Camera calibration using iterative refinement with density function selection over concentric rings, chessboard and asymmetric disks patterns

E. Wilderd Mamani[1], Roxana C. Soto Barrera[2]

Abstract—Camera calibration is a necessary and important precedent for obtaining and reconstructing 3D information from a 2D model (image defined as base, floor), on the other hand there are library functionalities that offer us a procedure to perform this task an example clear is OpenCV [**?**], so for this report we used the library implemented by this, in addition to a proper implementation due to problems presented with the data (video) as presence of noise and inappropriate choice of circles or rings (base template), Thus, the most important concepts for the implementation of the camera calibration are detailed here.The technique only requires the camera to observe a planar pattern shown at a few (at least two) different orientation. The procedure implemented here is well suited for use without specialized knowledge of 3D geometry or computer vision and those method step by step are achieved and studied to get final camera calibrated bases on 2D pattern.

Index Terms—Camara calibration, iterative, chessboard, circle, ring

1 Introduction

THE CALIBRATION accuracy determine the accuracy of the measures that are carried out from the images. It is for this reason that it is essential to perform the camera calibration with full guarantees that the parameters obtained are like the real ones. This commitment implies both: the right choice of calibration method as well as the correct use of it. So, the calibration process should start by making a exhaustive review of the state of art over different calibration methods to choose the one that could get better results under defined conditions. Due to the large amount of work done in calibration field, it is an arduous and uncomplicated task choice of method and conditions for develop it.

The first step in calibration process is achieve to identify the flat template of concentric circular patterns, to do so, a segmentation algorithm is needed, most known segmentation algorithms and techniques are described in [2] and [3]. Segmentation process use features in common inside images to define an object, which even in human vision and perception also causes many confusions.

• *Maestria Ciencias de la Computación, Universidad Catolica San Pablo.*

The second step, according to [4] is object tracking, this process allow us to follow an object of interest over a video, instead of segmenting per frame the whole video, which has highly-cost. The methods that have presented better performances in tracking are based on Monte Carlo and probabilities, considering that tracking could be described as Markov Chain [5]. We opted for Particle Filter over Extended Kalman Filter, which is the other most used method, not just due points described before but also for cost-benefit between the implementation, efficiency of the method and its computational cost.

The third step, is solves calibration equation, this approach runs from images distortions introduced by pinhole cameras (very common). Two major distortions are radial distortion and tangential distortion. Due to radial distortion, straight lines will appear curved. Its effect is more as we move away from the center of image.

The fourth step, implys to obtain fronto parallel projection using calibration camera parameters

(to get undistort projected image), obteined from step third. To do so, first we have to project corrected and undistorted images to our full windows in order to calculate collinearity (if points supposedly corrected look actually as original/printed pattern), then using others parameters like homography matrix and perspective parameters as stop-criteria we can repeat calibration process until a convergence.
Finally, to finish our ideal pipeline, first a good flat template tracking has done, many position estimations should be performed (in better way) and finally a propper calibration process have to be performed, this part gonna be explored in future presentations.

2 Methods

This section presents the mathematical and theoretical definitions of points described before.

2.1 Pre-processing

This procedure is necessary because you have multiple noise images and step to get images frames, so that this step reduces the present noises in some way, acting as filters. These prepossessing filters are a way to eliminate or soften the content of high frequencies as edges, noises, among others, for this project the Gaussian filter is used in the first step because it is one of the filters that does not affect the contours.
Before applying this filter we will use an image transformed into grays as input, but without affecting much the contours of the pattern, some empirical values to take into account are: Rings: 3x3 window with the value of 0.5 for the standard deviation in both X and Y. Circles: 5x5 window with standard deviation on X axis equal to 2.5 and on the Y axis with value equal to 3. Then We used Thresholding, has intuitive properties, simple implementation and speed computational and is widely used in applications that require image segmentation.In this work, tests were carried out with the following types of segmentation, for example, Basic global threshold, When the intensity distribution of the object and background pixels are sufficiently different, it is possible to use a single (global) threshold over the whole image, forming two groups, one belonging to the object and another to the background.

For this work it was initially proposed to use this type of thresholding (with a threshold equal to 70), but due to the constant movement of the pattern throughout the video and different inclination angles there was a constant variation of the illumination on the pattern which led to the loss of the pattern in several frames. Below are shown Some images of the tests carried out: Then,

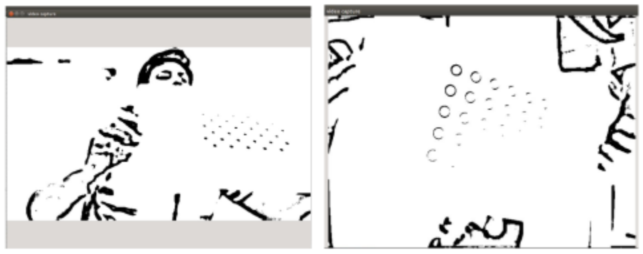
Figure 1. Results of global segmentation: circles pattern (left) and ring pattern (right)

we used Canny edge detector, is an edge detection operator that uses a multi-stage algorithm to detect a wide range of edges in images. It was developed by John F. Canny in 1986. Canny also produced a computational theory of edge detection explaining why the technique works. Canny edge detection is a technique to extract useful structural information from different vision objects and dramatically reduce the amount of data to be processed. It has been widely applied in various computer vision systems. Canny has found that the requirements for the application of edge detection on diverse vision systems are relatively similar. Thus, an edge detection solution to address these requirements can be implemented in a wide range of situations [**?**].

2.2 Region of Interest

We used to the contours are analyzed and the geometrical properties of the shapes are used to discard certain contours that do not comply with the minimum properties to be considered as possible candidates to conform the patter(image pixels). The geometric properties that have been used are:

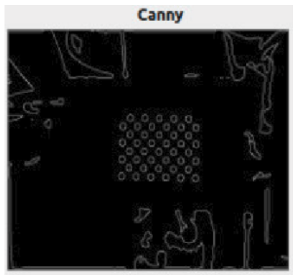

Figure 2. The Canny edge detector applied to a color photograph of a rings pattern.

- Aspect ratio: It is the relation between the length of the major axis and the length of the minor axis, in this case the bounding box was used to determine this factor. (asp = width / height) (aspect ratio of a perfect circle is 1))
- Rectangularity: The rectangularity represents how rectangular the form is, that is, that He fills his bounding box. (rect = area of the bounding / area of the bounding box). If of a perfect circle this value is 0,7853975
- Finally, for the most practical purposes, constants were used for the 0.5 rings: minimum factor aspect ratio of the parent ring, 0.4: aspec ratio of the child ring plus 0.7 and 0.4 minimum rectangularities for the father-in-child ring respectively.

The source of the setting parameters area show below in details:

```
// Constantes usadas para los circulos

#define C_MIN_ASPECT_RATIO 0.5
/*
* Relacion 2:1 entre el largo
* y ancho del bounding box
*/
#define C_MIN_RECTAN0.7
/*
* Rectangularidad mínima (circulo -> 0,7853975)
* Constantes usadas para los anillos
* Factor aspect ratio minimo del anillo padre
*/
#define R_PAR_MIN_ASPECT_RATIO 0.5
// Aspect ratio del anillo hijo
#define R_CHD_MIN_ASPECT_RATIO 0.4
// Rectangularidad mínima del anillo padre
#define R_PAR_MIN_RECTAN 0.7
// Rectangularidad mínima del anillo hijo
#define R_CHD_MIN_RECTAN 0.4
```

additionally, in the case of rings, since they are circles within one another, there is a hierarchy, in other words, these only have one circle within the other, in addition to one level.

2.3 Noise Reduction

In this section we will describe how some algorithms are used for the elimination of noise. Once the segmentation of a frame in the video was done, the following was the recognition of contours, as part of it, the fit ellipse function was used. This function in at the beginning return a noise frames that could be perceived easily to avoid those noises we are going to explain a several algorithms techniques to get better approaches in order recognize the contours with fit ellipse and then recognize rings.

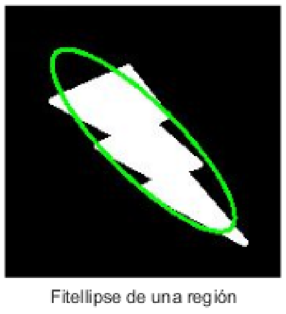

Figure 3. Fit Elipse from an image

During the previous procedures, we can obtain a detection of rings and circles, but with a series of noise errors. because we have some points that are still detected as the fitelipse. To improve this step, we are going to develop a series of algorithms so that we can obtain less background noise for this, we will apply the algorithm DFP, MST, AM, with a greedy euristica is which choose the best option in each time and this technique is used most on optimization problems.

2.3.1 Arithmetic mean

The arithmetic mean is the most commonly used and readily understood measure of central tendency in a data set. In statistics, the term average refers to any of the measures of central tendency. The arithmetic mean of a set of observed data is defined as being equal to the sum of the numerical values of each and every observation divided by the total number of observations.

$$A = \frac{1}{n}\sum_{i=0}^{n} a_i = \frac{a_1 + a_2+, ..., +a_n}{n} \qquad (1)$$

2.3.2 DFS

A depth search is an algorithm that allows you to traverse all the nodes of a graph or tree in an orderly, but not uniform way. Its operation consists in expanding each and every one of the nodes that it is locating, in a recurrent way, in a concrete way. When there are no more nodes left to visit on that path, it returns, so that it repeats the same process with each one of the brothers of the already processed node.

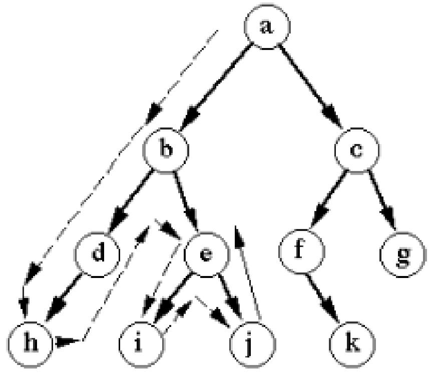

Figure 4. A DFS algorithm for a graph describing the track)

2.3.3 Minimum spanning tree

A minimum spanning tree (MST) or minimum weight spanning tree is a subset of the edges of a connected, edge-weighted undirected graph that connects all the vertices together, without any cycles and with the minimum possible total edge weight. That is, it is a spanning tree whose sum of edge weights is as small as possible. More generally, any edge-weighted undirected graph (not necessarily connected) has a minimum spanning forest, which is a union of the minimum spanning trees for its connected components.

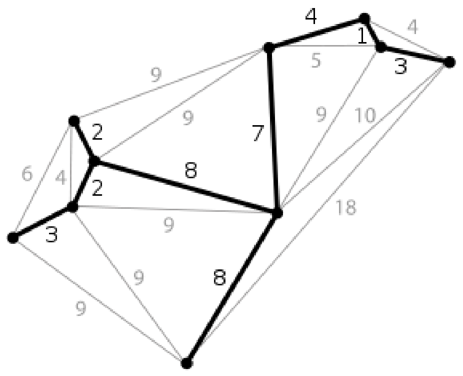

Figure 5. A planar graph and its minimum spanning tree. Each edge is labeled with its weight, which here is roughly proportional to its length (from wikipedia).)

2.3.4 Reduction approach

For the elimination of extra noise we perform a series of algorithms that we will explain step by step.

1) We start by finding a middle segment between the distances of all points, this segment will be useful to discriminate some distances that are very large with respect to the location of the pattern, this being our first filter.
2) in the second step we take a set of centers of the ellipses and then join them with their nearest neighbors, in order to form a graph.

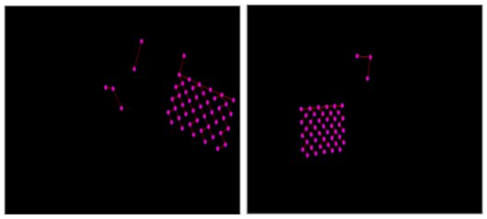

Figure 6. Frame of video with points join with the nearest neighbors to form the graph from test

3) the thirds approach's is to get which one of the graphs has large numbers of nodes, and we choose this as our patter section.
4) finally we apply MST to eliminate the large edge to discard it.

2.3.5 Kalman Filter

Developed by Rudolf E. Kalman, it is sought to estimate the real position of an object, since the

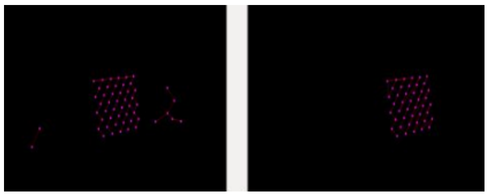
Figure 7. The Frames show, how we choose the graph with largest number of nodes to be the pattern location

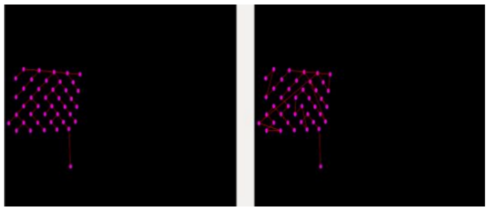
Figure 8. On the frame we can see a problem to fix with Minimal expanding tree for this special case

sensors estimate a real pseudo position due to the errors made, makes use of probability theory (Bayes Theorem, Gaussian distribution), dynamic, linear algebra and calculation to solve the equation to obtain the most possible position in which the object is. This can be used to estimate the following more real positions, useful for computational vision in the case of object occlusion when tracking is performed.

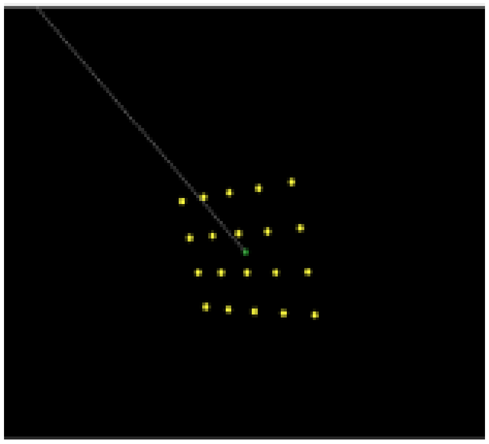
Figure 9. kalman filter to get all point in group

The Kalman filter was used to predict the next position of the pattern in the next frame and thus be able to estimate which points of the frame are part of the pattern. For this, we worked with the centroid of the pattern (average of all points by their positions x, y), to assemble the state vector. In the image the red point is the centroid of the pattern, the red point is the prediction of the centroid at time t (current) of the Kalman filter, and the white point is the prediction of the centroid at time ”t + 1” of the next position of the pattern. To calculate the points belonging to the pattern, a distance of each point was made against the estimated centroid, and then ordered from lowest to highest, then the first k was chosen, where k is the number of real points of the pattern.

2.4 Tracking Algorithm

We are implementing the track of the points, using the basic idea called for ours minareaRectangle who is a method of openCV to get the rectangle area of points in the grouping points of our patter of rings to determine that we calculate the angle with the position...

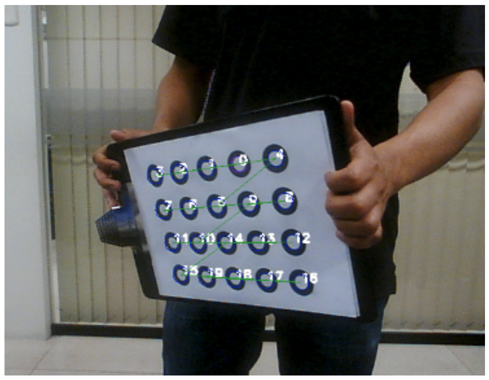
Figure 10. first tracking approach to get the lines on green and numbers

2.5 Frame Distribution

For the correct calibration of the camera (and in order to avoid of parameters overfitting over an exclusive zone of the images), it is necessary to have a quasi-equitable distribution of the patterns over the total of the scene, for which a density-based pattern distribution algorithm is applied, that is, the frames are selected for calibration based on how his spatial arrangement impacts on the average density and by sectors in the scene.

2.6 Camera Calibration

The camera calibration procedure on this paper use a so-called pinhole camera model. In this model, a scene view is formed by projecting 3D points into the image plane using a perspective transformation according to opencv definition.

Another definition According to [?], camera distortion is solved using five camera parameters, known as distortion coefficients (DC):

$$DC = k1, k2, p1, p2, k3 \tag{2}$$

Where K n = n th are the radial distortion coefficients and $P_n = n_th$, the tangential distortion coefficients.The radial distortion is irregular, the most commonly encountered distortions are radially symmetric, or approx-imately so, arising from the symmetry of a photographic lens.

$$\begin{bmatrix} X_corrected = x(1 + k_1r^2 + k_2r^4 + k_3r^6 \\ y_corrected = y(1 + k_1r^2 + k_2r^4 + k_3r^6 \end{bmatrix} \tag{3}$$

Similarly, another distortion is the tangential distortion which occurs because image taking lense is not aligned perfectly parallel to the imaging plane. So some areas in image may look nearer than expected. It is solved in 3:

$$\begin{bmatrix} X_corrected = x + [2p_1xy + p_{2(r^2+2x^2)]} \\ y_corrected = y + [p_1(r^2 + 2y^2) + 2p^2xy] \end{bmatrix} \tag{4}$$

In addition to this, we need to find a few more information, like intrinsic and extrinsic parameters of the camera.Intrinsic parameters are specific to a camera. It includes information like focal length (fx, fy) , optical centers (cx, cy) and others. It is also called camera matrix (CM). It depends on the camera only, so once calculated, it can be stored for future purposes. Extrinsic parameters corresponds to rotation and translation vectors which translates a coordinates of a 3D point to a coordinate system. It is expressed as a 3x3 matrix:

$$\begin{bmatrix} CM \end{bmatrix} = \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix} \tag{5}$$

As we said before,In order to reduce camera calibration process we decided to use calibrateCamera function from OpenCV. calibrateCamera function uses ordered arrays obteined from detection/tracking algorithm and windows size, and returns camera calibration matrix, distortion coefficients $(k_1, k_2, p_1, p_2[, k_3[, k_4, k_5, k_6]])$ of 8 elements, rotation and translation vectors.

2.7 Fronto Parallel projection

Is the view obtained by eliminating the perspective in an image. First a homography matrix is defined as the transformation between two planes inclusing scale factor (S), as can be seen in equation:

$$S \begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = H \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

The homography matrix is a 3x3 matrix but with 8 paramters as it is estimated up to a scale. It is generally normalized considering $h33 = 1$ or $h_{11}^2+h_{12}^2+h_{13}^2+h_{21}^2+h_{22}^2+h_{23}^2+h_{31}^2+h_{32}^2+h_{33}^2 = 1$. To get fronto parallel projection from each patter we first use keypoints and a distor in the function **findHomography** of opencv, this function obtain the homografy matrix from normal video to fronto parallel projection, then using **warpPerspective** to obtain a front-parallel view (with or without distotions depending on camera calibration process). This process became very usefull in order to calculate stop-criteria in the iterative process to refine calibration.

2.8 Iterative Method

The first calibration considerably reduces the distortion error, this can be observed visually in the corrected image (with corrected distortion) or by means of descriptors obtained from the corrected image, among the most important characteristics is the collinearity of the elements of each pattern (that the line formed by the elements stays in the direction). In order to improve the calibration, it is necessary to perform an iterative refinement of the calibration, to do so the parallel fronto projection of the corrected image is used (so the collinearity is made in the parallel frontal plane). Then, when performing the inverse projection of the fronto plane parallel to the normal view with distortion, new points are obtained on which the calibration calcula- tions can be made. From these 3 planes (parallel fronto, original/distorted and corrected), new control points can be estimated for next calibration process. A new calibration process which use these control points, improves image distortion and collinearity errors. When this process is performed iteratively, a greater gradient in descending order is expected in the error decay curve, this obserbations are exposed in the results sections of this paper. For our case, the barycenter was calculated between the control points of the 3 planes for the next step in the calibration. The iterative calibration refinement process described above is presented in the following image.

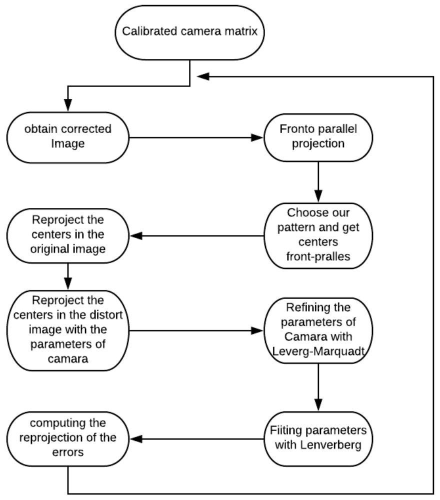
Calibrated camera matrix
obtain corrected Image
Fronto parallel projection
Choose our pattern and get centers front-pralles
Reproject the centers in the original image
Reproject the centers in the distort image with the parameters of camara
Refining the parameters of Camara with Leverg-Marquadt
Fiiting parameters with Lenverberg
computing the reprojection of the errors

Figure 11. Iterative Method

3 Results

This section presents results from all algorithms an modifications, errors included. So, each subsection shows results step by steps focusing on camera calibration and fronto-parallel parameters. Due most codes were developed for concentric ring pattern, most of our results are oriented to that particular patter (except for calibration and frontoparallel) because chessboard and asymmetrical circles have a their own detection/tracking function in OpenCV.

3.1 Pre-processing

Based on Gaussian Filter, Threshold adaptativo and Canny Edge Detection, we perform new results for feature recognition over elliptical patterns, those results are presented on figure 12.

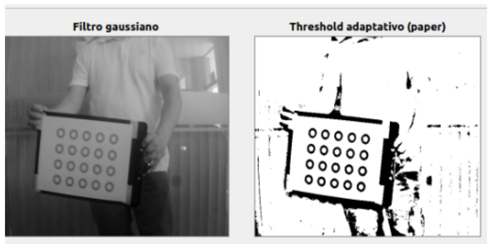

Figure 12. Preprocesing of Rings

3.2 Region of Interest

Based in geometric pro-perties: aspect ratio, rectagularity in others words we had used circles concentrics.

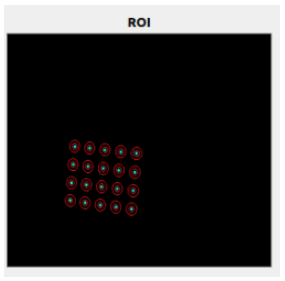

Figure 13. Region of interest to Rings

3.3 Noise Reduction

Based in Arithmetic means, minimun spaning tree and Kalman filters

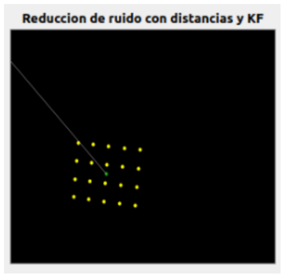

Figure 14. Noise reduction to Rings

3.4 Tracking Algorithm

For tracking, we use the first two frames in order to estimate initial conditions (needed for future estimations), this first initial variables are: angle and pose (usings points in arrange obtained from previous steps). Figure 15 shows average results from tracking.The main goal was solved, the method understand the pattern as a full element, which is composed by elements (rings/chessboard-/circles centroids). But a mal-detection of elements in rotation could cause a total estimation confution, and the main consecuence to fail in this part are future mal-estimation of states. To avoid it, we define a flag for special cases like: mal-detection in rotation or total missing of points. When those situations happens then all systems is rebooted matching last states calculated with new states estimated. As can be seen in figure 16, even in rotation the pattern are tracked. In some frames, a partial or total mal-detection happens (in middle of frames showed) then the cost function machting starts to work. In case of partial lose or total lose, the algorithm try to match elements from previous frame detected and new one. This is show in figure 17.

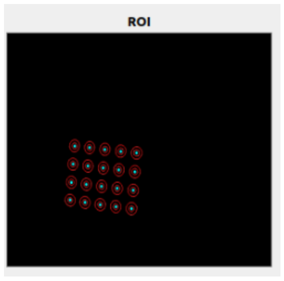

Figure 15. Concentric rings detection using pipeline in section 2

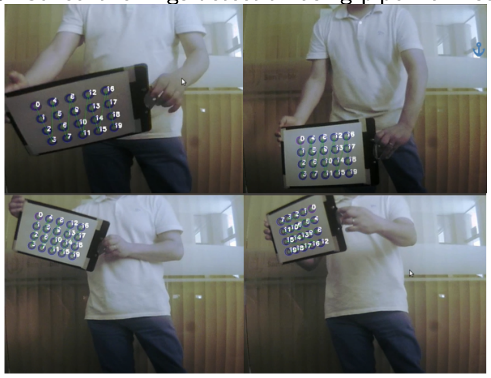

Figure 16. Concentric rings tracking results, the frames shows the results of Tracking in video, the video is presented by frames from top to down

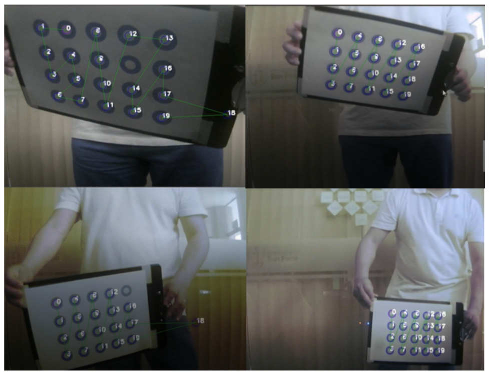

Figure 17. Matching function working over a pattern with partial or total mal-detection. In this case a partial mal-detection happens but the algorithm still works latter.

3.5 Frame Distribution

As was described before, density distribution could be very useful for calibration, due parameter obteined from a well-distributed vector of arranges should ensure a good distortion correction over the whole image. Image 18 shows a good distribution over scene using the first option: MANUAL, then the second option INTERVALS shows the same number of frames per pattern but using a frame capture by time, then in tables the collinearity error could be compared regarding good/ bad frames distribution. Figure 8 shows distribution results for concentric ring pattern in both videos, for 25 frames in 20 (4x5) sections.

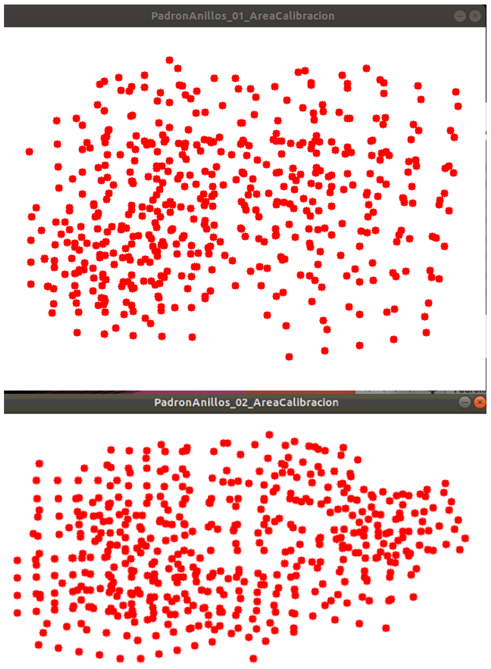
PadronAnillos_01_AreaCalibracion
PadronAnillos_02_AreaCalibracion

Figure 18. Frame Distribution to Rings

3.6 Camera Calibration

After tracking, and using OpenCV calibration function for 2 patters (P) :Asymmetric disks (0) and Chessboard (1), to calibrate 2 cameras: PS3 and LifeCam. $(c_x; c_y)$ is a principal point that is usually at the image center $(f_x; f_y)$ are the focal lengths expressed in pixel units.

P	F_x	F_y	C_x	C_y	RMS
			$N_F = 25$		
0	611.848062	620.398131	319.783601	273.558949	0.54837121
1	677.635412	690.63589	719.55126	658.36266	0.6253254
			$N_F = 35$		
0	699.972124	340.587525	701.081777	245.975391	0.222251
1	672.164181	672.164181	672.164181	672.164181	0.5959

TABLE 1. PS3 CALIBRATION RESULTS

P	F_x	F_y	C_x	C_y	RMS
			$N_F = 25$		
0	608.977979	601.998482	342.519779	227.219724	0.23051
2	525.377012	524.315352	358.92688	159.616455	0.459279
			$N_F = 35$		
0	509.876941	508.940996	276.055842	206.17429	0.372094
1	520.815631	511.098022	310.493886	164.373091	0.421195

TABLE 2. LIFE CAMERA CALIBRATION RESULTS

The Table 1 shows full results considering number of frames (N_F) used for calibration in ps3 camera, while 2 shows results in lifeCam camera. Corrected frames using calibration parameters are shown in figure 9 for concentric rings, chessboard and assimetric disks patterns in PS3 camera (due the correction is visually more remarkable in this camera), for this purpose we use undistort function from OpenCV, passing output parameters obteined from calibrateCamera function.

3.7 Fronto Parallel projection

Following the pipeline presented on subsection 2.7, we proceed to obtain the front-parallel view over the patterns. For this purpose as explained in section 2.6, we define the limit points (corners) of each pattern as the space to be projected (S_o) and the limits of the frontal screen as the space where the pattern have to be projected (S_f). Results in LifeCam camera are plotted in figure 19, while figure 20 present results in PS3 camera.

P	F_x	F_y	C_x	C_y	RMS
			Ct = PS3 Camera		
0	669.317504	656.649057	389.398628	269.763956	0.53092
			Ct= $LifeCamera$		
0	511.257561	510.402286	333.912424	199.896073	0.324985

TABLE 3. CALIBRATION RESULTS USING DENSITY DISTRIBUTION

3.8 Iterative Method

As explained in section 2.6, it is likely that the results using an iterative refinement method for calibration (generating new control points over the calibration functions of the camera and parallel fronto projection). For this reason and according to the results of table 3 (better results using density for the concentric ring pattern, as proposed in [7]) the pipeline presented in figure 1 was proposed, also using the frame selection function by density. Results are presented in tables 4 and 5 for LifeCam and PS3 cameras respectively with 30 iterations (It).

	Chessboard	Assim. disks		Concc. rings	
	RMS	RMS	iterative	RMS	iterative
30	0.6253254	0.548371206	0.513531	0.678611	0.303971

TABLE 4. ITERATIVE REFINEMENT CALIBRATION RESULTS USING DENSITY DISTRIBUTION CAPTION FOR PS3 CAMERA

	Chessboard	Assim. disks		Concc. rings	
	RMS	RMS	iterative	RMS	iterative
30	0.372094	0.421195	0.313727	0.324985	0.299

TABLE 5. ITERATIVE REFINEMENT CALIBRATION RESULTS USING DENSITY DISTRIBUTION CAPTION FOR LIFE CAMERA

As can be seen, mostly results from concentric ring pattern and assymetric disks are affected in iterations, but in general after effects produced by the first iteration, RMS and collinearity are not reduced much more in that magnitude.

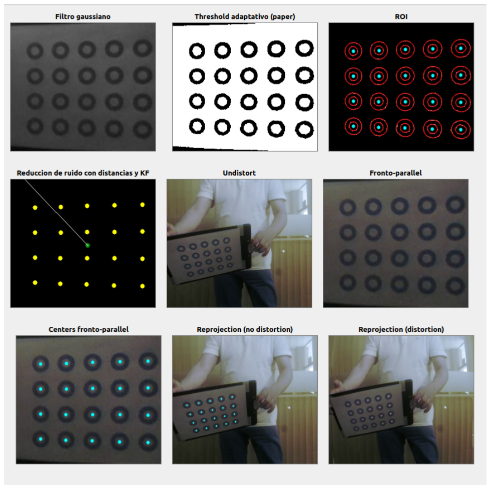
Filtro gaussiano
Threshold adaptativo (paper)
ROI
Reduccion de ruido con distancias y KF
Undistort
Fronto-parallel
Centers fronto-parallel
Reprojection (no distortion)
Reprojection (distortion)

Figure 19. Iterative Method Rings

4 Conclusions

In this implementation it is observed that for the recognition and calibration of cameras we have to apply a series of filters and algorithms to obtain a recognition of the sample template, in addition to adjusting the appropriate parameters we have to use an empirical way, segmentation and noise reduction should be applied to the image, finally when applying the algorithms to find the rings and circles on the real time video , we have to apply a heuristic of Ellipse fit for to find our images and a series of criteria to be taken into account and all characteristics were established according to the empirical experiment.

As seen in the results, the asymmetric circles and rings patterns are better than the chessboard because they reduce the error by detecting their centers instead of edges and vertices, which usually tend to fail. This generates an error in the calibration calculus obtaining very variable and distant results. This phenomenon is corrected using the iterative method but it does not apply to the rest of patterns. Between the assymetric circles and concentric rings, the last one have several concentric circles that help reduce the error when calculating the centers of the pattern. The iterative method does not imply a great advantage (in comparison with results obtained selection density function), it helps to reduce the error but it is not very significant, only in cases of cameras with greater distortion it is possible to appreciate like in ps3 case and for patterns that do not have much noise like concentric rings. About results presented on [7], which we are trying to reproduce or at least understand and extrapolate to our case (PS3 and LifeCam cameras), we conclude that improvement percentages are proportional to the distortions level presented in cameras used. So if a video has little distortion (LifeCam) will not be improved on the corrected image, but for videos with high distortion (PS3) this correction becomes most notorious.

References

[1] Zhang Z. (1998) A flexible new technique for camera calibration. Technical Report MSR-TR-98-71, Microsoft Research.

[2] Meier, Thomas, and King Ngi Ngan. ”Automatic segmentation of moving objects for video object plane generation.” IEEE Transactions on Circuits and Systems for Video Technology 8.5 (1998): 525-538.

[3] Cucchiara, Rita, et al. ”Detecting moving objects, ghosts, and shadows in video streams.” IEEE transactions on pattern analysis and machine intelligence 25.10 (2003): 1337-1342.

[4] Yilmaz, Alper, Omar Javed, and Mubarak Shah. ”Object tracking: A survey.” Acm computing surveys (CSUR) 38.4 (2006): 13.

[5] Yang, Changjiang, Ramani Duraiswami, and Larry Davis. ”Fast multiple object tracking via a hierarchical particle filter.” Computer Vision, 2005. ICCV 2005. Tenth IEEE International Conference on. Vol. 1. IEEE, 2005.

[6] Szeliski, Richard. Computer vision: algorithms and applications. Springer Science and Business Media, 2010.

[7] Datta, Ankur, Jun-Sik Kim, and Takeo Kanade. ”Accurate camera calibration using iterative refinement of control points.” Computer Vision Workshops (ICCV Workshops), 2009 IEEE 12th International Conference on. IEEE, 2009.