A System for Real-Time Multi-View 3D Reconstruction

Instituto Tecnológico de Informática

Abstract—A system for fast multi-view 3D model reconstruction of object sequences is composed of a number of hardware and software components: the multiple simultaneous image acquisition subsystems, the computation platform, the object/background segmentation algorithm, and in this case, a volumetric carving procedure based on the silhouettes of the objects from each view that generates a volumetric representation of the object under analysis. This volumetric representation can be then transformed into a 3D model applying isosurface generation techniques. However, as it will be shown, before any of these steps is performed a finely tuned camera calibration has to be obtained. The accuracy of this calibration is essential for the quality of the generated model. This paper describes the structure of a fast 3D reconstruction system, analyzes its features and focuses on the new techniques developed to enhance the calibration of its multiple cameras.

I. INTRODUCTION

Multi-view 3D reconstruction is based on the idea of using multiple views of an object to generate a 3D model of it. It requires either the use of multiple cameras or a static object and a single camera acquiring images from different points of view.

The system we propose is designed for near real-time operation: it should be able to generate 3D models of objects sequentially with a small latency of few seconds between them. To this end, we have designed a multi-view system composed of 16 synchronized cameras evenly distributed in a 1 meter diameter sphere around a useful central ellipsoid of about 13 cm of polar (minor) diameter (due to the rectangular field of view of the cameras).

II. MULTI-VIEW 3D MODEL GENERATION

This section analyzes the multi-view 3D model generation in the proposed system from the software point of view. The process starts after the hardware triggers the synchronized image acquisition from all cameras. As data is being received, an image segmentation process is applied on them to discriminate the object from the background.

A. Image segmentation

Image segmentation is an important step where we need to decide the membership of a pixel, whether to object or to background. In order to reduce as much variability as possible, the proposed system has color panels as backgrounds. We take an image of the background without the object and another one when the object is present. However, a simple image subtraction operation does not work due to the occurrence of shadows in the different backgrounds.

With this a priori information of the task, the segmentation has been performed in the HSV color space [1] which rearranges the geometry of RGB in an attempt to be more perceptually relevant. By means of HSV it is possible to compare clean backgrounds and shadows on backgrounds and decide if they are similar enough not. For example a small variation in hue allows only hues around a given color (blue in our system) and bigger variations in value permit shadows to be considered as background colors.

As can be seen in figure 1(b), there is often noise in the border between the object and the background, this is removed by applying a smoothing algorithm. Then, a smoothing step is applied in order to remove the noisy areas and assure a smooth 3D reconstruction (see figure 1).

Figure 1. Steps of the image segmentation process.

B. Octree carving and isosurface generation

Once the 16 input images have been segmented, a projective volume carving process is performed using the object silhouettes from each camera. This has been implemented using an octree carving method based on [2] and [3] with some additional features like vertex data caches and integer-based coordinates for extra performance and precision.

After the carving, an isosurface algorithm is applied to generate a 3D model from the octree and the segmented silhouettes. In this case we have applied a modification of the Marching Cubes algorithm [4] based on fractal curves to eliminate all point replicas from without performing any geometric comparison. This feature notably speeds up the isosurface generation process and allows the isosurface vertices to be calculated only once.

Because of the sensibility of the carving process, a surface decimation step is performed to simplify the generated mesh eliminating too small and degenerated surface triangles. For this task we adapted the methods described in [5] and [6] to our context.
C. Texturing

Optionally, a projective texturing step can be performed to increase the 3D model visual quality. The methodology used is based on [2], enhanced by the effect of the previous surface decimation step.

Each triangle is divided in particles, equispaced points on the triangle surface, with a number of particles proportional to its area. A bilinear interpolation is applied to the vertex normals to associate normals to particles. Then, the particles are projected to the cameras that have a look vector nearly parallel to their normals and actually see the particle. An OpenGL-accelerated depth buffer is used to ensure this last condition. After this, these results are combined using the dot product of the normalized camera look vectors and particle normals as weighting factors. All the particle textures are then combined to form the triangle textures of all triangles in the 3D model. An example of a textured 3D model can be seen in figure 2.

![Textured 3D model](image)

Figure 2. Textured 3D model reconstructed using the proposed system.

III. MULTI-VIEW CAMERA CALIBRATION

Camera calibration is the process of finding a set of parameters which define a camera model from a set of world ↔ image point correspondences. These parameters also define the transformation processes between the world and image coordinate systems. Once this model has been computed, the camera can be used in different applications like, for example, 3D object tracking, augmented reality or immersive virtual reality.

Camera calibration is a mature field with a large amount of scientific literature available ([7], [8], [9], [10]). However, research is often focused in the calibration of a single camera, extending it to multiple cameras by repeating the process in as many cameras as needed. This approach indeed works but it can easily introduce errors in the calibration parameters when working in real world applications since cameras are being calibrated independently. Depending on the context where the cameras are being used these errors can have a strong impact in the results, which is indeed the case of our 3D reconstruction problem.

Due to the use of cone intersections during the carving step, a small deviation in the position and orientation of a camera can produce big distortions in the 3D reconstruction. The light gray region in figure 3 shows the effect of a misplaced camera $C_2'$ cutting the object right part of the box. This problem is also observed in the faceted ball of figure 6(a) where an incorrectly calibrated camera has removed the upper left part. Another clear example of this problem can be seen in figure 7(a) where the generated model becomes unconnected.

![Convex hull](image)

Figure 3. Convex hull for a good calibration ($C_1, C_2$) and a bad calibration ($C_1, C_2'$)

The problem of per-camera calibrations in multiple camera environments is basically a problem of inconsistencies in the world coordinate systems of each camera. As no global extrinsic calibration is performed, real-world environments are likely to produce small position and orientation errors that will move the world origin to a different position in each camera. To solve this problem we have developed a simple and efficient method to correct the extrinsic parameters calibration from independent calibrations.

All camera intrinsic parameters have been calibrated using the undistorted composite calibration method, also proposed by us in [11], by means of 5 non-parallel views of a planar point target for each camera. Extrinsic parameter calibration required a complex 3-dimensional target especially built for this purpose. This target is geometrically defined as the polyhedron with the position of the cameras as its vertices. It ensures the view of at least 95 points in 5 different faces from any camera when located in the center of the system. A view of this calibration target is shown in in figure 4.

However, the proposed correction method makes use of the theoretical positions and orientations of the cameras in the polyhedron geometry to avoid any previous extrinsic parameter calibration. As a consequence, as only previous intrinsic calibrations are required, the proposed method eliminates the dependency on the 3D target and all the problems derived from its manufacture, its imperfections and the segmentation and detection of its many calibration points.

IV. CORRECTIONS TO THE CALIBRATION OF 3D MULTI-VIEW ENVIRONMENTS

As previously stated, the problem of per-camera calibrations in multiple camera environments is a problem of inconsistencies
in the world coordinate systems of each camera. Suppose we have a point in the world that it is mapped somewhere inside the field of view of all cameras. Each camera will project the real 3D world point to a 2D point in the camera plane. However, as a consequence of the stated inconsistencies the real 3D coordinates of the point become quite useless as there are as many world point coordinates as cameras.

In spite of that, a geometrical mean value of the 3D point can be computed if there are at least two cameras. This point is defined as the world point that minimizes the distance to the nearest point in each of the camera projection rays. These camera projection rays are the ones going from each camera position through its 2D projection of the point in the camera plane, as in the usual pinhole-based camera model.

Let \( R_i = \{ r_i = P_i + tD_i \mid t \in \mathbb{R} \geq 0 \} \) be the camera projection rays of each \( i = 1 \ldots n \) cameras where \( P_i \) denotes the \( i \)-th camera position and \( D_i \) the direction vector from \( P_i \) to the corresponding 2D projected point \( p_i \). For the sake of simplicity, we can assume all \( D_i \) to be normalized.

Let \( p_w \) be the desired 3D point nearest to all camera projection rays, it can be defined as

\[
p_w = \arg\min_{p} \sum_{i=1}^{n} d(x, r_i)
\]

where the distance can be defined for general lines instead of rays as

\[
d(x, r)_i^2 = (x - P_i) \left[ I - D_iD_i^T \right] (x - P_i)
\]

A closed-form solution to \( p_w \) is proposed in [12]

\[
p_w = \left[ nI - \sum_{i=1}^{n} D_iD_i^T \right] \sum_{i=1}^{n} \left[ I - D_iD_i^T \right] P_i
\]

A simple real approach to this theoretical model would be to use spherical objects in different positions as world points, and set a measurement of its center in each camera projection as 2D coordinates.

With this, we proceed repeating iteratively the following algorithm.

Both the number of spherical objects and of iterations are free variables that must be empirically set. However, there are some mathematical requirements to consider. Camera extrinsic parameters are composed of both 3D position and 3D rotation: a total of 6 degrees of freedom. Hence, at least 6 objects should be used in different non-coplanar space locations to avoid underdetermined calibrations. Adequate values for both iterations and number of objects will be explored in the next section.

It is important to point out that this method requires an initial approximation of each camera position and orientation to work. In some cases like the described scenario it can completely remove the need of previous calibrations of camera extrinsic parameters. If this is not the case, a per-camera initial calibration, based on a common world coordinate system should be enough to start with.

V. EXPERIMENTAL RESULTS

We have applied the proposed method to the proposed system with good practical results. We decided to use a simple set of ping-pong balls as spherical objects and the center of mass of all the object pixels as its projection center. We also tried to fit a circle into the silhouette from the spherical projection and use its center, but no improvements were observed.

For the initial testing we used 79 ping-pong balls as points randomly distributed in the useful region of space (the intersection of the field of view of all cameras). Figure 5 shows the positions of the balls used in the experiments. We evaluated the calibration error using two metrics. First we used the mean reciprocal error described in [11] to measure the correctness of the per-camera calibrations. This metric is based on the distances between the expected and the transformed world ↔ image coordinates.

Second, we calculated the mean distance \( m \) from \( p_w \) to the rays \( R_i \). This can be defined as

\[
m = \frac{1}{n} \sum_{i=1}^{n} d(p_w, \tilde{r}_i)
\]

where \( \tilde{r}_i = P_i + ((P_i - p_w) \cdot D_i)D_i \) is the nearest point in \( R_i \) to \( p_w \). The larger this value is, the more inconsistent the world system origin coordinates are.

Surprisingly, only one iteration was enough to effectively eliminate all the inter-camera inconsistencies while the reciprocal error remained stable or decreased. We also detected a
strong tendency to move the cameras closer to their centroid as the number of iterations grows. Because of these reasons we decided to reduce the correction to only one iteration of algorithm 1.

From these results we decided to evaluate the minimum number of points required to obtain 3D models without the crudely chiseled effect problem. We found that using only 7 random points and one iteration led to better results. In fact, both error metrics used decreased with the mean reciprocal error nearly reduced to half and a slower tendency to move to the camera centroid.

Figures 6 and 7 show generated 3D models before and after the proposed corrections. Also, with the reduction of the problem size the runtime of the calibration correction dropped to only a small fraction of a second in a conventional CPU.

VI. CONCLUSIONS

We have presented a complete multi-view system for reconstructing 3D models from a number (16 in our prototype) simultaneous views. The proposed system acquires simultaneous synchronized images of objects uses a specific set of positions for the cameras to maximize the viewing volume and includes a refined camera calibration to notably increase the quality of the generated models with respect to a conventional scheme. The practical result is a fast reconstruction of 3D object sequences with a small latency between each object, about less than 3 seconds in a modern workstation. This leads to many interesting applications of the system like 3D inspection in a production line. An important part of the work has focused on camera calibration in multi-view environments because of its crucial importance in the developed system. We have presented some comparative images to illustrate the effects of calibration errors on the volumetric reconstruction and the effectiveness of the proposed method.

Finally, it is clear that some of the techniques described in this paper are not strictly limited to the multi-view volumetric carving problem but to any multi-view environment.

VII. ACKNOWLEDGEMENTS

Work partially supported by the Spanish MICINN grants TIN2009-14205-C04-02 and Consolider Ingenio 2010: MIPRCV (CSD2007-00018) and by the "Instituto de la Pequeña y Mediana Industria de la Generalitat Valenciana" (IMPIVA) and by the European Union by means of the European Regional Development Fund (ERDF) in the context of the R+D Program for Technological Institutes of IMPIVA network (IMIDIC-2009/204).

REFERENCES

Figure 7. Example of the damages caused from inaccurate camera calibrations in the 3D reconstruction of a fork.


