# Paper Review: Large Pose 3D Face Reconstruction from a Single Image via Direct Volumetric CNN Regression


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

Master in computer Science, Catolic University of San Pablo
Arequipa, Perú
wildr.10@gmail.com roxana.soto.barrera@gmail.com



**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(2), 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.


## 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.

**Gaussian Filter.-** 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.

## Threshold Segmentation

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:


Figura 1: Image output after Gaussian filter

### 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:

**Canny edge detector** The 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

Figura 2: Results of global segmentation: circles pattern (left) and ring pattern (right)

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(can).

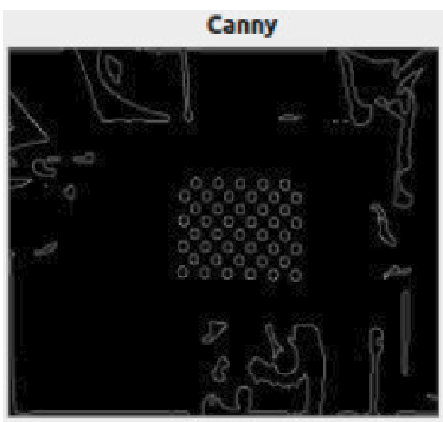



Figura 3: The Canny edge detector applied to a color photograph of a rings pattern.

## Region of Interest

In this stage 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:

- 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.

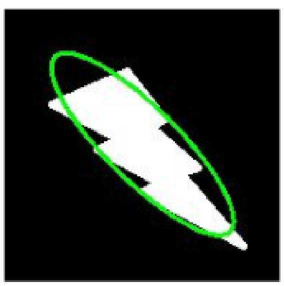
## Canny Algorithm - detection of edges

A variety of algorithms have been developed to help solve this problem. The Canny algorithm is used to detect all the existing edges in an image. This algorithm is considered one of the best methods of detection of contours by using convolution masks and based on the first derivative. Contour points are like areas of pixels where there is a sudden change of gray level. In the treatment of images, we work with pixels, and in a discrete environment, it is so in the Canny algorithm masks are used, which represent approximations in finite differences.

After doing the thresholding and all the filters we are applying the canny filter for our rings images, but we got more noise when we are going to fit the ellipses and that's why we are avoiding to use this method for rings pattern as we were testing with different videos we empirically achieve that for our example.

## Recognition of circles and ellipses

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.



Fitellipse de una región

Figura 4: Fit Elipse from an image)

## Noise Reduction

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.

### 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.

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

### 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.

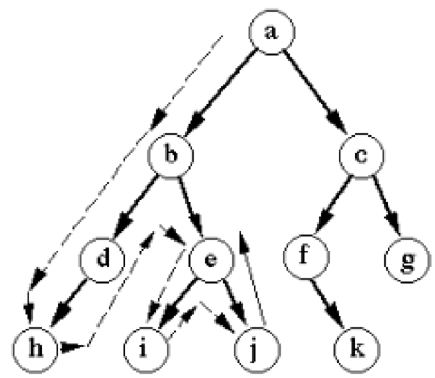



Figura 5: A DFS algorithm for a graph describing the track)

### 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.

### 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

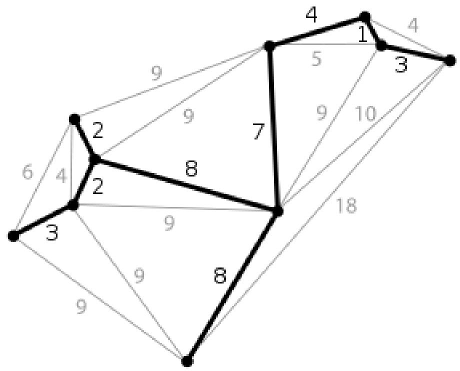



Figura 6: A planar graph and its minimum spanning tree. Each edge is labeled with its weight, which here is roughly proportional to its length (from wikipedia).)

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.

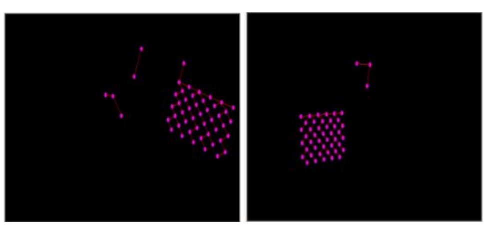


Figura 7: Frame of video with points join with the nearest neighbors to form the graph(from test).)

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.

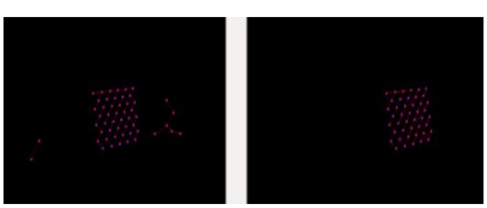


Figura 8: The Frames show, how we choose the graph with largest number of nodes to be the pattern location.)

4. finally we apply MST to eliminate the large edge to discard it.

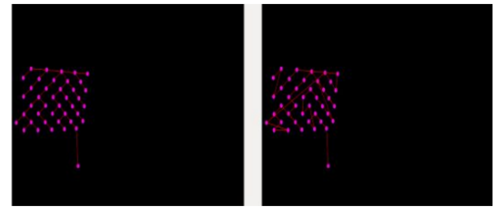


Figura 9: On the frame we can see a problem to fix with Minimal expanding tree for this special case.)

### Tracking

We based of the keypoints and number of points of patron We implemented a method of sort

## Results

## Kalman Filter

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

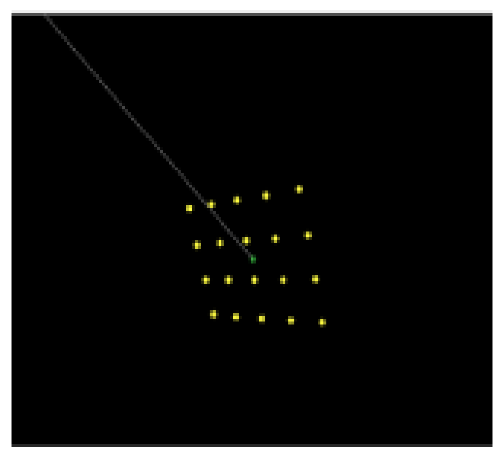


Figura 10: kalman filter to get all point in group

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 (44 the one with circles, 30 with the rings).

## Tracking

This step is made using 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...

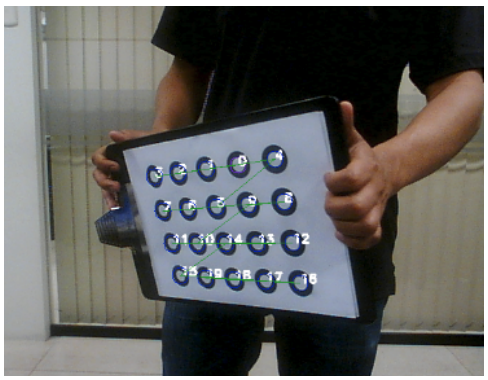


Figura 11: first tracking approach to get the lines on green and numbers

## 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 (3), 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 $P_n = n_t h$ , 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.

$$\begin{bmatrix} X_c orrected = x(1 + k_1 r^2 + k_2 r^4 + k_3 r^6 \\ y_c orrected = y(1 + k_1 r^2 + k_2 r^4 + k_3 r^6 \end{bmatrix} \tag{3}$$

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:

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

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:

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

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.

### Calibration Parameters

Once we calibrate our camera, we test our result with different pattern(P) : chessboard, circle grid, asimetric circle grid, and concentric rings(Rings), so our result are tested with LIfe Cam, getting the following data denoting :$(c_x, c_y)$ is a principal point that is usually at the image center $(f_x, f_y)$ are the focal lengths expressed in pixel units,

| N frames : 50 | | | | | |
|---|---|---|---|---|---|
| P | f$_x$ | f$_y$ | C$_x$ | C$_y$ | RMS |
| 0 | 0 | 0 | 0 | 0 | 004 |
| 1 | 0 | 0 | 0 | 0 | 004 |
| 2 | 9.78406222[illegible] | [illegible] | [illegible] | [illegible] | 0.315517 |
| 3 | 0 | 0 | 0 | 0 | 004 |

Cuadro 1: Table tested on camera Life Cam

The above Table 1 shows full results considering number of frames ( N F ) used for calibration in LifeCam camera. we also included density fuction for the calibration function.

| PadronAjedrez_01 | | NroFrames | 30 |
| --- | --- | --- | --- |
| CameraMatrix | 691.375231 | 0 | 322.338726 |
| | 0 | 690.694387 | 268.250375 |
| | 0 | 0 | 1 |
| RMS | 0.588431634 | | |
| Time(ms) | 10.8052 | | |

| PadronAjedrez_02 | | NroFrames | 30 |
| --- | --- | --- | --- |
| CameraMatrix | 543.6087974 | 0 | 286.261332 |
| | 0 | 542.951142 | 176.25501 |
| | 0 | 0 | 1 |
| RMS | 0.464893359 | | |
| Time(ms) | 11.6288 | | |

Figura 12: resultado

| Ring_01 Simetrico | | NroFrames | 30 |
| --- | --- | --- | --- |
| CameraMatrix | 715.726597 | 0 | 453.863777 |
| | 0 | 712.11748 | 354.352896 |
| | 0 | 0 | 1 |
| RMS | 7.54 | | |
| Time | | | |

| Ring_02 Simetrico | | NroFrames | 50 |
| --- | --- | --- | --- |
| CameraMatrix | 501.269921 | 0 | 324.569284 |
| | 0 | 503.930261 | 189.466954 |
| | 0 | 0 | 1 |
| RMS | 0.55801105 | | |

Figura 13: resultado

## Conclusions

## Conclusions

## Conclusions

## 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

| Circulos_01 Asimetrico1 | | | |
| --- | --- | --- | --- |
| | | NroFrames | 30 |
| CameraMatrix | 735.232922 | 0 | 365.312003 |
| | 0 | 747.453121 | 270.416452 |
| | 0 | 0 | 1 |
| RMS | 0.46032502 | | |
| Time | 0.00E+00 | | |

| Circulos_01 Asimetrico2 | | NroFrames | 30 |
| --- | --- | --- | --- |
| CameraMatrix | 491.667249 | 0 | 335.705767 |
| | 0 | 494.66104 | 173.704643 |
| | 0 | 0 | 1 |
| RMS | 0.22447361 | | |
| Time | 7.24602e-310 | | |

Figura 14: resultado

experiment.

## source code github repository:

https://github.com/roxanasoto/P1$_{C}alibracionCamara$

## References

Canny edge detector wikipedia. https://en.wikipedia.org/wiki/Canny$_{e}dge_{d}etector. Accessed : 2019-08-10.$

opencv (2018). OpenCV web page. https://opencv.org/about.html.

Szeliski, R. (2010). *Computer vision: algorithms and applications*. Springer Science & Business Media.