Performing iterative camera calibration using integral threshold filter and meta-heuristics to get Fronto Parallel images on ring-elliptical, Chessboard, and points-circles to refine parameters of the intrinsic matrix

1st Felipe A. Moreno
Department of Computer Science
Arequipa, Peru
felipe.moreno.vera@gmail.com

***Abstract*—In this work, I present a way to calibrate a camera using different filters like Gauss, median and others to detect patterns like circles, ellipses and point in a determinate area in Real Time through cameras.**

***Index Terms*—calibration, camera, patterns, detection, filters**

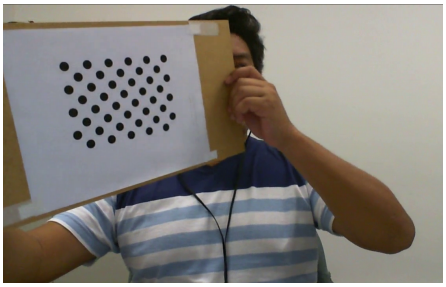
I. Introduction

Camera calibration have a lot of applications nowadays in fields like Deep Learning, Images analyses, CV, AR, VR and so on. But in some cases the most difficult task to do is calibrate or make the images sharper. In Deep Learning is very commonly work with images using filters and dimensional reduction to get a better results from them, to do this process we use computer vision algorithms to make filters and get characteristics from them.

II. Methodology

A. Gray Scale

The most common way to remove noise from the image is convert the image to Gray Scale and then using a Gaussian filter.As you can see in Figure 1(b), image turns to gray Scale.



(a) Original

(b) Gray Scale

Fig. 1. RGB to GraySCALE

B. Gauss Filter

With the Gauss filter we can remove noise from the image to get an standardized image (see Figure 2). This is very useful to get an image regularizated, with this first filter we can add more to get a better patterns to clean or highlight.

C. Integral Thershold

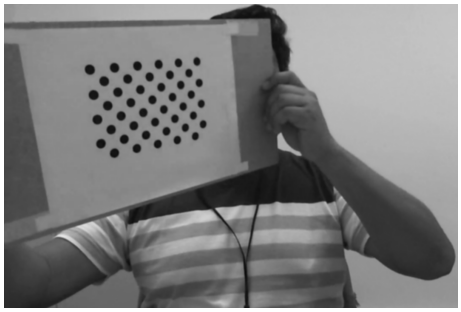
I implement this algorithm propose in some works accepted by CVPR or ICCV conference that provide a new way to remove noise (using Threshold) without binary or otsu filter, with this method we have an organized behavior of pixels in a continuos way.

D. Contours localization

After to apply Integral threshold (we tried more filters before integral) and we need to localize some contours similar to ellipses (our edges). With the findContours function from OpenCV we detect those (instead using Canny algorithm that generates a lot of noise).

E. Fitting Ellipses

To fit Ellipses, we use the fitEllipse function from OpenCV who returns a contours which satisfied the condition of "ellipse shape". But in this function we have extra objects detected by the algorithm and those objects dont be part of our patterns goals.



(a) Gauss Filter

(b) Canny Algorithm

Fig. 2. Gauss filter and Canny Algorithm

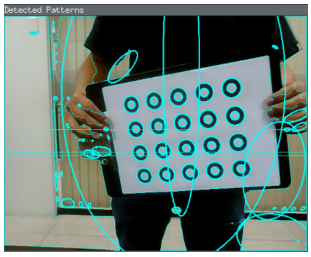
F. Clean objects

To clean objects or "extra patterns" detected we use some heuristics as parameters like Size, radio, shape, area, with this technique we discard some extra objects detected by our fitEllipse.

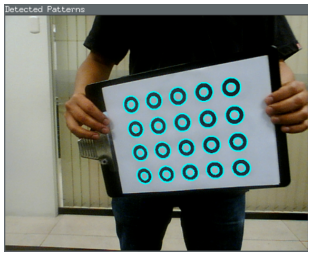
III. Methodology

After to filter all noise from the original video we can follow the movement of our patterns (in this case rings and points) as you can see in Fig. 3.

To do that, we use some heuristic te remove and calculate using mathematical fundamentals and concepts about ellipses or similar patterns. once we finally



(a) Noise



(b) Without Noise

Fig. 3. Noise remove

IV. Results

Our camera is 640x480 so our calibration centers should be nearly to (320, 240). After to do iterative calibrate, we get following results:

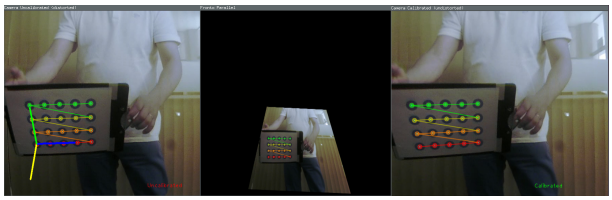
Now with the sorted points (centers) we can do the calibrate method to get camera parameters for the distortion matrix.

Camera 1 (640x360)			
Parameters\Patterns	Rings	Asymmetric Circles	ChessBoard
fx	255.8385039893901	497.3848627129802	692.3860785658198
fy	256.7572484359887	498.4529799964733	692.9770200510819
Cx	312.3261142683546	320.0362475738945	324.5863403004979
Cy	199.4131828877674	177.6429976795606;	205.3832720933052
Re-Projection error	0.348439	0.449519	0.53702

(a) Camera parameters

Fig. 4. Camera Parameters

To generate fronto parallel, we need to get for first time the camera parameters and get the undistorted and re-projected points, so with both set of points we can calculate the Homography Matrix, and with that homography matrix we can change the point of view to get Fronto Parallel.



(a) Fronto Parallel

Fig. 5. Fronto Parallel view

To do this process of iterative method, we need to generate the fronto parallel view for patterns that we are detecting, the process is get matrix coefficients, undistort camera (with the parameters) then project points from undistorted to distorted frame, generate new camera parameters and do that process iterative. All this procces was proposed by Ankhur.

So, Once we can track and order our centers and also calibrate our camera using, we are able to use and implement the Iterative method of Ankhur to calibrate our camera (using distortion and re-projection). We get following results:

Parameters\Patterns	Iter 0	Iter 1	Iter 5	Iter 10	Iter 15	Iter 20	Iter 25
fx	1309,928693	665,4105195	708,6898262	705,1401215	691,3742381	683,0611491	675,4484083631678
fy	1168,793562	667,7956909	703,8800999	702,9959326	688,8151507	679,1030005	678,481389058379
Cx	347,0104134	334,6353837	293,2904425	342,8106079	324,6743947	319,4102864	338,6881020412026
Cy	507,2860439	269,9136176	272,3544074	272,2438921	261,7435702	257,5005938	242,9039863123267
rms	0,523891	0,455933	0,286075	0,350958	0,27287	0,250603	0,179749

(a) Camera parameters

Fig. 6. Camera Parameters Iterative Method

V. Conclusions

We can calibrate iterative the camera using Ankhur algorithm and fronto parallel technique, With both and another metrics and heuristics we can improve and reduce the RMS

and distort coefficients error in camera parameters, getting a better calibrated frames.

REFERENCES

[1] Derek Bradley, Gerhard Roth, "Adaptive Thresholding Using the Integral Image" Phil.

[2] Ankur Datta, Jun-Sik Kim, and Takeo Kanade, Accurate Camera Calibration using Iterative Refinement of Control Points.