INSTITUTO DE COMPUTAÇÃO
UNIVERSIDADE ESTADUAL DE CAMPINAS

Accurate Tridimensional Reconstruction with Unsynchronized Cameras Regardless of Time Information

F. Dias *N. J. Leite*

Technical Report - IC-14-04 - Relatório Técnico

February - 2014 - Fevereiro

Accurate Tridimensional Reconstruction with Unsynchronized Cameras Regardless of Time Information

Fábio Dias* Neucimar Jerônimo Leite†

Abstract

In this work, we approach the problem of the tridimensional reconstruction of the trajectory of an object using unsynchronized cameras. While most methods from the literature try to measure, as accurately as possible, the timing difference between the cameras, we chose a very different path: we ignore most, or all, of the time information from the videos. The idea is quite simple. Each pair of camera and projected trajectory generate a surface on the volume. The intersection of these curves is the desired result. Since this intersection can be very complex, we considered a Monte Carlo approach for the reconstruction, using random tridimensional points to estimate the region of intersection. Therefore, any camera calibration schema can be used. These points are later used to compute a continuous curve, which is the final result of the method. We compared this method to a very simple reconstruction approach, which assumes the frames are synchronized, and obtained outstanding results. Our implementation and data are freely available on `https://code.google.com/p/ucr-timeless/`.

1 Introduction

With the popularization of video cameras, their use as a data acquisition tool is increasing on many domains. A very common task is the tridimensional reconstruction of the trajectory of an object of interest through a captured scene. Assuming some hypothesis to be true, this is indeed an easy problem to solve. In this work, we will explore the removal of one of these hypothesis: the need for synchronized cameras for tridimensional reconstruction.

In this work, we consider a camera to be synchronized if it is part of a multi-camera setup and it is connected to a synchronization signal, assuring that all cameras on the setup are capturing the frames at the same time. This method of synchronization by hardware is usually recommended if the application needs good accuracy.

While such synchronization inputs are a basic feature on industrial and professional cameras, they are not usually present in over-the-shelf digital cameras, so the synchronization must be achieved in a different way.

*Faculdade de Educação Física, Universidade Estadual de Campinas, 13083-851, Campinas, SP. FAPESP 2012/15811-1

†Instituto de Computação, Universidade Estadual de Campinas, 13081-970 Campinas, SP.

The most common way to deal with unsynchronized videos is to estimate the time difference between the involved cameras, usually bundled with the camera calibration itself [14, 2, 15, 13, 16, 7, 1]. Another common method to compute this time difference involves the use of an external audio signal [3, 5]. Other methods resort to the frequency domain [9, 10]. Once the time difference is estimated, images [17, 12] or data [8] from one camera can be interpolated to match a reference camera, used to help space carving [6] or compute the scene flow [11]. Elhayek *et. al.* [4] avoid the time difference computation by developing a new tracking method in a continuum domain, however we are not interested in modifying the tracking procedure, only the reconstruction.

In this work, we approach the unsynchronized tridimensional reconstruction of the trajectory of a single point by ignoring the time information in most, or all, cameras. To the best of our knowledge, our approach is original and fundamentally different from the ones in the literature. Indeed, if the required result can be expressed only as a curve in the tridimensional space, we can completely ignore the temporal information. However, our approach does require that all the videos encompass the same period of time. While our results are promising, the main contribution of this work is this new way of considering the problem.

2 PROPOSED METHOD

We start by describing the problem we aim to explore. The observed scene is composed of a single, punctual, object moving around freely. The scene is captured by a number of calibrated cameras and the coordinates of the point are known for all frames of each video. All the cameras start and stop recording at, roughly, the same time. Approximate boundaries of the movement, in real world coordinates, are also known.

The main idea behind our approach is quite simple. Consider a long exposure camera. All the movement observed is present in a single image. This image represents projections of all the places the object has been, but it does not contain the information of *when* it was there nor its tridimensional position. Each pair of camera and trajectory generate a surface on the volume of interest. The intersection of these surfaces is our result. A very simple graphical illustration of this principle is depicted in figure 1, where two cameras are depicted, along with the respective surfaces and their intersection.

Since the intersection of these surfaces can be reasonably complex, a closed form solution is not practical, in most cases. Therefore, we considered a *Monte Carlo* approach to this problem, using random points to estimate the trajectory of the observed object. Algorithm 1 presents an outline of the base procedure.

The basic idea of the algorithm is to create a number of random points in the volume. Each point is verified and rejected if it is deemed unacceptable. New random points are created and the iteration continues.

The definition of what makes a point acceptable is flexible. In our case, the point is projected, using the calibration information, by each camera and the distance from its projection to the corresponding trajectory is calculated. To improve the accuracy, we interpolate the bidimensional trajectory using a spline. If this distance is smaller than a

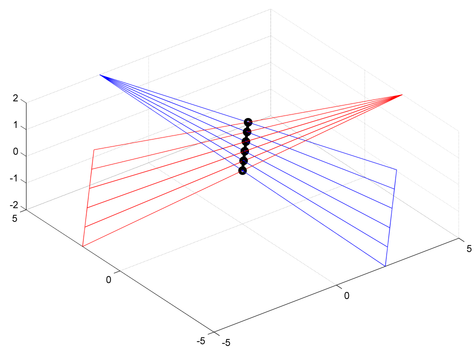


Figure 1: Illustration of the method's principle.

Data:
ϵ: Vector of rejection parameters;
N_p: Number of points to be used at each iteration;
V: Approximate volume of movement;
C: Number of cameras;
$Calib$: Structure containing calibration information for all cameras;
U: Structure containing the position of the object in the images for all cameras;
Result:
T:Tridimensional trajectory of the object;

```
points=[];
for e ∈ ϵ do
    points = points ⋃ CreateRandomPoints(V, N_p);
    for p ∈ points do
        for c ∈ {1..C} do
            u_p = ProjectPoint(Calib(c), p);
            if not Acceptable(p,Calib(c),U(c),e) then
                RejectPoint(p);
            end
        end
    end
end
```

Algorithm 1: Base algorithm

parameter ϵ, for all cameras, the point is preserved, otherwise it is discarded. Then we reduce the value of the parameter ϵ and the process is repeated. These iterations stop after a minimum value for ϵ is considered.

If the result is required to be a function of time, one of the cameras can be used as reference. In this case, we associate each random point with the value of the parameter of the closest point of the spline, which corresponds to the time of the closest sample. Note that, even in this case, the time information of the other cameras is completely ignored.

Finally, the random points that are preserved after the last iteration are used to estimate the tridimensional trajectory of the object. We considered a smoothing spline, parametrized by the time of the reference camera, but similar methods can be used as well. We evaluated this spline for the integer values corresponding to the frames of the reference camera, leading to the desired tridimensional result.

A very interesting feature of our method is that since the involved operations always project tridimensional points of the scene into bidimensional points on the images, we do not back-project points or compute the inverse of the lens distortion model. Any distortion model can be used. In fact, any calibration method can be used, even neural networks or other closed methods.

2.1 Implementational Considerations

The base algorithm we presented on the previous section has inherent problems. One of them is a well known problem for space carving reconstruction, the presence of false positives in the reconstruction results. Similarly to space carving, our method works by removing regions of the space, leaving only the plausible answer, but all other plausible points are accepted. One easy way to lessen this issue is to divide the video into separated segments, and to run the reconstruction on these segments separately, as long as each segment covers roughly the same period of time. Proper camera positioning is also a key factor for this problem.

Another interesting issue is convergence. Due to the randomness nature of the method, we have a compromise between its accuracy and the time necessary to process all points. Thus, a set of too few points may not properly fall in our region of interest, while too many points can significantly increase computational cost.

In our implementation, we tried to keep the number of points near a certain control parameter N_P. If too many points were present, we removed random points until the total of them was equal to this parameter.

The introduction of new points is also relevant to the final result. We considered the introduction of new points in three different ways:

- Linear combinations of existing points: We add points from the line formed by two random points.
- Random points around the existing points: We create points around existing ones, using a normal distribution.
- Random points: we introduce new points by considering an uniform distribution in the volume of interest. At the start, this volume is the whole possible region of

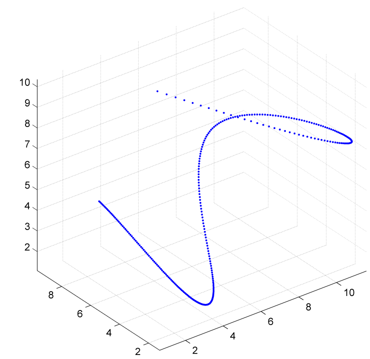


Figure 2: Position of the object in our experiment.

the movement. In the refinement iterations, it is defined by the mean and standard deviation of the existing points. We create new points until we reach our control parameter concerning the corresponding number of points.

To improve the point coverage, we introduce another parameter, an integer N_{min}. For each iteration, we keep generating points until we have at least N_{min} acceptable points.

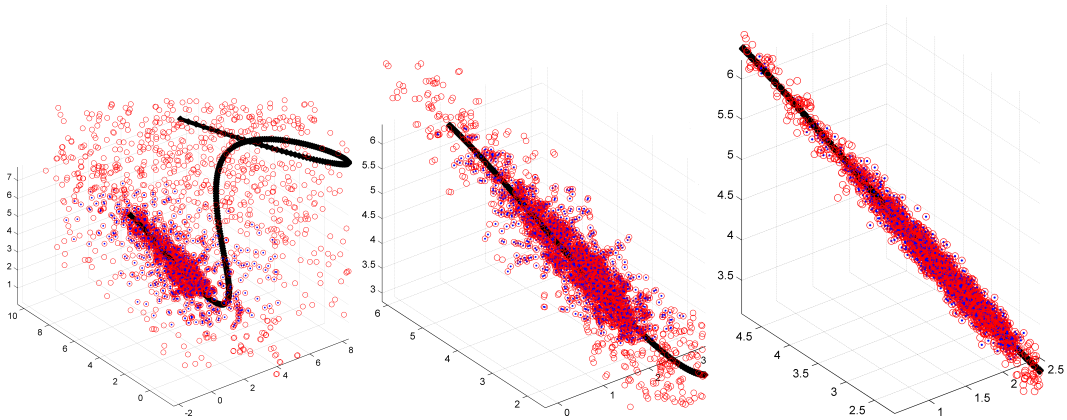
3 EXPERIMENTAL RESULTS

In this section, we evaluate our proposed method using only synthetic data, so the errors we find are not caused by calibration or quantization, but due to the imposed synchronization errors.

We created an artificial dataset containing 300 trajectory points, interpolated from 6 random points. The trajectory of the object is depicted in figure 2. We then placed three cameras, at coordinates $c_1 = (30, 5, 20)$, $c_2 = (5, 30, 20)$ and $c_3 = (-20, -20, 20)$. However, to ensure that the samples are not synchronized, each camera captured only 100 frames.

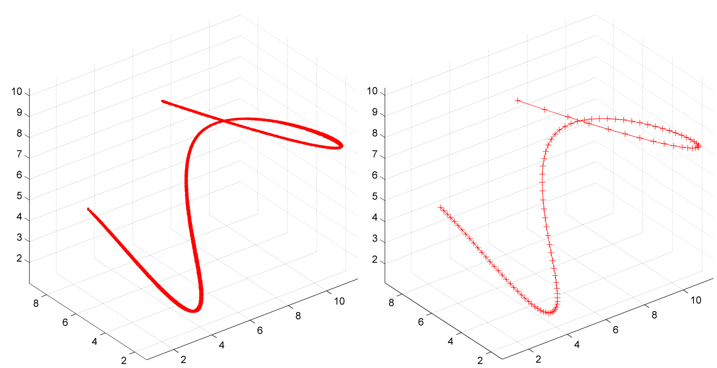
For this experiment, we divided the trajectory in twenty segments. To improve the results, in between segments, we also processed the last half of the previous segment, combined with the first half of the next segment. This approach improves the results because our method usually focus the random points in the middle of the segment, leaving holes in the coverage. The parameter ϵ was set to the following values: $[10, 5, 2, 1, 0.5]$. The random and acceptable points, for the first segment, are illustrated on figure 3 for three different values of ϵ, to demonstrate how the points converge to the desired result, depicted as a black line. The random points are depicted as red circles, while the accepted points are depicted as blue dots.

The control parameter for the number of points, N_p was $2,000$. In each iteration, we required at least $N_{min} = 500$ points. The resulting points and the smoothing spline are depicted in figure 4. Since we have the original data, we can compute the reconstruction error. The mean error found was 0.0108, with standard deviation of 0.0140. The error graph for our reconstruction method is depicted on figure 5.



(a) Points for $\epsilon = 10$. (b) Points for $\epsilon = 2$. (c) Points for $\epsilon = 0.5$.

Figure 3: Random and acceptable points for the first segment.



(a) Points found. (b) Smoothing spline from the points.

Figure 4: Results of our method using three cameras.

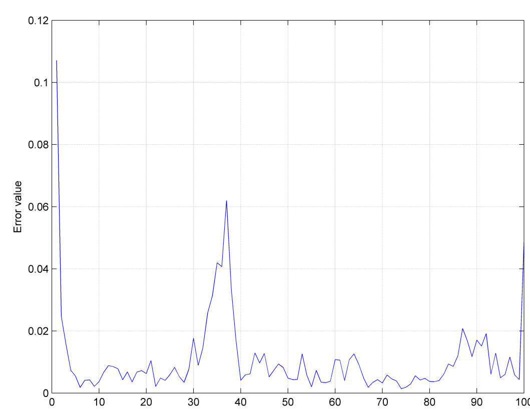


Figure 5: Error graph for our method

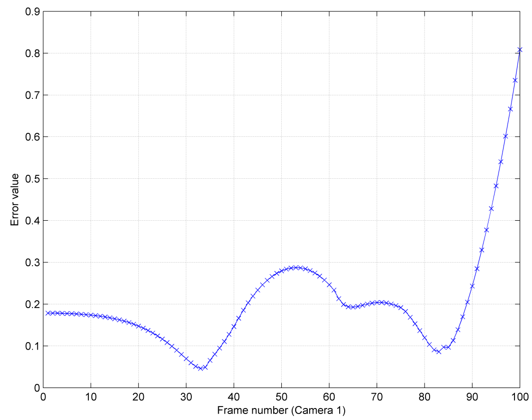


Figure 6: Naive reconstruction error [see text].

For comparison purposes, we also computed a naive reconstruction, assuming that frames of same number correspond to the same time. While this should perform worse than most of the methods from the literature, it provides an estimate of the involved error. The mean error found was 0.2078 with standard deviation 0.1332, using the first camera as time reference. The error graph is depicted in figure 6.

As we can see from the error values and graphs, our method greatly outperforms the naive reconstruction, which is expected, since it falsely assumes that the frames are synchronized. However, further investigation is needed to properly assess its performance against better methods from the literature. Our method also needs to better address the point coverage in the extremes of the segments, as we can see from the error graph, where we have an error peak.

Previously, we stated that proper camera position is a key factor to good results. To illustrate this point, we removed the third camera from the setup used previously. To increase the error further, we doubled the segment size, dividing the sequence into ten segments, instead of twenty. The resulting points of the first segment is illustrated, along with the original trajectory, on figure 7.

The effects of such outliers can probably be reduced by considering a filtering step before the final result, since they generate a very distinguished discontinuity, but this was not considered here. However, by dividing the trajectory into small segments, both their occurrence and magnitude can be reduced.

4 CONCLUSIONS

In this work we presented a different approach to the tridimensional reconstruction of a punctual object which does not require the computation of time differences between the cameras, can be used with any camera calibration structure and can be easily used for many cameras at once, including cameras with very different frame rates. Additionally, our method can generate a *continuous curve* that can be evaluated at any required interval. While our experimental results are promising, the main contribution of this work is the

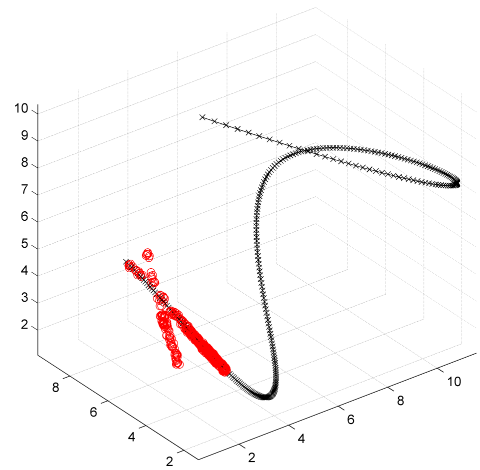


Figure 7: Example of false positives [see text].

difference in the approach. To the best of our knowledge, this method is not only original but quite different from the methods found in the literature.

There are several factors to be considered in this method, such as the optimal camera positioning and the limits of several control parameters. The last step of our method, when we use the resulting points to generate a spline, can also be improved to reject the kind of outliers we demonstrated on the experimental section. This is indeed a work in progress, but it is our opinion that this approach is quite interesting and should, with more work, yield excellent results.

Our implementation and data are freely available on `https://code.google.com/p/ucr-timeless/`. We invite the readers that wish to replicate, or further investigate, our results to refer to them.

References

[1] D. Bradley, B. Atcheson, I. Ihrke, and W. Heidrich. Synchronization and rolling shutter compensation for consumer video camera arrays. In *Computer Vision and Pattern Recognition Workshops, 2009. CVPR Workshops 2009. IEEE Computer Society Conference on*, pages 1–8. IEEE, 2009.

[2] R. L. Carceroni, F. L. Pádua, G. A. Santos, and K. N. Kutulakos. Linear sequence-to-sequence alignment. In *Computer Vision and Pattern Recognition, 2004. CVPR 2004. Proceedings of the 2004 IEEE Computer Society Conference on*, volume 1, pages I–746. IEEE, 2004.

[3] R. M. L. de Barros, T. G. Russomanno, R. Brenzikofer, and P. J. Figueroa. A method to synchronise video cameras using the audio band. *Journal of Biomechanics*, 39(4):776 – 780, 2006.

[4] A. Elhayek, C. Stoll, N. Hasler, K. I. Kim, H. Seidel, and C. Theobalt. Spatio-temporal motion tracking with unsynchronized cameras. In *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*, pages 1870–1877, 2012.

[5] N. Hasler, B. Rosenhahn, T. Thormahlen, M. Wand, J. Gall, and H.-P. Seidel. Markerless motion capture with unsynchronized moving cameras. In *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, pages 224–231. IEEE, 2009.

[6] T. A. Haufmann, A. R. Brodtkorb, A. Berge, and A. Kim. Real-time online camera synchronization for volume carving on gpu. In *Advanced Video and Signal Based Surveillance (AVSS), 2013 10th IEEE International Conference on*, pages 288–293, 2013.

[7] C. Lei and Y.-H. Yang. Tri-focal tensor-based multiple video synchronization with subframe optimization. *Image Processing, IEEE Transactions on*, 15(9):2473–2480, 2006.

[8] A. Masiero and A. Cenedese. A kalman filter approach for the synchronization of motion capture systems. In *Decision and Control (CDC), 2012 IEEE 51st Annual Conference on*, pages 2028–2033, 2012.

[9] H. Matsumoto, J. Sato, and F. Sakaue. Multiview constraints in frequency space and camera calibration from unsynchronized images. In *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*, pages 1601–1608, 2010.

[10] S. Miyan and J. Sato. Reconstructing sequential patterns without knowing image correspondences. In K. Lee, Y. Matsushita, J. Rehg, and Z. Hu, editors, *Computer Vision – ACCV 2012*, volume 7727 of *Lecture Notes in Computer Science*, pages 484–496. Springer Berlin Heidelberg, 2013.

[11] A. Sellent, K. Ruhl, and M. Magnor. A loop-consistency measure for dense correspondences in multi-view video. *Image and Vision Computing*, 30(9):641 – 654, 2012.

[12] S. Shankar, J. Lasenby, and A. Kokaram. Warping trajectories for video synchronization. In *Proceedings of the 4th ACM/IEEE international workshop on Analysis and retrieval of tracked events and motion in imagery stream*, ARTEMIS '13, pages 41–48, New York, NY, USA, 2013. ACM.

[13] S. N. Sinha and M. Pollefeys. Visual-hull reconstruction from uncalibrated and unsynchronized video streams. In *3D Data Processing, Visualization and Transmission, 2004. 3DPVT 2004. Proceedings. 2nd International Symposium on*, pages 349–356. IEEE, 2004.

[14] G. Stein. Tracking from multiple view points: Self-calibration of space and time. In *Computer Vision and Pattern Recognition, 1999. IEEE Computer Society Conference on.*, volume 1, pages –527 Vol. 1, 1999.

[15] P. Tresadern and I. Reid. Uncalibrated and unsynchronized human motion capture: a stereo factorization approach. In *Computer Vision and Pattern Recognition, 2004. CVPR 2004. Proceedings of the 2004 IEEE Computer Society Conference on*, volume 1, pages I–128. IEEE, 2004.

[16] J. Xiao and T. Kanade. Uncalibrated perspective reconstruction of deformable structures. In *Computer Vision, 2005. ICCV 2005. Tenth IEEE International Conference on*, volume 2, pages 1075–1082. IEEE, 2005.

[17] C. Zhou and H. Tao. Dynamic depth recovery from unsynchronized video streams. In *Computer Vision and Pattern Recognition, 2003. Proceedings. 2003 IEEE Computer Society Conference on*, volume 2, pages II–351. IEEE, 2003.