

# Calibración de una cámara usando un padrón en forma de anillo

Alejandra Callo Aguilar<sup>1</sup> y Jose Jaita Aguilar<sup>2</sup>

**Abstract**—En este trabajo se propone un método para la calibración de cámara usando un padrón de anillos, posteriormente se compara los resultados usando otros padrones como el chessboard y el de symmetric circles. Una parte del pipeline es la detección de anillos, para lo cual se propone un enfoque simple, robusto y rápido para dicha tarea.

## I. INTRODUCCIÓN

La calibración geométrica de la cámara, también conocida como resección de cámara, estima los parámetros de una lente y un sensor de imagen de una imagen o cámara de video. Puede usar estos parámetros para corregir la distorsión de la lente, medir el tamaño de un objeto en unidades mundiales o determinar la ubicación de la cámara en la escena. Estas tareas se usan en aplicaciones tales como visión artificial para detectar y medir objetos. También se usan en robótica, para sistemas de navegación y reconstrucción de escenas tridimensionales.

El trabajo basicamente consta de detectar los anillos, luego con los puntos de control (centro de los anillos) se pasa a la parte de la calibración usando OpenCV [1] para dicha tarea.

El enfoque que se propone para la detección es simple, debido a que no se usan heurísticas complejas, esto conlleva a que sea muy rápido y pueda correr en tiempo real.

## II. DETECCIÓN DEL PADRÓN

### II-A. Detección de contornos

Primero, para la detección de los anillos, la imagen de entrada se convierte a gray, luego se le aplica una binarización usando el método de Otsu[2], para esto usamos la función de OpenCV "threshold", con la particularidad que le decíamos que nos calcule un threshold adecuado usando el método de Otsu, se pudo haber elegido manualmente un threshold, pero debido a la variación de iluminación, esta etapa no hubiese sido robusta. Luego tenemos una imagen binaria a la cual le aplicamos la función "findContours", entonces tenemos todos los contornos de los anillos pero también de otros objetos.

### II-B. Filtros

Esta etapa consiste en remover todo el ruido de las imágenes, es decir remover todos los contornos que no formen parte de los anillos, el primer filtro que se usa es en cuanto a la relación de ancho y altura, debido a que es un círculo, la relación de tales parámetros debería estar cerca a 1, en

la práctica se comprobó que dicha relación fue hasta un límite de 0.4, esto se debió a la perspectiva en que se tomaba la imagen. El segundo filtro era el área que formaba el contorno detectado, para el video de Kinetic2 se tuvieron áreas que fueron desde 1000 hasta 9000 píxeles, el tercer filtro fue escoger solamente los contornos que sean padres, como se sabe el anillo tiene dos contornos, el círculo exterior y el interior, entonces al usar la función "findContours" se obtenía la jerarquía de los contornos encontrados, entonces al tener dicha información se filtraba todo contorno que no sea padre. Como se menciona los parámetros de relación y área de los contornos eran variables, entonces dichos parámetros se actualizaban con cada imagen de entrada.

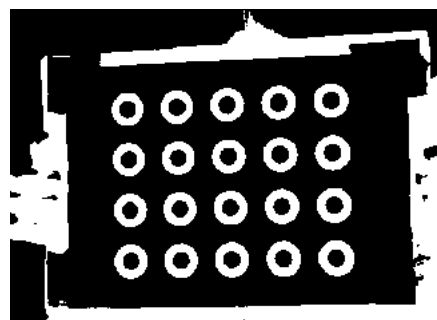


Fig. 1. Imagen binarizada usando Otsu

## III. REGIÓN DE INTERÉS (ROI)

La eficacia del método Otsu para hallar un threshold adecuado, depende mucho a qué región de la imagen se aplique, lo ideal es que solamente se use la zona en donde se encuentran los anillos.

### III-A. ROI

Una vez detectado todos los anillos se define un roi, para esto simplemente hallamos la ubicación del anillo que se encuentra más cerca al origen y el más lejano, de este modo se define el roi, pero debido a que dicho roi se aplicará en el siguiente frame, y los anillos podrían cambiar de posición, entonces se le aplicaba un pequeño padding de 10% del tamaño del roi, con tal ajuste era suficiente ya que entre frame y frame no hay mucha variación ahora, si subimos a porcentaje, se corre el riesgo de que el método de Otsu use píxeles irrelevantes haciendo que no de un buen threshold. En la Fig 2. se muestra el roi de color celeste.

## IV. TRACKING

### IV-A. Transformación

Para la calibración es necesario hacer el tracking de los anillos detectados, cuando hallamos los contornos en la

\*Este trabajo es patrocinado por CONCYTEC

<sup>1</sup>A. Callo Aguilar, estudiante de la maestría en ciencias de la computación, Universidad Católica San Pablo, alejandra.callo@ucsp.edu.pe

<sup>2</sup>P. J. Jaita, estudiante de la maestría en ciencias de la computación, Universidad Católica San Pablo, jose.jaita@ucsp.edu.pe

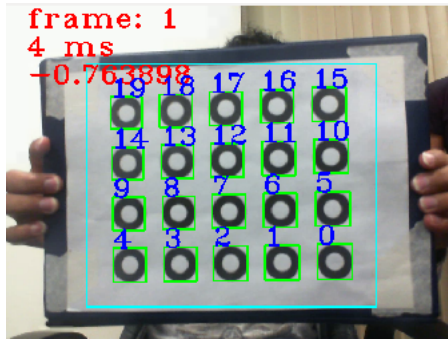


Fig. 2. Imagen con tracking

etapa anterior, si la imagen se encontraba sin ningun tipo de rotación, nosotros ya teniamos los anillos ordenados, la funcion "findconoturs" devuelve los contorno con respecto al eje y. Los problemas comenzaban cuando se rataba la imagen entonces lo que se hizo fue una tranformacion de espacio, simplemente rotabamos la imagen y en esta nueva imagen encontrabamos los contornos, de esta forma teniamos el orden correcto de los contorno en la imagen original. Para que la rotación funcionase se debía conocer el angulo correcto, entonces nosotros actualizabamos constantemente dicho angulo, el cual era hallado usando la funcion de arcotangente entre la posicones del anillo 0 y el 1, obviamente el siguiente frame puede ser que cambie el angulo entonces no se trendra una ratación perfecta, ya que en cada frame se usa el angulo hallado con los anillos del frame anterior , entonces lo que se hace es llamar dos veces a la funcion de tranformacion, primera con el angulo pasado, luego hallabamos el nuevo angulo, de nuevo llamamos a la tranformación , entonces se tenia una tranformacion perfecta.

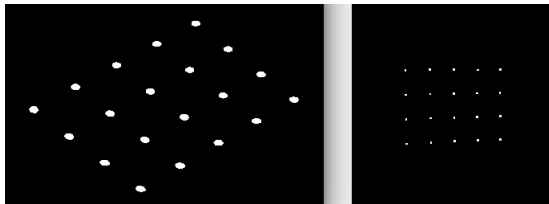


Fig. 3. Parte izquierda, la imagen original, lado derecho imagen rotada

En la Fig 3. vemos el resultados de transformar una imagen hacia otro espacio, en donde tenemos perfectamente ordenados lo anillos.

#### IV-B. Puntos de control

La calibración depende exclusivamente de cuan preciso calculemos los centros de los anillos (puntos de control), una vez detectado el anillo volviamos a la imagen original para no perder informacion ya que , todos los metodos aplicados anteriormet se aplicaba al roi entero , el cual mete ruido , perdiendo detalles de los anillos ,por lo tanto para detectar el centro del anillo se trabaja solo en la region del contorno. Se aplica el metodo de Otsu para binarizar luego se encuentra los contornos, en este caso obteniamos

solamente 2 contorno el del anillo exterior y el del interior, luego hallamos el centro de masa del objeto formado por cada contorno, en nuestros experimetos los centros de los dos círculo diferian por poco, menos de un pixel en promedio, entonces simplemente hallabamos el promedio de los dos, y este vendria hacer nuestro punto de control para nuestra calibración.

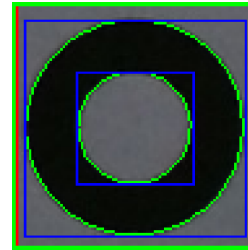


Fig. 4. Anillo segmentado

[1206.02, 656.321]	[1205.97, 655.897]
2	2
[1084.8, 621.017]	[1084.88, 620.831]
2	2
[967.232, 587.129]	[967.041, 586.782]
2	2
[852.644, 553.738]	[852.473, 553.547]
2	2
[1232.72, 541.848]	[1232.76, 541.443]
2	2
[741.03, 520.803]	[741.055, 520.456]

Fig. 5. Diferencia de centros de masa, izquierda exterior, derecha interior

En la Fig 4. se muestra un anillo segmentado con los bordes dectados, ahora se necesita el punto de control, en la Fig 5. se muestra la diferencia entre los centros de masa calculados para el círculo exterior y el interior. Por ejemplo [1206.02, 656.321] es la ubicacion del círculo exterior y [1205.97, 655.897] del círculo interior para el anillo de la fig 1. Los resultados anteriores fue par un frame relativamente bueno, en cambio cuando se calculo el centro de masa para otros frames en donde habia cierto blur debido al movimiento, la diferencia llego hace de hasta 1.2pixeles.

## V. CALIBRACIÓN

Una vez hallado los 20 anillos pasamos a la calibración usando la función de OpenCV "calibratecamera"[1], a dicha funcion debiamos pasarle los puntos de control de la imagen (en nuestro caso los centros de los anillos) y las coordenadas de los objetos, en este caso seria la ubicacion de los puntos fisicos, para esto se uso como referencia a la imagen mas no a la cámara, entonces para nuestros puntos fisicos nuestro eje de coordenadas es el tablero del padrón , y ya que cada punto de control esta distribuido uniformemente no era necesario medir distancia.

#### V-A. Parámetros intrínsecos

La preposicion anterior era valida debido a que solo estamos hallando los parametros intrinsecos, basicamente en la calibracion se halla la matriz de la cámara, la cual contiene

las distancia focales  $f_x$  y  $f_y$  además de los centros opticos  $c_x$  y  $c_y$  expresados en pixeles. Pero la funcion de calibracion no solo calculaba eso, sino que además nos dio los coeficientes de distorsión radial.

## VI. REFINAMIENTO ITERATIVO DE LOS PUNTOS DE CONTROL

Nos basamos en el método propuesto por A. Datta[3], los pasos para obtener los puntos refinados se enumeran a continuación.

### VI-A. Inicialización

Lo primero que se hizo fue hacer una primera calibración, de modo que se obtenga tanto los parámetros intrínsecos como los extrínsecos, estos últimos nos da el vector de rotación y traslación por cada imagen usada para la calibración (supongamos que usamos  $N$  imágenes).

### VI-B. Un-Distorcionar y Un-proyectar

Como ya calculamos los parámetros de la cámara lo que hacemos es quitarle la distorsión a los  $N$  frames usando el metodo de OpenCv (undistord). Lo siguiente es obtener las imagenes en perspectiva fronto-paralela, cuando se hizo la primera calibracion, OpenCv nos pidió la ubicación de los puntos de control en el mundo real(puntos 3D), para esto nosotros definimos un eje de coordenadas relativo en donde el padron de anillos era el origen de coordenadas, con  $z = 0$ , y los puntos de control se encuentran a lo largo del eje XY, entonces para hacer la proyeccion de dichos puntos en la imagen necesitamos una matriz de homografia la cual nos permite pasar del sistema de coordenada del objeto al de la cámara. Como se menciono anteriormente ya tenemos el vector de rotacion y translacion, el siguiente paso es hallar la matriz de homografia.

$$H = H_{rotacion} * H_{traslacion} \quad (1)$$

En (1) se tiene la matriz de homografía, la cual nos permitirá tener una vista fronto-paralela de padrón, de modo que se pueda calcular los puntos de control sin perder detalles. La homografia se descompone en  $H_{rotacion}$  y  $H_{traslacion}$ , actualmente la imagen esta a una distancia focal del centro optico además que se le aplico el desplazamiento del centro, entonces tenemos que regresar la imagen de modo que el centro coincida con la coordenada (0,0) y para una distancia focal de 1, para esto lo multiplicamos por la inversa de la matriz de la cámara( $K$ ), luego le aplicamos la matriz de rotacion inversa(es igual a la transpuesta), ahora ya tenemos el centro optico de la cámara paralelo al padrón objeto, en (2  $R$  es la matriz de rotacion, previamente nosotros tenemos la rotacion pero en forma de vector, para convertirlo a matriz usamos Rodriguez de OpenCv), finalmente le aplicamos  $K$  para obtener la proyeccion del padrón en la imagen, esto viene hacer  $H_{rotacion}$ , si bien la imagen esta fronto-paralela al padrón, aun falta hacer un desplazamiento para tener un alineamiento perfecto entonces pasamos a calcular  $H_{rotacion}$  (3), en donde  $T$  es el vector de traslación.

$$H_{rotacion} = K * R^T * K^{-1} \quad (2)$$

$$H_{traslacion} = K * R^T * T \quad (3)$$

### VI-C. Nuevos puntos de control

Para tener el padrón completo en la imagen hacemos una transformacion usando una homografía, se uso 4 puntos de control (especificamente las esquinas), entonces tenemos el padrón en fronto-paralelo con  $z = 1$ , para esto usamos  $h1$ , luego hallamos los puntos de control usando el algoritmo de detección adecuado dependiendo del padrón, luego regresamos a la imagen obtenida de la un-proyección, y finalmente re-proyectamos (4) los puntos usando la inversa de  $H$ , en este punto ya tenemos el punto de control en la imagen original sin distorsión usando  $H_{regresar}$ .

$$H_{regresar} = H^{-1} * h1^{-1} \quad (4)$$

### VI-D. Distorcionar

El punto de control actual esta en una imagen sin distorsion, pero como la imagen original estaba distorsionada, entonces es necesario distorsionarla, para esto usamos los coeficientes de distorsión y la matriz de la cámara.

### VI-E. Nueva calibración

Con los nuevos puntos de control para las  $N$  imágenes volvemos a calibrar pero previamente calculamos la media entre los puntos antiguos y los nuevos, esto se hace hasta que el rms converja. Para nuestros experimentos se uso 5 iteraciones, aunque en algunos videos de prueba la convergencia se alcanza en la segunda iteración. Para los videos finales la iteracion recién se alcanza en la quinta iteración. En Fig(6) se muestra la corrección de los puntos de control luego de hacer la re-proyección.

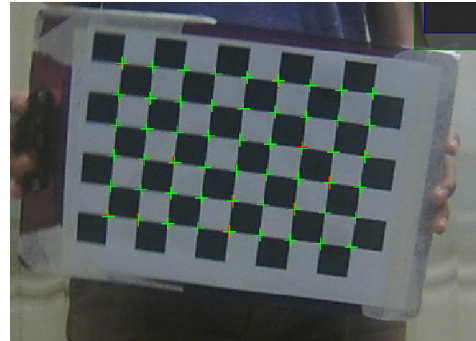


Fig. 6. Nuevos puntos de control

## VII. EXPERIMENTOS

Para nuestros experimentos usamos dos cámaras (Playstation PS3 y lifecam), para la calibración de dichas cámaras se uso 3 padrónes:

- Chessboard: los puntos de control eran hallados usando la funciones de OpenCv "findChessboardCorners".
- Symmetric circle: usamos la funcion de OpenCV "findCirclesGrid".

TABLE I  
RMS

Camera	Chessboard	Circle	Ring
LifeCam	0.308884	0.28751	<b>0.182048</b>
PS3	0.504785	0.41254	<b>0.305087</b>

TABLE II  
DISTANCIA FOCAL FX, FY

Camera	Chessboard	Circle	Ring
LifeCam	597.6 587.5	580.4 585.2	582.2 580.1
PS3	857.7 847.1	842.1 824.8	810.9 814.6

TABLE III  
CENTROS CX,CY

Camera	Chessboard	Circle	Ring
LifeCam	313.6 226.9	314.8 228.2	321.8 224.9
PS3	338.4 262.9	325.7 249.5	316.8 241.6

TABLE IV  
COLINEARIDAD

Camera	Chessboard	Circle	Ring
LifeCam	0.0879 - 0.0625	0.0825 - 0.0684	0.0966 - 0.0522
PS3	0.124 - 0.0642	0.1025 - 0.0498	0.1102 - 0.0587

- Rings: para los anillos usamos nuestro metodo.

En las tablas I,II,III y IV se muestran los resultados llevados a cabo, usando tres patrones , dos cámaras y 20 imagenes por cada video, como se puede ver los mejores resultados fueron obtenidos usando el padrón de anillos, en el caso de la colinealidad se muestra dos valores, el primero es la colinealidad usando una simple calibracion y la segunda es usando el refinamiento.

Como se sabe un parámetro para saber cuan bien se calibro es el RMS, pero no solo es ese, sino que ademas de calcular dicho parámetro. se necesita ver como actua la calibracion en imagenes, en nuestros experimentos se probo muchas calibraciones, y cuando se aplicaba la corrección usando su respectiva matriz de la camara y los factores de distorsión a veces se obtenia una imagen concava y aveces convexa. Ademas de tener una referencia gráfica, se aplico la colinealidad de modo que se tenga una metrica para comparar las calibraciones. Para nuestros experimentos usamos imagenes seleccionadas manualmente ya que nos da mejores resultados debido a que se evita seleccionar imagenes repetidas o con poco rotacion ya sea a travez de eje X o Y, pero aparte de eso se hizo pruebas usando un filtrado de imagenes, en donde solo se usaban imagenes que tenga una rotacion entre 5 y 60 grados, en la Fig(7) se muestra la comparacion usando diferentes cantidades de imágenes, como se observa se tiene mejores resultados cuando las imágenes son filtradas.

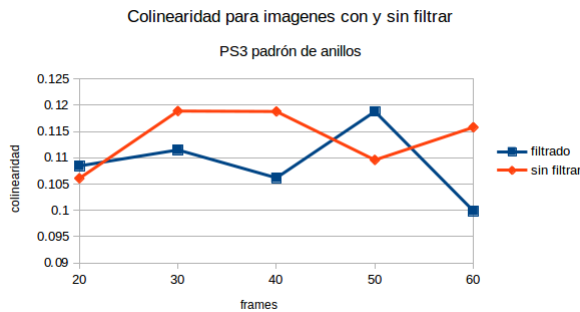


Fig. 7. Colinealidad para imágenes filtradas

En la Fig(8) ahora se observa como se comporta el rms con respecto al filtrado de imagenes para la calibración usando

diferentes cantidades de imagenes para la calibracion.

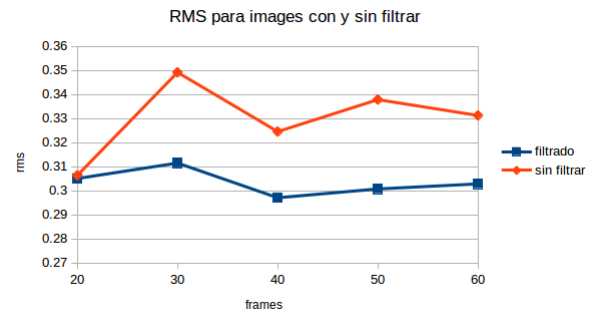


Fig. 8. RMS para imágenes filtradas

Como ya se mencionó se usaron 5 iteraciones para el refinamiento, en la Fig(9) se muestra en comportamiento del rms a traves de las 5 iteraciones, para los tres patrones usando tanto PS3 como LifeCam.

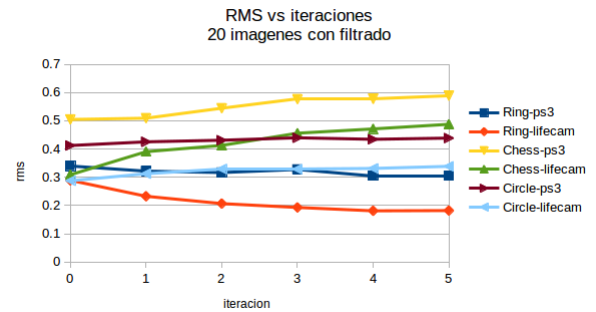


Fig. 9. RMS a través del refinamiento

Ademas del RMS, en las Fig(10) y Fig(11) se muestra la distancia focal(fx,fy) y centro óptico(cx,cy) respectivamente a travez de las 5 iteraciones, para este experimento se uso el padrón de anillos con 20 imágenes y la PS3 ademas que para todos los experrmientos se uso el filtrado de imagenes (6 a 60 grados) ya que nos da mejores resultados como se muestra en la Fig(7) y Fig(8)

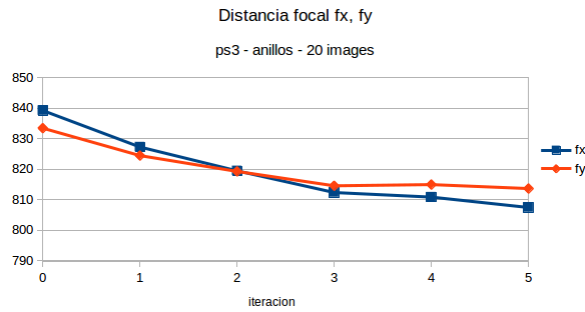


Fig. 10. RMS a traves del refinamiento

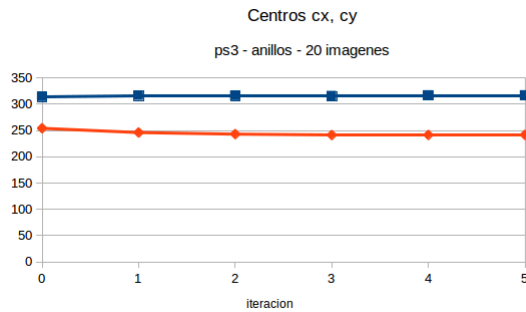


Fig. 11. RMS a traves del refinamiento

## VIII. RESULTADOS

### IX. CONCLUSIONES

El método propuesto para la detección de los anillos es robusto y rapido, corre en tiempo real, es mas puede correr a 60fps, usando las imagenes de la camara lifescan le tomaba entre 5 a 10ms hacer el procesamiento. El que los anillos tenga dos circulos ayuda mucho a la detección, facilmente se elimina falsos negativos. En cuanto a la calibración se escogio 20 imagenes por video, fue con el que nos dio los mejores resultados, la tabla 1 son los mejores resultados obtenidos despues de haber corrido varios experimentos con diferentes numero de imagenes. Tanto el padrón circular como el de anillos muestra una mejora con respecto al clasico chessboard, con esto se concluye que el padrón en forma de anillo es el mejor para la calibración. El motivo que nosotros pensamos es porque los circulos son invariantes a la rotación, aunque con cierta perspectiva el circulo tiende a convertirse en elipse, pero aun asi se conserva el mismo punto de control, ya que lo que se halla es el centro, en cambio en el chessboard cuando hay rotacion mas un cambio de perspectiva se obtienen puntos de control diferentes. Se concluye que el parametro principal para que la calibracion sea buena, son los puntos de control calculados, estos deben ser lo mas preciso posible, tambien ayuda el numero de imagenes y cuantos puntos de control se utilizan por cada imagen, por ejemplo si hay un error en un punto y si los demas estan bien, entonces el error se reduce. Se comprobó que usar imagenes entre 5 y 60 grados da una mejor calibración, para los videos de prueba el refinamiento no funciono para el padrón de ajedrez y el



Fig. 12. Imagen original, ps3

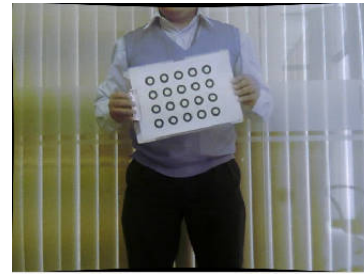


Fig. 13. Imagen calibrada, ps3, 1era iteracion

circular en cambio para el de anillos si hubo convergencia. Si bien en los videos de prueba no funcionó en otros videos de prueba si hubo convergencia. La principal diferencia entre los videos es que en los nuevos videos de prueba el padrón se encuentra muy alejado.

### X. TRABAJOS FUTUROS

Para el siguiente trabajo se determinará porque en los padrones del ajedrez y el de circulos no funciona el refinamiento.

### REFERENCES

- [1] OpenCV, camera calibration: [http://docs.opencv.org/doc/tutorials/calib3d/camera\\_calibration/camera\\_calibration.html](http://docs.opencv.org/doc/tutorials/calib3d/camera_calibration/camera_calibration.html)
- [2] PoenCV, Otsu: <https://docs.opencv.org/3.0-beta/doc/tutorials/imgproc/threshold/threshold.html>
- [3] A. Datta, J. S. Kim and T. Kanade, "Accurate camera calibration using iterative refinement of control points," *2009 IEEE 12th International Conference on Computer Vision Workshops, ICCV Workshops, Kyoto, 2009, pp. 1201-1208*.

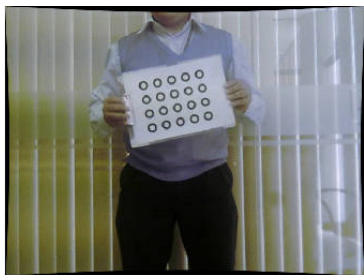


Fig. 14. Imagen calibrada, ps3, 5ta iteracion

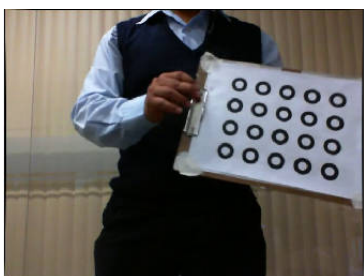


Fig. 15. Imagen calibrada, usando lifecam

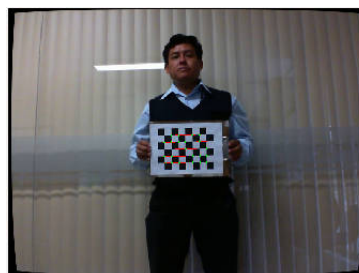


Fig. 16. Imagen calibrada, usando lifecam