Research, Society and Development, v. 11, n. 14, e173111436168, 2022
(CC BY 4.0) | ISSN 2525-3409 | DOI: http://dx.doi.org/10.33448/rsd-v11i14.36168

Automatic Template Detection for Camera Calibration

Detecção Automática de Modelo para Calibração de Câmera

Detección Automática de Modelo para la Calibración de la Cámara

Received: 10/05/2022 | Revised: 10/12/2022 | Accepted: 10/18/2022 | Published: 10/23/2022

Marrone Silvério Melo Dantas
ORCID: https://orcid.org/0000-0002-7927-8472
Universidade Federal de Pernambuco, Brazil
E-mail: marrone.dantas@gprt.ufpe.br
Daniel Bezerra
ORCID: https://orcid.org/0000-0002-3839-3642
Universidade Federal de Pernambuco, Brazil
E-mail: daniel.bezerra@gprt.ufpe.br
Assis T. de Oliveira Filho
ORCID: https://orcid.org/0000-0001-9873-6929
Universidade Federal de Pernambuco, Brazil
E-mail: assis.tiago@gprt.ufpe.br
Gibson Barbosa
ORCID: https://orcid.org/0000-0001-9023-4019
Universidade Federal de Pernambuco, Brazil
E-mail: gibson.nunes@gprt.ufpe.br
Iago Richard Rodrigues
ORCID: https://orcid.org/0000-0002-8242-9059
Universidade Federal de Pernambuco, Brazil
E-mail: iago.silva@gprt.ufpe.br
Djamel H. J. Sadok
ORCID: https://orcid.org/0000-0001-5378-4732
Universidade Federal de Pernambuco, Brazil
E-mail: jamel@gprt.ufpe.br
Judith Kelner
ORCID: https://orcid.org/0000-0002-2673-5887
Universidade Federal de Pernambuco, Brazil
E-mail: jk@gprt.ufpe.br
Ricardo Souza
ORCID: https://orcid.org/0000-0001-5378-4732
Ericsson Research, Brazil
E-mail: ricardo.s.souza@ericsson.com

Abstract
Camera calibration is the process of extract the intrinsic and extrinsic parameters of a camera. Those parameters guide the 3-dimensional localization into relation to the 2-dimensional space from the images acquired by the camera. The 3-dimensional correlation can be generated with an object with known measures, being the most common checkerboard for this purpose. From these checker- boards, the usual approach extracts the position of the inner points, equivalent to the corners of the squares, to generate this correlation. A broad range of algorithms tries to find those points on the image. Still, usually, they require previous knowledge about the dimensions of the image, the pattern distribution, or even the pattern type. In some scenario, maybe is difficult, or impossible, to implement such precise solution, targeting these limitations our work proposes a two-step end-to-end convolutional neural network architecture that processes the corner detection on a unique flow. Our proposal is agnostic to checkerboard size, pattern disposal, and positioning. In our work, first, a segmentation CNN extracts only the checkerboard from the input image (CheckerNet); from the extracted checkerboard, we extract the corner points with a corner detection CNN (Point- Net). The PointNet also works as a segmentation CNN, and the generated points are heatmaps related to points on the checkerboard corners. We performed post-processing with a K-Means-based clustering to convert those heatmaps into single positions (x,y) from the image. We compare our proposed method with the other well-known convolutional neural networks used for corner detection MATE and CCDN. For the evaluation, two datasets were used: GoPro e uEye. Our method provides better results in both datasets, reducing missed corners, double detections, false positives, and competitive results on pixel accuracy.

Keywords: Template detection; Camera calibration; Deep learning.

Resumo
A calibração da câmera é o processo de extrair os parâmetros intrínsecos e extrínsecos de uma câmera. Esses parâmetros orientam a localização tridimensional em relação ao espaço bidimensional a partir das imagens adquiridas pela câmera. A correlação tridimensional pode ser gerada com um objeto com medidas conhecidas, sendo o tabuleiro de xadrez mais comum para este fim. A partir desses tabuleiros, a abordagem usual extrai a posição dos pontos internos, equivalentes aos cantos dos quadrados, para gerar essa correlação. Uma ampla gama de algoritmos tenta encontrar esses pontos na imagem. Ainda assim, geralmente, eles exigem conhecimento prévio sobre as dimensões da imagem, a distribuição do padrão ou até mesmo o tipo de padrão. Em algum cenário, talvez seja difícil, ou impossível, implementar uma solução tão precisa, visando essas limitações, nosso trabalho propõe uma arquitetura de rede neural convolucional de duas etapas que processa a detecção de canto em um fluxo único. Nossa proposta é agnóstica ao tamanho do tabuleiro de xadrez, disposição do padrão e posicionamento. Em nosso trabalho, primeiro, uma segmentação CNN extrai apenas o tabuleiro de damas da imagem de entrada (CheckerNet); do tabuleiro de damas extraído, extraímos os pontos de canto com uma CNN de detecção de canto (Point-Net). O PointNet também funciona como uma CNN de segmentação, e os pontos gerados são mapas de calor relacionados a pontos nos cantos do tabuleiro de xadrez. Realizamos o pós-processamento com um agrupamento baseado em K-Means para converter esses mapas de calor em posições únicas (x,y) da imagem. Comparamos nosso método proposto com outras redes neurais convolucionais conhecidas usadas para detecção de cantos MATE e CCDN. Para a avaliação, foram utilizados dois conjuntos de dados: GoPro e uEye. Nosso método fornece melhores resultados em ambos os conjuntos de dados, reduzindo cantos perdidos, detecções duplas, falsos positivos e resultados competitivos em precisão de pixel.
Palavras-chave: Detecção de modelo; Calibração de câmera; Aprendizagem profunda.

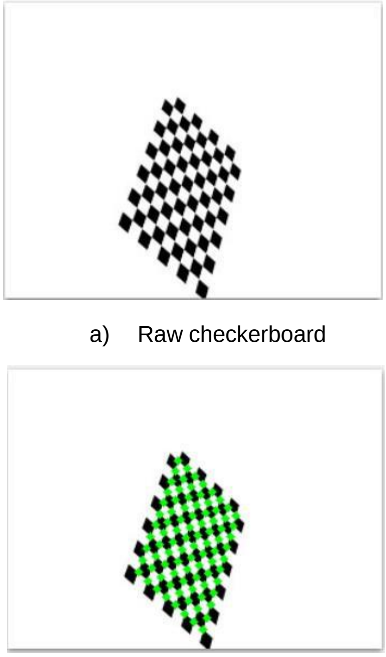
Resumen
La calibración de la cámara es el proceso de extraer los parámetros intrínsecos y extrínsecos de una cámara. Esos parámetros guían la localización tridimensional en relación con el espacio bidimensional de las imágenes adquiridas por la cámara. La correlación tridimensional se puede generar con un objeto de medidas conocidas, siendo el damero más común para este fin. De estos tableros de ajedrez, el enfoque habitual extrae la posición de los puntos interiores, equivalentes a las esquinas de los cuadrados, para generar esta correlación. Una amplia gama de algoritmos intenta encontrar esos puntos en la imagen. Aún así, por lo general, requieren conocimientos previos sobre las dimensiones de la imagen, la distribución del patrón o incluso el tipo de patrón. En algún escenario, tal vez sea difícil, o imposible, implementar una solución tan precisa, teniendo en cuenta estas limitaciones, nuestro trabajo propone una arquitectura de red neuronal convolucional de extremo a extremo de dos pasos que procesa la detección de esquinas en un flujo único. Nuestra propuesta es independiente del tamaño del tablero de ajedrez, la eliminación de patrones y el posicionamiento. En nuestro trabajo, primero, una CNN de segmentación extrae solo el tablero de ajedrez de la imagen de entrada (CheckerNet); Del damero extraído, extraemos los puntos de las esquinas con una CNN de detección de esquinas (Point-Net). PointNet también funciona como una CNN de segmentación, y los puntos generados son mapas de calor relacionados con puntos en las esquinas del tablero de ajedrez. Realizamos un procesamiento posterior con un agrupamiento basado en K-Means para convertir esos mapas de calor en posiciones únicas (x, y) de la imagen. Comparamos nuestro método propuesto con las otras redes neuronales convolucionales bien conocidas utilizadas para la detección de esquinas MATE y CCDN. Para la evaluación, se utilizaron dos conjuntos de datos: GoPro e uEye. Nuestro método proporciona mejores resultados en ambos conjuntos de datos, reduciendo esquinas perdidas, detecciones dobles, falsos positivos y resultados competitivos en precisión de píxeles.
Palabras clave: Detección de modelos; Calibración de cámara, Aprendizaje profundo.

1. Introduction

Corner detection remains an essential task when related to computer vision techniques. Usually, those corner points are detected on an image, a two-dimensional representation of the world. The correlation between those points and the real world can be used in various applications like motion analysis, image registration, image matching, object recognition (Dantas et al. 2002; Dutta et al., 2008). An optical camera is a device that acquires a correlation between a three-dimensional environment and a two- dimensional image, returning the color, intensity, and spatial information. The correlation process between those points is referred to as the process of calibration.

A traditional camera calibration method is based on a known structure, with high precision between space points and image points, usually in the shape of a geometric pattern (Qi et al, 2010). Most applications use a checkerboard as their geometric pattern to allow calibration. The solution must detect the checkerboardt’s corners, then these are fitted on a camera model, and the calibration data is acquired (Hartley & Zisserman, 2004). The acquired data is the so-called internal (intrinsic) and external (extrinsic) parameters. The internal parameters depend on the specific camera model. These parameters encompass focal length, image sensor format, and principal point. The external parameters define the relation between the camera and the world or different cameras (He et al., 2006). Figure 1 depicts a checkerboard and a set of detected points.

Figure 1: Checkerboard and detected points. In (a), we have the raw checkerboard with no mark. In (b), we have an example of manually marked detect points on the same checkerboard.



a) Raw checkerboard

b) Checkerboard with points marked

Source: Author (2022)

As seen o Figure 1, first, we have the raw image on (a), then the points (in green) concerning the corners (b). Most of existing solutions for checkerboard detection often require a complex and sometimes a high level of expertise for their correct application. For instance, (Duda & Frese, 2018) proposes a method for corner detection that requires a nine steps approach. Despite offering satisfactory results, most detection results require a further fine-tuning process and need some special conditions to work correctly, such as the presence of adequate lightning and rotation (Zhang & Xiong, 2017). One way for solving these limitations is the application of Convolutional Neural Networks (CNN's). While previous approaches rely on the manual feature extraction of complex compound solutions, CNN automatically extracts the important features, generating a single and simpler approach for the decision and a more generic solution (Albawi et al., 2017). Think of CNNs as a natural and well-integrated solution path; they are applied across a broad range of applications, and in many cases they represent the state-of-art, especially in the context of computer vision tasks (Li et al., 2021).

Despite their widespread application and success, as far as we know, not many works applied CNNs to solve the calibration problem. MATE (Donné et al., 2016) and CCDN (Chen et al., 2018) are no exceptions. But each of these two models suffers from some limitations. On the one hand, MATE adopts a straightforward CNN, which can lead to errors for more complex data. On the other hand, CCDN is tough and can leak concerning response time. In this paper, we propose an alternative based on CNN. It improves the accuracy of corner detection compared to MATE and CCDN (Chen et al., 2018).

2. Related Studies

The task of corner detection is essential in many tasks. As part of computer vision techniques, it has received much interest over recent years, with contributions dating back to the '70s (Dutta et al., 2008). Previous works mainly focused on solving the corner detection problem and, in many cases, did not work appropriately with checkboard patterns. These limited solutions include the well-known HARIS (Harris & Stephens, 1988). Nowadays, some commercial and public solutions try to handle this limitation. A case in point is the OpenCV implementation (Zhang, 2000), where a set of erosions degrades the patterns, then reconstructed as a set of points. While robust, OpenCV requires preliminary data regarding the checkerboard and can leak information when dealing with complex scenes. OCamLib (Scaramuzza et al., 2006) improves the results of OpenCV with a set of features for camera distortion lens and blur. Another important benchmark is ROCHADE (Placht et al., 2014), an algorithm that uses features, including gradient calculation and threshold computation, among many others, to find the inner corners. Next, these are refined based on a proposed method (Lucchese & Mitra, 2002). From the same contributors, following the ROCHADE design, OCPAD emerged to improve ROCHADE (Fuersattel et al., 2016). It allows the detection of the corners, even for cases where the checkerboard is occluded.

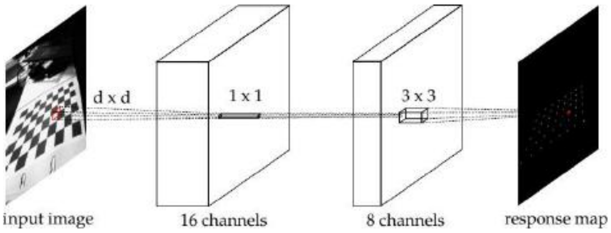
Although some of the previously cited works achieved many significant results, they require preliminary information about the environment and patterns, such as the checkerboard size and others. Unfortunately, only a few little works were concerned with removing the required preliminary information needed prior to calibration. Previous works propose using classical machine learning as an additional step, such as FASTER (Rosten et al., 2010), using a classifier based on a multilayer perceptron. However, the necessity of feature extraction e and feature engineering skills is a required step. More recently, (Zhang & Xiong, 2017) used the Rayleigh difference distribution, similar to a Gaussian distribution, to determine if the corner is contained in the distribution using a threshold. The work of (Pedra et al., 2013) uses classic methods to obtain the data from the checkerboard but also implements a neural network for camera calibration. More recently (Butt & Taj, 2022), proposed a new novel for the calibration parameters, using the so-called camera projection loss (CPL), focusing on reconstructing 3D points. While presenting excellent results, it demands high computational power and some additional processing. (Zhang et al., 2021), uses a similar approach, but focusing on the noise and focusing issues, the authors propose a framework for data enhancement, feature extractions, and finally, parameter estimation. (Pak et al., 2022) proposes a workflow for calibration based on machine learning, relying on phase-shifted cosine patterns, demonstrating promising results in the submillimeter range.

In summary, as far as our review of state-of-the-art goes, there is a lack of proposals that evaluate corner detection using CNN or deep learning techniques. There is no exact reason why these studies are not so popular, maybe because the classical methods have a great outcome or the difficulties from getting such small features as a set of corners. However, even in these scenarios, some works show promising results. MATE (Donné et al., 2016) and CCDN (Chen et al., 2018) are notable contributions. Since the two methods represent a basis for our study, we will briefly explain each.

2.1 MATE

The work "MATE: Machine Learning for Adaptive Calibration Template Detection" (Donné et al., 2016) proposes a single CNN to detect the so-called template, the group o corner points from the checkerboard. The MATE architecture is composed of three convolutional layers, where the last one operates as an output activation. Figure 2 shows the structure of the proposed CNN.

Figure 2: MATE Architecture. Adapted from.



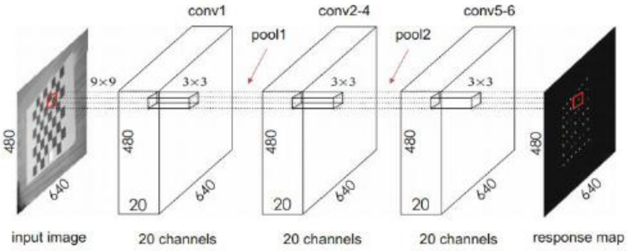
Source: Donné et al. (2016).

As seen of Figure 2, from left to right, the object of the first layer is to obtain the information covering the corner sizes and possible shape changes. The next combines this information into a heatmap in order to predict the correct points positions. The training process used a private dataset, generated by the team, containing 85 images, manually annotated. To complement their dataset, the authors applied augmentation techniques, simulating distortion, lens and noise. A process of non-maximum suppression refines the final points to reduce the total number of false positives.

2.2 CCDN

The work (Chen et al., 2018) also relies on using CNNs but introduces some remarkable improvements concerning the MATE architecture. CCDN contains four convolutional layers and two new max-pooling layers. Besides adding the max-pooling layers, the architectures increase in depth, from 24 to 60 filters. The overall architecture can be seen in Figure 3.

Figure 3: CCDN Architecture. Adapted from.



Source: Chen et al. (2018)

As seen in Figure 2, the CCDN follows a similar structure to the MATE, but after each convolution, a max-pooling layer (poo1 and pool2) was added. The authors claimed that adding the max-pooling layers improved the generalization power, summarizing the resulting feature mapping on the convolutional outputs. Like the MATE architecture, it still required heavy post-processing for an ideal result, reducing the false-positive results. Besides the previous non-maximum suppressing, the authors added support for a dynamic threshold and a clustering process. The dynamic threshold sets a level for decision-making, determining if a pixel is contained on the set of corner points. The threshold was determined based on half of the maximum activation level. Clustering refers to the process applied when even after the filtering, the solution ended up with many false positives. Clustering grouped a set of near detected points, and a point was only valid if the cluster that originated it contained more than ten samples.

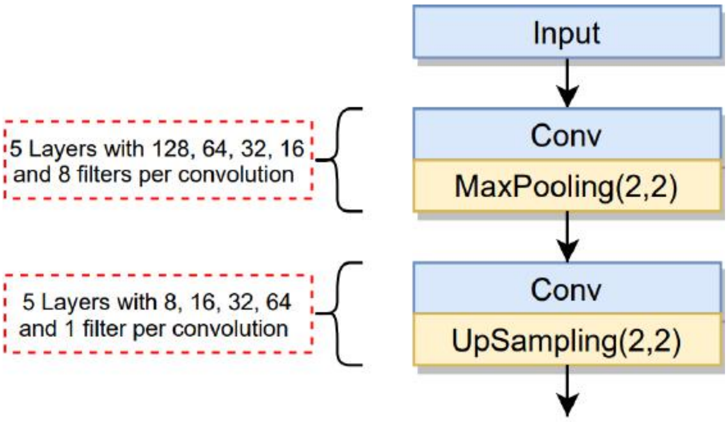
3. Proposed Method

Previous discussed approaches used the concept of convolutional neural networks but adopted simple architectures. This work considers that perhaps the environment was too complex for these CNNs, causing some levels of under- fitting. Our approach combines solution object segmentation and a point detection network to avoid this type of limitation. The first level of our solutions is a segmentation based on CNN, named CheckerNet, where only the checkerboard is segmented for the image. Then with this output, we proceed towards point detection, named PointNet, and finally the post-processing process with a clustering step. These steps are explained next.

3.1 CheckerNet

Our approach follows a straightforward process; It adopts an architecture based on convolutions and up-sampling. We call the segmentation architecture ChekerNet. Figure 4 shows the proposed architecture.

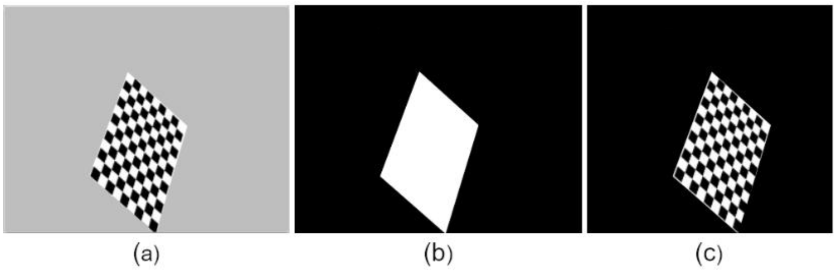
Figure 4: Proposed segmentation architecture, the CheckerNet.


Input
5 Layers with 128, 64, 32, 16 and 8 filters per convolution
Conv
MaxPooling(2,2)
5 Layers with 8, 16, 32, 64 and 1 filter per convolution
Conv
UpSampling(2,2)

Source: Authors (2022).

As seen in Figure 4, ChekerNet consists of ten layers. The initial five layers work as an encoder responsible for condensing the features. The next four layers operate as a decoder, and finally there is the output, a single activation map of the checkerboard mask. In all the convolution layers, we keep a kernel with a size of 3x3. In the max-pooling and upsampling layers, we used a kernel with a size of 2x2. For this CNN, we expect to input an image containing a checkerboard from the proposed CNN and output the mask describing the region of interest, having the cropped part of the checkerboard. Figure 5 shows an example of the expected inputs.

Figure 5: Example of inputs and outputs of the Chekernet. (a) Raw input of the Checkernet, (b) Generated mask as output, and finally (c) The cropped checkerboard.



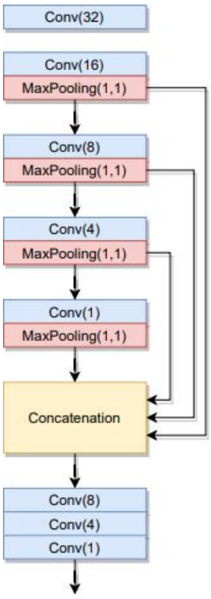
Source: Authors (2022).

As seen in Figure 5, as input, we have a raw image without the checkboard (a); on the center, we have the mask, indicating the checkerboard boundaries (b), and finally, on the right, the checkerboard segmented (c).

3.2 Point Detection

The corner detection process, also called point detection, is performed by another CNN. We call this separate CNN PointNet. In terms of PointNet input, we expected the image segmented with just the checkerboard. PointNet design has been inspired by CCDN and DenseNet (Huang et al., 2016). We chose to apply a basic implementation of stacked convolutional layers, skip the convolution outputs, and combine multi-levels of feature extraction. Similar to the original work by DenseNet, the inclusion of skip connections can help with vanishing-gradient, feature propagation, and feature reuse. Figure 6 shows the stages of the proposed architecture.

Figure 6: PointNet architecture.

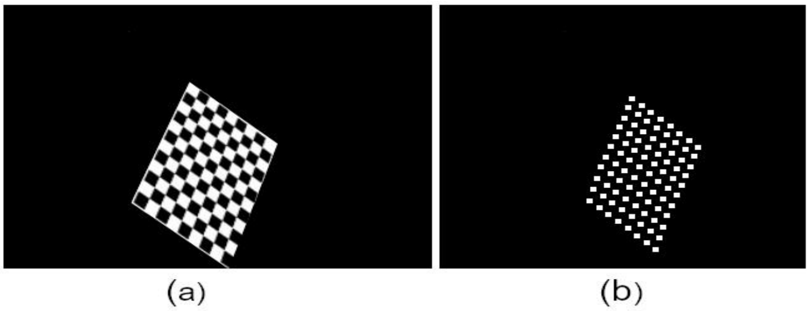


Source: Authors (2022).

As seen in Figure 6, we maintain the basic stack layers, and for each max-pooling output, we concatenate and proceed to the final processing. We applied zero padding to fit the shape from each skipped connection for the concatenation and the final activation map. We kept a kernel size 3x3 on all the convolutions and 1x1 for the max pooling.

PointNet takes, as its input, the segmented image, where the checkerboard is in the foreground, with no background. The output is the activation map containing the checkerboard corners. Figure 7 shows an example of the desired input and output.

Figure 7: PointNet input and output. (a) We have the input of a segmented image, and as output (b), we have the desired activation map. In the image, we increase the region of interest on the activation map for better visualization.



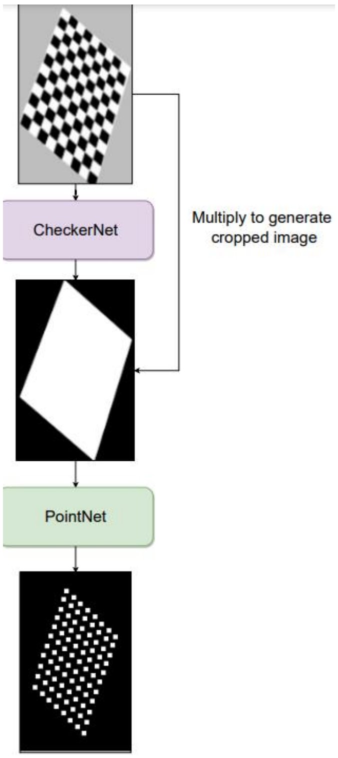
Source: Authors (2022).

As seen in Figure 7, we have the input image of the checkerboard on the left (a) and the expected output, the corner points (b).

3.3 CheckerNet and PointNet Blending

As mentioned earlier in this text, our proposed method relies on reducing the search space for the final decision of the activation map the corner detection. Our general workflow is based on the concatenation of both CNNs and decision- making. Innately the CheckerNet is obtaining the segmented image and feeding the PointNet for the corner detection. Figure 8 presents the flowchart adopted for corner detection.

Figure 8: Workflow based on the concatenation of CheckerNet and PointNet.



Source: Authors (2022).

As seen in Figure 8, from top to bottom, first, we start with the raw image. The following step is the extraction of the checkerboard by segmentation. It is valid to notice that the output of the CNN is just the mask containing the checkerboard to feed the following CNN, namely, PointNet. We generate the new images by multiplying by the masks. Finally, we obtained the output image, offering the activation map for the corners on the image.

3.4 Post-Processing

Once we obtain the activation map, we need to maintain only the relevant corner detected and positions as the final step. In this phase, false positives must be removed. We used a clustering technique based on the so-called expansion search to achieve this. One can use a wide range of checkerboard dimensions for the calibration process for columns. Regarding clustering, we needed techniques that do not require prior knowledge of the number of clusters, in our case, valid corners.

We added two capabilities to the expansion search clustering based on these limitations. First, we create a set of positions (x and y) of the activated pixel on the output activation map. We ordered the set of positions, and from the first-pixel position, we looked for the pixels at a minimum distance and added them to the cluster. We performed the same search for each newly added point until there was no point left in the minimum distance range. The added points are removed from the initial set of points, and we repeat the process until all the points are allocated. The final position is stipulated by the average of all points in the cluster, or centroid. Algorithm 1 shows the proposed method.

Algorithm 1: Expansion Clustering.

Expansion Clustering

```
S = set of points (x, y)
Clusters = set of clusters
M = Minimum distance
while S is not empty do
  for Each point p in S do
    Create a new cluster C
    Add the point p in the cluster C
    while New point is added on C do
      for Each point $p_i$ in C do
        for Each point $p_j$ in S do
          if $p_i$ != $p_j$ and EuclidianDistance($p_i$,$p_j$)<=M then
            Add the point $p_j$ to the cluster C
            Remove $p_j$ from S
          end if
        end for
      end for
    end while
    Add cluster C into Clusters
  end for
end while
Return the centroids of each cluster C in Clusters
```

Source: Authors (2022).

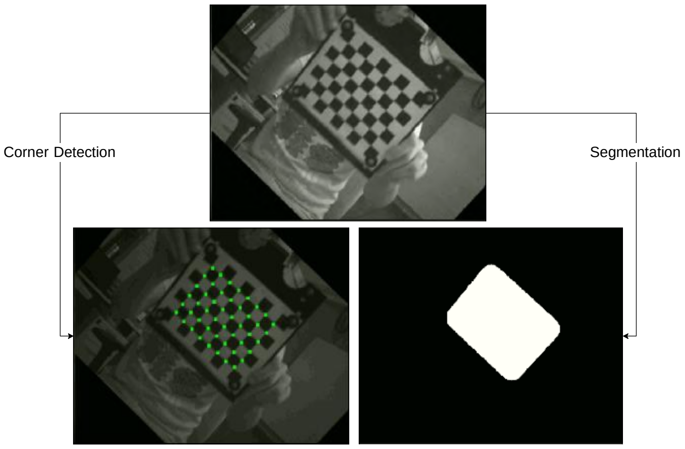
Due to the nature of the CNNs, we may produce some artifacts, such as activated points, in the minor region, generating invalid clusters. To reduce the occurrence of these artifacts, we set a threshold of minimum detected points per cluster. In our experiments, the threshold of 20 reproduced the best results.

.4. Dataset

To the best of our knowledge, there is no available public dataset that contains checkerboard annotated information related to segmentation or corner location. The two datasets, uEye and GoPro, used for validation on both papers, from the ROCHADE (Placht et al., 2014) project, do not have any ground-truth type. To overcome this limitation, we used an alternative dataset provided by the Vision Group, a benchmark from the RGB-D Slam, a solution for 3D odometry (Sturm et al., 2012).

We selected the benchmarking freiburg1_ir_calibration, which contains 1763 images from a checkerboard in different positions, distances, and rotations. We manually marked each corner from this dataset, and from each corner, we generated the segmentation masks. Figure 9 illustrates the generated images.

Figure 9: Generated datasets.



Source: Sturm et al., *(2012).*

As seen in Figure 9, on the left side, we have the ground truth of the PointNet, targeting the corners. On the right side, check the segmentation ground truth for CheckerNet. In addition to the training process, we also applied data augmentation. It included rotations from the center of the image from -90 to 90 degrees, vertical flip, brightness, and contrast variations.

5. Experiments

We first trained our segmentation CNN. For the training, we used a fixed image size with 480 pixels height and 640 pixels length. With this size of image, we ended up with 124785 parameters. As activation, we kept the ReLU activation in all intermediary layers. We use the sigmoid activation for the output, and we apply the binary cross-entropy for loss. The prediction is classified into two types, background (zero values) and foreground or checkerboard (one value).

We split our original dataset randomly into 80%, 10% for validation, and 10% for conducting tests. We maintained the same split for the segmentation and corner detection. We trained until the validation did not improve for at least ten epochs. Table 1 lists the results.

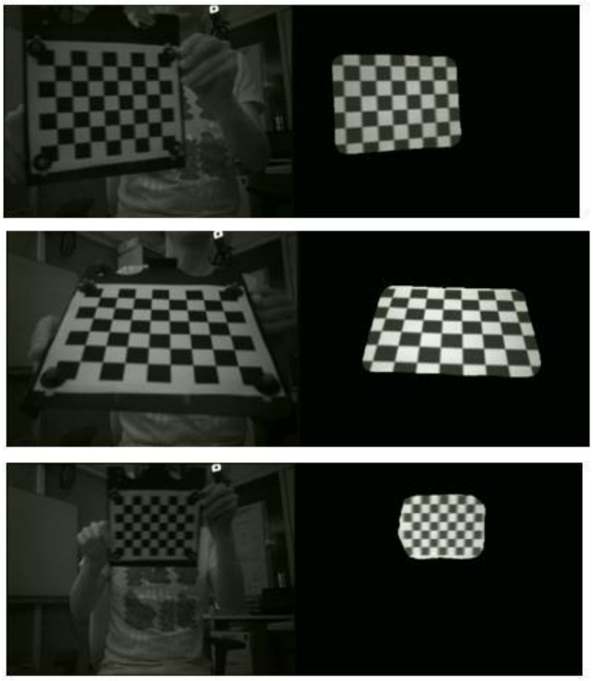
Table 2: Results CheckerNet.

Architecture	**Train Loss**	**Train IOU**	**Validation Loss**	**Validation IOU**
CheckerNet	0.017	0.993	0.008	0.996

Source: Authors (2022).

We have drawn attention to the value of the IOU for the CheckerNet, in Table 1, which performs above 0.99. The main objective is to acquire a comprehensive checkerboard from the input images under a good performance level. We believe that the present contribution achieved this design goal. Figure 10 shows some examples of segmented images.

Figure 10: Predicted images on CheckerNet.



Source: Sturm *et al.,* (2012).

As seen in Figure 10, on the left column, we have the input images; on the right, we have the segmented images, with only the checkerboards. With the trained CheckerNet, we proceeded next to produce the new dataset for use by PointNet. This is an important step; note that we used the segmentation generated by the CheckerNet, not the manually annotated, to evaluate the performance of the segmentation.

We followed similar approaches for the training of Point-Net. We adopted the same dataset split levels for training, test, and validation. All the intermediary layers contained ReLU activation. We used a sigmoid activation for the feature map activation and binary cross-entropy as a loss. Being a smaller CNN, PointNet ended with 11162 parameters. Table 2 shows the results.

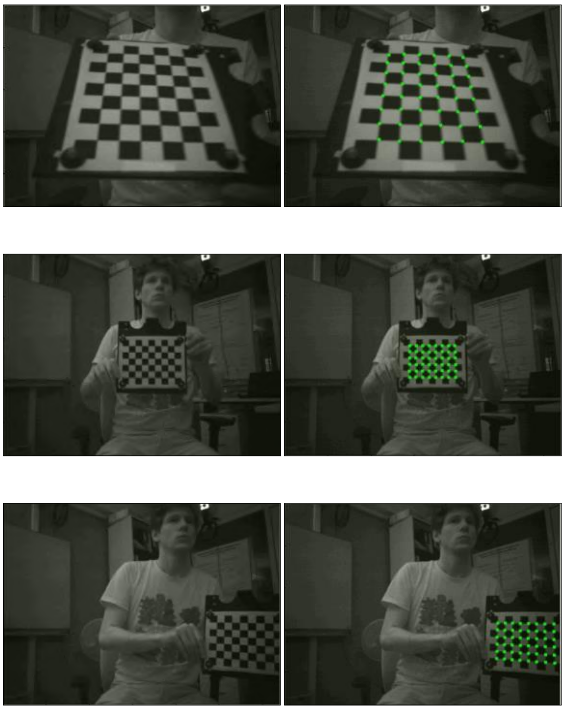
Table 2: Results PointNet.

Architecture	Train Loss	Train IOU	Validation Loss	Validation IOU
PointNet	0.002	0.998	0.003	0.998

Source: Authors (2022).

It is valid to notice that we keep the same performance on the PointNet, as seen in Table 2, with an IOU above 0.99. Figure 11, offers some examples of our prediction using PointNet, after the clustering process.

Figure 11: Predicted images on PointNet.



Source: Authors (2022).

As seen in Figure 11, we have the input images on the left and the detected point from the corners on the right. We tested the method on the two datasets used on the MATE and CCDN, the GoPro and the uEye, for validation purposes. Since MATE and CCDN do not provide a public implementation, and some parameters are not available, we used the best-published results from each respective paper.

As part of the evaluation task, we added other metrics. The accuracy refers to the minimal distance in pixels from the predicted point to the ground truth. We needed to annotate those two additional datasets manually to generate this metric. We considered a positive case, one where the expected point is at a distance of a maximum of five pixels. Tables 3 and 4 present the results.

Table 3: Results uEye.

Architecture	**Accuracy (px)**	**Missed Corners**	**Double Detection**	**False Positives**
CheckerNet+ PointNet	0.915	0.898	0.000	53
CCDN	0.812	1.169	0.000	93
MATE	1.009	3.065	0.809	492

Source: Authors (2022).

Table 4: Results GoPro.

Architecture	**Accurac y (px)**	**Missed Corners**	**Double Detection**	**False Positives**
CheckerNet+ PointNet	0.617	0.870	0.000	0
CCDN	0.576	0.907	0.000	0
MATE	0.835	4.566	4.566	389

Source: Authors (2022).

On the uEye and GoPro, in Table 3 and 4, we only underperform the CCDN on the accuracy for the GoPro dataset. From the above, one can notice that our proposed method reached the state-of-art on deep learning methods for template estimation. It reduced the missed corners rate in all cases. For instance, the uEye, reduced false positives from 93 to 53.

6. Conclusion

This contribution presents a two-step end-to-end solution based on segmentation and corner detection. Was generated the initial module based on a segmentation network (CheckerNet), enabling the extraction of the checkerboard, limiting the researching area for the corner detector, and providing a more reliable output. The second module (PointNet) was generated based on a stack of layers, improving the previous corner networks, and was based on the DenseNet. The points skipped connection between the layers, trying to keep the information through the layers since we believe that the corners can be information lost in the convolution process.

Finally, a post-processing stage that uses point clustering. In the clustering, we aimed just the hard centroids, centroids with a minimal number of discarded components. The clustering process also tries to create the connection requirements, where all the points in the clusters have other points in the same cluster with a distance minus or equal to one.

We improved state-of-the-art in all datasets, as proven by our comparison to other adopted deep learning architectures. For the uEye, we get we miss 0.898% of the points, and for the GoPro, only 0.870%. The proposed method is agnostic to the pattern sizes in columns and rows and does not require prior information concerning the used pattern. The method lacks accuracy per pixel, but we believe this results from our generalization, where we have a much more complex decision region, generating a broad region for approximation and misconceptions.

Acknowledgments

This research was supported by the Ericsson Research (Brasil), Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPq) and Coordenação de Aperfeiçoamento de Pessoal de Nível Superior (CAPES).

References

Albawi, S., Mohammed, T. A., & Al-Zawi, S. (2017). Understanding of a convolutional neural network. *2017 International Conference on Engineering and Technology (ICET)*, pp. 1–6. doi:10.1109/ICEngTechnol.2017.8308186

Butt, T. H., & Taj, M. (2022). Camera Calibration Through Camera Projection Loss. *ICASSP 2022 - 2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP*), p. 2649–2653. doi:10.1109/ICASSP43922.2022.9746819

Chen, B., Xiong, C., & Zhang, Q. (2018). CCDN: Checkerboard Corner Detection Network for Robust Camera Calibration. *Intelligent Robotics and Applications*, pp. 324–334. doi: 10.1007/978-3-319-97589-4_27

Dantas, M., Dreyer, P., Bezerra, D., Reis, G., Souza, R., Lins, S., … Sadok, D. (2022). Video object segmentation for automatic image annotation of ethernet connectors with environment mapping and 3D projection. Multimedia Tools and Applications. doi:10.1007/s11042-022-13128-z

Donné, S., De Vylder, J., Goossens, B., & Philips, W. (2016). MATE: Machine Learning for Adaptive Calibration Template Detection. *Sensors (Basel, Switzerland)*, 16(11), 1858. doi:10.3390/s16111858

Duda, A., & Frese, U. (2018). Accurate Detection and Localization of Checkerboard Corners for Calibration. *29th British Machine Vision Conference (BMVC-29)*, 126. doi: http://bmvc2018.org/contents/papers/0508.pdf

Dutta, A., Kar, A., & Chatterji, B. N. (2008). Corner Detection Algorithms for Digital Images in Last Three Decades. *IETE Technical Review*, 25(3), 123-133. doi:10.4103/02564602.2008.10876651

Fuersattel, P., Dotenco, S., Placht, S., Balda, M., Maier, A., & Riess, C. (2016). OCPAD - Occluded checkerboard pattern detector. *2016 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pp. 1–9. doi:10.1109/WACV.2016.7477565

Harris, C. & Stephens, M. (1988). A Combined Corner and Edge Detector. *Proceedings of the 4th Alvey Vision Conference*, pp. 147-151. doi: dx.doi.org/10.5244/c.2.23

Hartley, R., & Zisserman, A. (2004). Multiple View Geometry in Computer Vision (2nd ed.). doi:10.1017/CBO9780511811685

He, X., Zhang, H., Hur, N., Kim, J., Wu, Q., & Kim, T. (2006). Estimation of Internal and External Parameters for Camera Calibration Using 1D Pattern. *2006 IEEE International Conference on Video and Signal Based Surveillance*, pp. 93–93. doi:10.1109/AVSS.2006.48

Huang, G., Liu, Z., & Weinberger, K. Q. (2016). Densely Connected Convolutional Networks. *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 2261–2269. doi:10.1109/CVPR.2017.243

Li, Z., Liu, F., Yang, W., Peng, S., & Zhou, J. (2021). A Survey of Convolutional Neural Networks: Analysis, Applications, and Prospects. *IEEE transactions on neural networks and learning systems*, pp. 1-12. doi: /10.1109/TNNLS.2021.3084827

Lucchese, L., & Mitra, S. K. (2002). Using saddle points for subpixel feature detection in camera calibration targets. *Asia-Pacific Conference on Circuits and Systems*, 2002(2), 191–195. doi:10.1109/APCCAS.2002.1115151

Pak, A., Reichel, S., & Burke, J. (2022). Machine-Learning-Inspired Workflow for Camera Calibration. *Sensors (Basel, Switzerland)*, 22(18), 6804. doi: 10.3390/s22186804

Pedra, A. V. B. M., Mendonça, M., Finocchio, M. A. F., de Arruda, L. V. R., & Castanho, J. E. C. (2013). Camera Calibration Using Detection and Neural Networks. *IFAC Proceedings Volumes*, 46(7), 245–250. doi:10.3182/20130522-3-BR-4036.00077

Placht, S., Fürsattel, P., Mengue, E. A., Hofmann, H., Schaller, C., Balda, M., & Angelopoulou, E. (2014). ROCHADE: Robust Checkerboard Advanced Detection for Camera Calibration. *Computer Vision (ECCV 2014)*, pp. 766–779. doi: 10.1109/ICVS.2006.3

Qi, W., Li, F., & Zhenzhong, L. (2010). Review on camera calibration. *2010 Chinese Control and Decision Conference*, pp. 3354-3358. doi:10.1109/CCDC.2010.5498574

Rosten, E., Porter, R., & Drummond, T. (2010). Faster and Better: A Machine Learning Approach to Corner Detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 32(1), 105-119. doi:10.1109/TPAMI.2008.275

Scaramuzza, D., Martinelli, A., & Siegwart, R. (2006). A Flexible Technique for Accurate Omnidirectional Camera Calibration and Structure from Motion. *Fourth IEEE International Conference on Computer Vision Systems (ICVS'06)*, pp. 45-45. doi:10.1109/ICVS.2006.3

Sturm, J., Engelhard, N., Endres, F., Burgard, W., & Cremers, D. (2012). A Benchmark for the Evaluation of RGB-D SLAM Systems. International *Conference on Intelligent Robot Systems (IROS)*, 573-580. 10.1109/IROS.2012.6385773

Zhang, J., Luo, B., Xiang, Z., Zhang, Q., Wang, Y., Su, X., Wang, W. (2021). Deep-learning-based adaptive camera calibration for various defocusing degrees. *Opt. Lett*., 46(22), 5537–5540. 10.1364/OL.443337

Zhang, Q., & Xiong, C. (2017). A New Chessboard Corner Detection Algorithm with Simple Thresholding. *Intelligent Robotics and Applications*, 532–542. 10.1007/978-3-319-65292-4_46

Zhang, Z. (2000). A flexible new technique for camera calibration. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(11), 1330–1334. doi:10.1109/34.