ISSN:1575-6807
D.L.: GC-1317-1999

INSTITUTO UNIVERSITARIO DE CIENCIAS Y TECNOLOGIAS CIBERNETICAS

IWCVIA'03: International WorkShop on Computer Vision and Image Analysis

Editor: Luis Alvarez

Nº 0026

December 4-6, 2003

Cuadernos del Instituto Universitario de Ciencias y Tecnologías Cibernéticas
Instituto Universitario de Ciencias y Tecnologías Cibernéticas
Universidad de Las Palmas de Gran Canaria
Campus de Tafira
35017 Las Palmas, España
http://www.iuctc.ulpgc.es

AMICam: A Video Processing Software for Combining Real and Virtual Scenes

L. Alvarez, C. Cuenca, A. Salgado and J. Sanchez
Departamento de Informática y Sistemas
Universidad de Las Palmas de Gran Canaria
Campus Universitario de Tafira, 35017 Las Palmas, SPAIN

Abstract

In this paper, we present a brief description of a master thesis done by Agustín Salgado in the computer science department of Las Palmas University. The master thesis deals with the problem of inclusion of $3-D$ virtual objects in a real video sequence. We do not introduce any new real relevant contribution in the field, we focus our attention in the hole problem and we try to provide a solution for the different tasks we have to deal with. In particular, we study techniques for characteristic point extraction, tracking, multiple camera calibration, synchronization of real and virtual cameras and the rendering of virtual objects in the real video sequence. Finally, we present the experimental results obtained. In particular, we insert some $3-D$ virtual objects in a real video sequence and we show the result obtained and the problems we have noticed.

1. Introduction

The digital image technology has experimented an important increase for the last years, mainly, due to the increase of computers performance and the digital video. This technology allows the combination of synthetic objects and real scenes. We can add virtual objects to real video sequences in such a manner as to appear part of the 3D world. This technology can be applied in many fields, like movies or advertisement. This is the context of the master thesis we present here.

In this work, we use a real video sequence recorded with a video camera which has been moved thought a 3D scene. We want to include one or various virtual 3D objects. After that, a new video sequence will be created with the original sequence and the 3D objects. The final video sequence will be equivalent that the virtual objects would have been in the scene when the video sequence was recorded.

The organisation of the paper is as follows: In section 2, we introduce the Harris corner detector technique that we use to extract characteristic points in an image. In section 3, we study the problem of tracking the characteristic points, obtained in each image by the Harris detector, across the video sequence. In section 4, we present an overview of multiple camera calibration techniques. In section 5, we study the render process to generate new video sequences from the inclusion of virtual 3D objects in a real video sequence. In section 6, we present a video processing software for combining real and virtual scenes. Finally, in section 7 and 8, we present the experimental results and the main conclusions of the paper.

2. Harris corner detector

An image contains information about a scene. However, only a few of this information let's us to understand the scene. We will try to extract that information that help us to know the camera movement. A corner is a kind of information it presents in many images and it is useful for the camera calibration.

In this section, we focus in the Harris corner detector algorithm develop by AMI group. This corner detector has two important features, it is very fast and the corners information is precise. This implementation can estimate the corner position in subpixel precision by interpolating the Harris values and computing the maxima of some interpolated function.

We tested the Harris detector on real images. The number of corners detected depends directly of the three parameters of Harris detector. The behaviour of this corner detector is determined by three parameters, which are described as follows:

- *Harris radio.* It shows the minimum distance (in pixels) allowed between two corners. When the detection process finishes, the best corners are chosen by spacial location.
- *Sigma.* It shows the standard deviation of the Gaussian. It is used to balance the derivates values around a pixel.
- *Threshold.* It is the Harris value threshold used to choose the pixels which are corners. It affects directly the number of corners detected.

When we select the value of the Harris detector parameters, we must be a deal, between the number of corners

detected and the total computational time. Restricted values reduce the computational time. If the number of corners is high, it will affect to the total computational time of the whole video creation process. The experiment results have shown that the corner method has good detection and excellent location performance.

3. Tracking sequences of corresponding singular points across the video sequence

In this section, we describe the tracking process. This process computes automatically sequences of corresponding singular points across the video sequence. The path of a 3D point is a list of projection points in following frames, where that 3D point is shown.

3.1. Succession criterias

The tracking computes sequences of corresponding singular points across the video sequence. The corners detected are local information in each frame. So, we must try to relation the corners in following frames. The tracking process needs some criterias to know when two corners are in correspondence. We define two criterias: Harris and Correlation test. These criterias are the tracking "intelligence".

The Harris detector gives us for each corner, its location (x, y) in subpixel precision and its Harris value. If two corners, placed in following frames, correspond to the same 3D point they will have similar Harris values. The Harris test compares the Harris values of two corners. The second test, correlation test, computes the correlation between two windows centred in two corners. These tests are passed when the comparation and correlation results are less than a threshold.

3.2. Tracking process description

The tracking is an iterative process. Its aim is to compute the path of the singular points across a video sequence. This process is divided into the following steps:

Step 1:

In the first step we extract the singular points from all frames of the video sequence. We use the Harris detector described in the previous section.

Step 2:

It is a simplification of the tracking process. We have only two frames and we want to know the correspondence of a set projections in these frames. If we want to extrapolate this solution to the rest of the sequence, we consider in the next iteration, the second frame first one, and the following (to that second frame) second one.

Now, for each corner, we try to find its predecessor point in the previous frame.

1. Looking for predecessor.

   Fixed a singular point, we look for its predecessor point into a window NxN, centred in the corner location, in the previous frame. We will go to the next step, only if we find a candidate predecessor point. Otherwise, we consider this singular point the beginning of a new sequence.

2. Apply the Harris and correlation test.

   In this step, we apply the succession criterias. If the two tests are passed, we consider these singular points in correspondence. When one of these tests is not passed, we came back the previous step and the search continues.

Corre Har	**15%**	**30%**	**50%**
5%	363.825	332.989	311.059
10%	345.129	300.762	266.779
20%	330.398	275.143	230.590

Table 1: Number of sequences obtained.

3.3. Tracking process experimental results

In this part, we examine the performance of the tracking process. We tested the tracking process on a real video sequence of 171 frames, where an average 1200 corners per frame was detected. This process was tested and compared on their basis tests: *Harris* and *Correlation*. In figure 1, we can see the graphical results obtained. In table 1, we can see the results obtained when we change the thresholds values for the Harris and Correlation test.

The Harris test avoids to apply the Correlation to non-correspondence corners. So, it reduces the compute time. However, the thresholds must not be restricted, because the sequences length will be short. Short sequences are not useful for the camera calibration.

With the same way it happened in the previous section, we have to get a deal between restricted and tolerated threshold values, to obtain good results in short time.

4. Camera Calibration

The problem of multiple camera calibration consists in recovering the camera positions and orientations with respect to the world coordinate system, using as input data tokens,

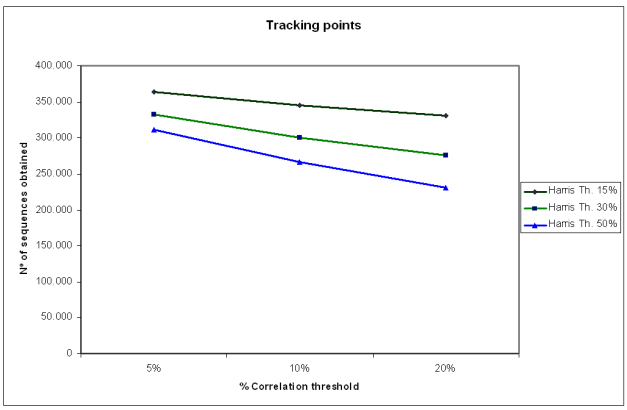

Figure 1: Graphical results of the table 1. It shows the numbers of sequences obtained when we modify the Harris and Correlation thresholds.

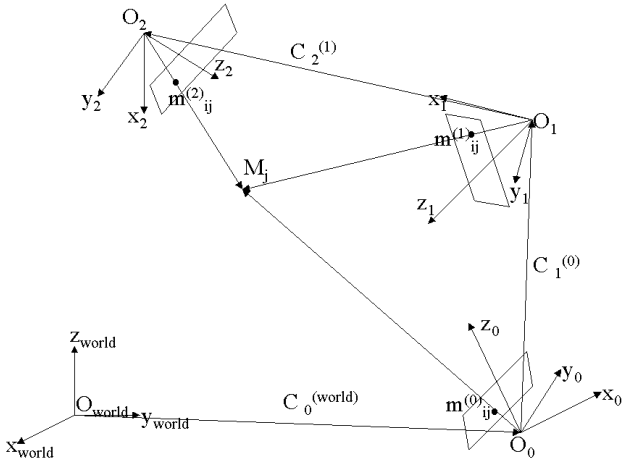

Figure 2: Motion parameters derived from point matches.

such as pixels or lines, in correspondence in different images. Figure 2 shows this scenario for a system with three cameras.

The specification of the i-th camera position is the 3D point $C_i^{(world)}$, where the superscript is the reference system in which the magnitude is expressed. The orientation specification is a rotation matrix $R_i^{(world)}$ or any equivalent representation, such as quaternions or Euler angles.

When the image tokens in correspondence are projections of a set of 3D points $\{M_j\}_{j=1..N}$ where N is the number of points, it is possible to reconstruct each 3D point expressed in the world coordinate system by simply estimating the intersection point of the line set:

$$\left\{ r_i \equiv C_i^{(world)} + \lambda \overrightarrow{C_i^{(world)} R_i^{(world)} m_{ij}^{(i)}} \right\}_{i=1..N}$$

where $C_i^{(world)}$ are the coordinates of the optical center in the world reference system, and $m_{ij}^{(i)}$ are the coordinates of the projection of M_j in the normalised reference system for the i-th camera. A reference system is said to be normalised when the optical center is in the origin, the focal distance is 1 and the pixel is a square of size 1. We will assume that the intrinsic parameters of the cameras are known, which allows us to normalise the reference system.

In order to estimate the intersection 3D point of the line set it is necessary to know the position of the optical center and the rotation matrix for each one of the cameras. The computation of these parameters solves the problem of the multiple camera calibration. After estimating these parameters, we can evaluate the accuracy of the solution by projecting the reconstructed 3D points in each camera, and the best solution for the calibration problem is the one that minimises the energy function:

$$f(C_0^{(world)}, C_1^{(world)} \quad , \quad ..., R_0^{(world)}, R_1^{(world)} ...) = \sum_{i,j} \left\| m_{ij}^{\prime(i)} - m_{ij}^{(i)} \right\|^2$$

where $m_{ij}^{\prime(i)}$ is the projection of the reconstructed point M_j in the i-th camera.

To study the problem of camera calibration, we will use the classical "pinhole model" which assumes the simplest projective model for the camera image acquisition. To calibrate the camera means to find out the parameters which determine the way that the projection works. There are two types of parameters: intrinsic and extrinsic parameters. The extrinsic parameters determine the $3-D$ location of the camera in the scene with respect to some *a priori* fixed reference system. They are a translation vector and rotation matrix. The intrinsic parameters do not depend on the camera location in the $3-D$ scene. They are dependent of the camera, and are the focal length, the pixel size and the focus position. A camera is defined by the projection matrix. This matrix determines the projection from the 3D space (world points) into 2D space (projection plane).

The calibration algorithm returns one projection matrix for each frame. We use this information to position the camera into the scene.

5. Inclusion a virtual object in the video sequence, render and creation of a new video sequence

In this section, we describe the inclusion of virtual $3-D$ objects process into a scene and the making of a new video sequence. We have two types of cameras, a real and other

artificial, designed for solving problems completely different. Our aim is the projections of both cameras will be identical. If you get to fit the projections, the final video will be equivalent that the virtual objects would have been in the scene when the video sequence was recorded.

5.1. Inclusion a virtual object in the video sequence

To simule the inclusion a 3D virtual object in a real video sequence, we make a 3D world and the $3-D$ objects are included into it. In our virtual 3D world, the user can modify the $3-D$ objects, as their location as the transformations to apply them. To complete our 3D world we put in the background a frame (extracted from the video sequence).

To make the 3D world we use a graphical library, such as Open Inventor. Open Inventor is a library of objects and methods used to create interactive 3D graphics applications.

5.2. Render process

This process is automatically computed by Open Inventor render engine, so we avoid the complexity of this process.

5.3. Creation the new video sequence process

When the $3D$ objects are placed into the scene, we create the new video sequence with the virtual objects inserted. To keep the sense the objects are in the real-world scene, we have to put in the background the frame that is watched from the camera location. This process is divided into the following steps, which are executed for each frame:

1. We create a new frame, which size is the same that the original. We copy the original frame on a new one.
2. The camera is positioned in the scene, with the information extracted from the projection matrix of this frame. After that, Inventor renders our scene (only objects) and its output is stored in a buffer.
3. Finally, we join the frame (background) and the render output. We overwrite on the frame those pixels that correspond with the objects projection.

5.4. Experimental results

When the virtual objects are inserted in the scene, we assume they stay static. This master thesis does not consider dynamic objects. To keep the static objects sense, the projections of the real and Inventor cameras must be identical. However, the real and Inventor cameras have been designed for solving problems completely different and they have some difference.

In one hand, the real camera has intrinsic and extrinsic parameters. The extrinsic parameters determine the $3-D$ location of the camera in the scene with respect to some *a priori* fixed reference system. The intrinsic parameters do not depend on the camera location in the $3-D$ scene. They are dependent of the camera, and are the focal length, the pixel size and the focus position. In the other hand, the Inventor camera projects objects in a 3D virtual scene. So, this camera has extrinsic parameters but it has not intrinsic parameters (or it has ideal intrinsic parameters).

Camera		(X_1, Y_1)	(X_2, Y_2)
1	P_{2D}	(163.50, 888.07)	(293.78, 987.6)
	P_{IV2D}	(163.25, 888.07)	(293.59, 987.6)
	Dif	(0.25, 0.0)	(0.19, 0.0)
50	P_{2D}	(336.63, 889.75)	(469.28, 988.31)
	P_{IV2D}	(336.47, 889.75)	(469.18, 988.31)
	Dif	(0.16, 0.0)	(0.10, 0.0)
100	P_{2D}	(551.16, 889.34)	(686.21, 983.86)
	P_{IV2D}	(551.10, 889.34)	(686.22, 983.86)
	Dif	(0.06, 0.0)	(-0.01, 0.0)

Table 2: Projection comparative of a 3D point in the real and Inventor cameras.

To patch the difference between the intrinsic parameters of both cameras, we assume in the real camera that the focus projection is in the center of the projection plane and the pixel is square (the same width and height). In table 2, we can see the experimental results obtained, when we placed two spheres in the corner location in the real-world.

6. AMICam

One of this master thesis aims was the design and building of an application. This application is called *AMICam*. AMICam is an application that offers to users tools for inclusion of $3-D$ objects in a video sequence. The tools are easy to use for any user.

Our application is divided in two parts: the interface and compute programs. The interface manages and stores the information the user inserts in the system. The compute programs are in a lower level, and they computes the most important process of this master thesis (tracking, camera calibration and render). An AMICam snapshot is shown in figure 3.

7. Experimental results. Inclusion of virtual objects in a real video sequence

One of the main applications of video sequence calibration is the inclusion of virtual $3-D$ objects in a real video sequence. We tested our methods in a real video sequence of

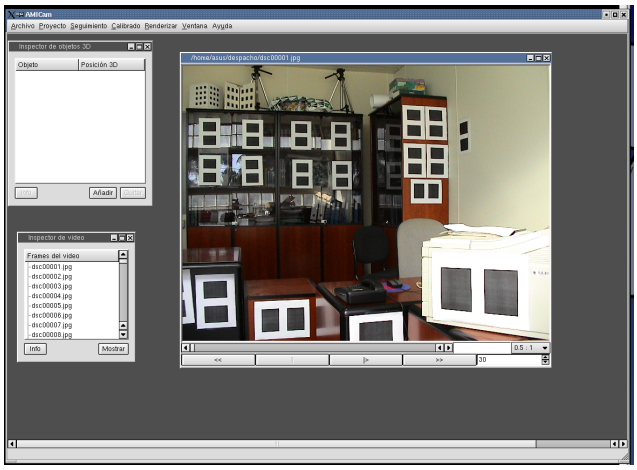

Figure 3: Main window of AMICam.

120 frames where we included eight artificial objects. The video sequence was taken with a digital camera. The camera zoom was constant and the camera was placed in the end of the office.

We have used AMICam to create the new video sequence with the virtual $3-D$ objects inserted. We placed in the office some white sheets with black squares printed on them. The black squares corners are easily detected by Harris corner detector. The points sequences detected by tracking process were long (average of 70 points per sequence, 120 frames). When the tracking finished, we selected the best sequences for the calibration process.

We have inserted eight virtual $3-D$ objects, in different planes and depth. In table 3 and 4, we present six frames of the real video sequence (left column) and the same frames with the inclusion of the virtual objects using the calibration parameters obtained (right column).

The final results are very well, the objects stay in their location. However, it exists an error between half and one pixel. This results could be improved if we use a nonlinear algorithm, like Levenberg-Maquard.

8. Conclusions

In this paper, we present a master thesis oriented to inclusion $3-D$ object in a video sequence. In this field, the techniques must be, efficiency, robustness, precise and flexibility. We have focus on reach these aims: (1) development and implementation of methods for inclusion $3-D$ objects in a video sequence and (2) development of a user interface that manages those methods (AMICam).

The final results, how we could see, were excellent and the range of application of the proposed methods are very wide.

References

[1] Faugeras, Olivier, *Three-Dimensional Computer Vision* The MIT Press. (1993).

[2] Richard Hartley and Andrew Zisserman, *Multiple View Geometry in Computer Vision* The MIT Press. (2001).

[3] Alvarez, Luis and Cuenca, Carmelo and Mazorra, Luis, *Signal Processing, Pattern Recognition and Applications. IASTED.* Morphological Corner Detector. Application to Camera Calibration. (2001).

[4] Roger S. Pressman, *Ingeniería del Software, quinta edición* McGraw-Hill Companies. (2001).

[5] Josie Wernecke, *The Inventor Mentor* Addison Wesley Publishing Company. (1994).