results

We have realized an implementation of the method described in this paper. We have firstly tested our method on synthetic sequences, generated by our render module. There are two main reasons for that choice:

- we have a total control of the sequences content, allowing us to test particular features;
- we know the parameters values used to generate the sequences and have a perfect model of the tracked subject. It allows us to measure errors between the real solution and the results found by our method.

Synthetic sequences are generated with an additive noise and supersampling in order to make the images more “real”.
For each test, the program only knows the subject model and ignores other informations such as background or other objects in the scene (see [7] for other synthetic examples).

4.1 Synthetic sequences

For this first test, the sequences contains a moving chair in a room. This chair is filmed by three virtual cameras, with images size of 400×400 pixels. The chair moves and rolls in the room along the sequences (see figure 2).
Parameters for this test are the three translation values and the three rotation angles which define the chair is the space, so a total of six parameters.
Figure 3 shows errors along the sequences, for angular and position parameters.

Figure 2: Three images from the “Chair” sequences at different time (for one of the cameras)

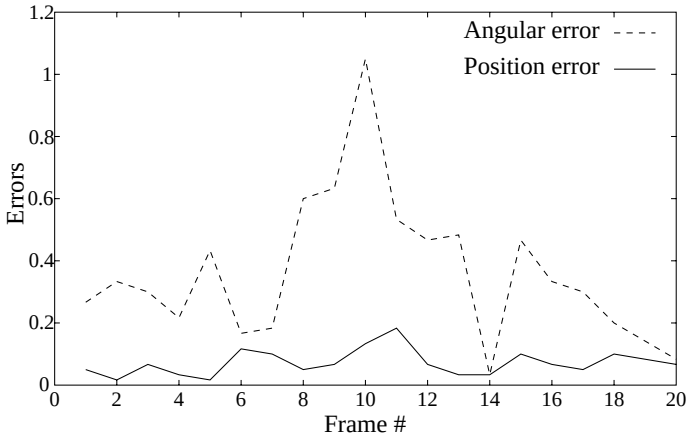

Figure 3: Results for “Chair” sequences: average angular errors: 0.15 degrees; average translation errors: 0.05 units (chair size is 5x5x10 units)

Another important point is the occlusion robustness. To test this, we have taken again the chair sequences, but with a grid around it. Sequences have been generated with various occlusion level from 30% to 65%, and the same conditions for the rest (cameras, parameters...). Figure 4 shown samples of various sequences, each corresponding to different occlusion level.

Figure 5 shows the angular errors along the sequences, with three different levels of occlusion (the translation error is not reported here for legibility purpose, but shows a similar behavior). We can see that the errors globally increase with the occlusion level, but the algorithm still manages to track parameters. It shows the ability of our method to resist to occlusions, with an acceptable error level.
The algorithm starts to diverge for occlusion level higher than 70% with the chair subject.

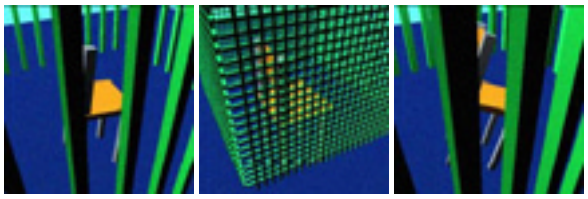

Figure 4: Samples from the ”occlusion” sequences. Each sequence present a different kind of grid, with various occlusion level of the chair.

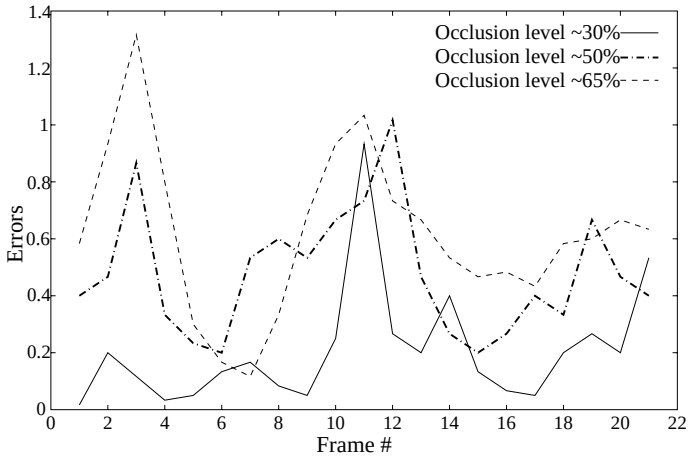

Figure 5: Results for occluded chair. Lines shown the angular error along the sequences, for various occlusion level.

4.2 Real examples

We have tested our method on real video sequences. In that case, it is not possible to compute results error, as theoritical values are not known. To obtain an idea of the results validity, we generate the synthetic images of the subject using the founded parameters along the sequence, for each views. These images are then superposed to the real ones in order to visually compare their adequation.
The first example is a two cameras sequence in which a little box is moving on a table. The two sequences are about seventy 320×240 images each, taken every 0.06 seconds (top of fig. 6), with a very bad quality (color planes are shifted, making blue or red bands on the objects sides because the low quality cameras). Bottom of the figure 6 shown the images addition at the beginning, the middle and the end of the sequences (here just for one of the cameras). For both cameras, superpositions are good all along the sequences.
The second example is a two-pieces articulated object, filmed by two cameras. The object is moving and the angle between the two pieces is changing (see top of fig. 7). Bottom of figure 7 shows the images addition at the beginning, the middle and the end of the sequences (here just for one of the cameras).

4.3 Model adjustment and texture extraction

An important point of our method is that no semantic information about parameters is needed *a priori*. It is so possible to use various kind of parameters, such as colors, sizes of objects and so one. In particular it is possible to define subject characteristics as parameters. It allows to refine an existing model with regards to real images of the tracked subject.
In this example we have a two-cameras sequence of a *Scotch* box moving on a table (see Fig. 8). We first create an *a priori* model of

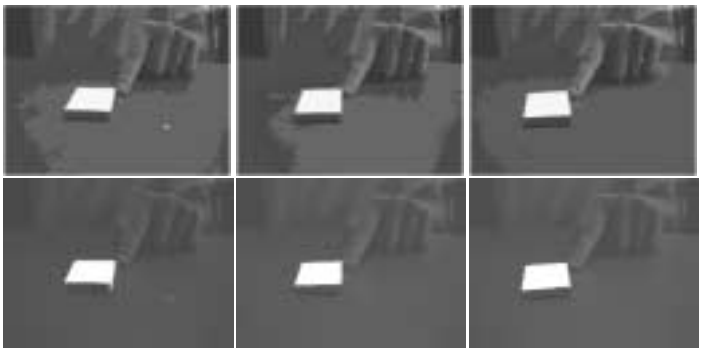

Figure 6: First line: Images of the box from one of the cameras. Second line: superposition of real images and the synthetic box, using final parameters

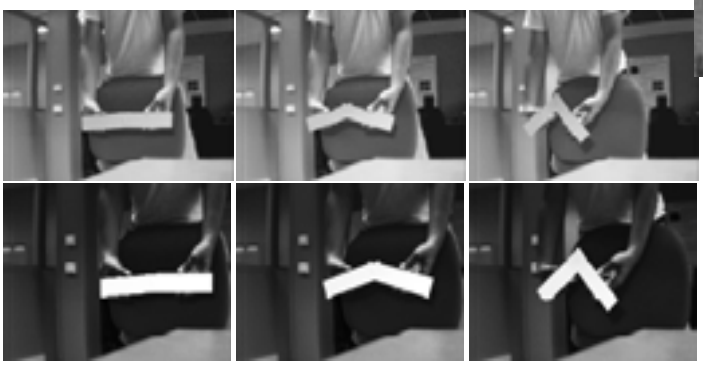

Figure 7: First line: Images of the two-pieces object from one of the cameras. Second line: superposition of real images and the synthetic object, using final parameters

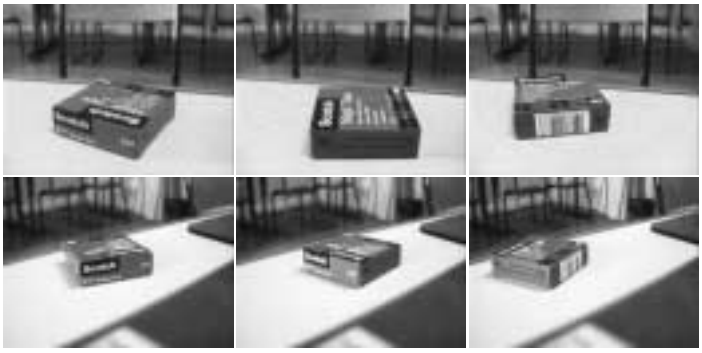

Figure 8: Samples of the two cameras at the beginning, the middle and the end of the sequence.

the box, without textures. We search an initial parameters vector, has described in section 3.3. Then the program refines this initial solution, giving the parameters values at the initial step (see Fig.9). We then apply the texture auto-extraction, which collect for each part of the model the textures in each view (if available). These textures are combined (see Fig.10) and are applied to the object model, as seen in Fig.11). Finally, parameters tracking is achieved with this improved model. Results for the sequences can be seen in the figure

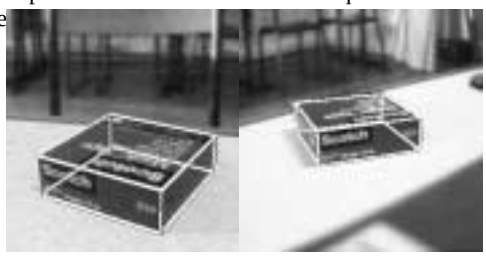

Figure 9: Synthetic box superposed to the real one. To be more visible, the synthetic box is rendered with lines.

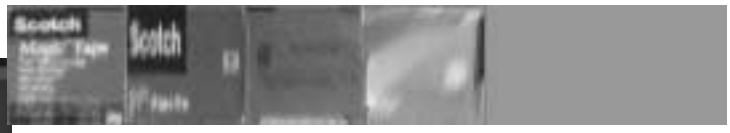

Figure 10: Textures generated by the program. The lasts are full green because the corresponding parts of the subject are not visible in the images. The default face color is then used.

Figure 11: The synthetic box, rendered with texture-mapping using extracted textures.

5 Conclusions

In this paper we have described an approach to perform parameters tracking of subjects in video sequences. This approach, based on synthesis / analysis collaboration, has shown its ability to treat parameters tracking with a good precision. Main features of this approach are:

- occlusion robustness, by the use of dense comparisons for the similarity measure;
- parameters definition flexibility. Many subject characteristics can be used has parameters such as sizes, colors... This allows an auto-refinement of the geometric model;
- texture extraction which permit to grab and update the subject textures, in order to improve the model fidelity or for study purpose.

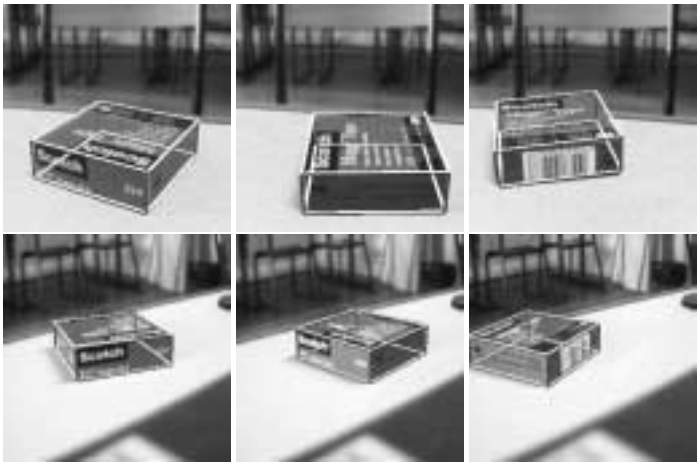

Figure 12: Superposition of the real and synthetic box along the sequence.

We currently test our method with more complex subjects. We are working on hand tracking, which correspond to thirty liberty degrees, and around fifty internal parameters such as length, sizes *etc*. (see Fig. 13). However, some points need to be improved in

Figure 13: An image of the hand and the actual adjusted model.

the future.
The first point is the computation time – around height minutes per frame for a two-cameras sequence – doesn't allow real-time treatments. Two complementary approaches are being tested: parallelization of synthesis rendering and images comparison, and multi-resolution for treated images. This last approach consist in using low resolution images, giving fast but less precise results, and to increase resolution at the final steps of the optimization algorithm to obtain higher precision.
We are also improving the auto-texture extraction part, in order to obtain a more automated model improvement.
In the near future we will axe our efforts on parameters analyze during treatments. A "on-the-fly" computation of parameters effects may be use to guide treatments priorities, and to exploit dependences between parameters.

References

[1] R. Basri and D. Weinshall. Distance metric between 3d models and 2d images for recognition and classification. *IEEE Transactions on Pattern Analysis and Machine Intelligence 18(4)*, pages 465–470, 1996.

[2] D. DeCarlo and D. Metaxas. Optical flow constraints on deformable models with applications to face tracking. *University of Pennsylvania technical report MS-CIS-97-23 (to appear in IJCV 2000)*.

[3] D.E. Goldberg. *Genetic Algorithms in Search, Optimisation and Machine Learning*. Addisson-wesley books edition, 1994.

[4] J.Isodoro and S.Sclaroff. Active voodoo dolls: A vision based input device for nonrigid control. *Proc. Computer Animation*, June 1998.

[5] M. Kass, A. Witkin, and D. Terzopoulos. Snakes: Active contour models. *International Journal of Computer Vision*, 1(4):321–331, 1988.

[6] J.A. Nedler and R. Mead. A simplex method for function minimisation. *The Computer Journal 7*, pages 308–313, Jully 1965.

[7] Y. Perret, T. Excoffier, and S. Bouakaz. Parameters matching of objects in video sequences. *In Proceedings of SPIE, Three-Dimensional Image Capture and Applications*, 3958, January 2000.

[8] R. Polana and R. C. Nelson. Recognition of motion from temporal texture. *Proc. IEEE Conference on Computer Vision and Pattern Recognition, Champaign, Illinois*, pages 129–134, June 1992.

[9] William K. Pratt. *Digital Image Processing*, chapter 2 and 3, page 21 to 89. Wiley Interscience, 2nd edition, 1991.

[10] S. Santini and R. Jain. Similarity matching. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 1996.

[11] S.Yonemoto, N.Tsuruta, and R.Taniguchi. Tracking of 3d multi-part objects using multiple viewpoint time-varying sequences. In IEEE Computer Society, editor, *Proc. International Conference on Pattern Recognition 14(1)*, pages 490–494, August 1998.

[12] T.Molet, A.Aubel, D.Thalmann, and et al. Anyone for tennis? *Presence 8(2)*, pages 140–156, April 1999.

[13] T.Molte, R.Boulic, and D.Thalmann. A real time anatomical converter for human motion capture. *International Conference on Computer Vision, Bombay, India*, pages 79–94, 1996.