Extracting Features and Detecting Objects in Images

Vivek Panchal
School of Computer Science
University of Adelaide
a1774676@adelaide.edu.au

Mahdi Kazemi Moghaddam
Australian Institute for Machine Learning
mahdi.kazemimoghaddam@adelaide.edu.au

ABSTRACT

From autonomous vehicles to video surveillances, extracting features and detecting objects of images and videos in the real world has become increasingly essential. But how is object detection possible you may ask? That's what exactly this paper establishes and reviews, object detection and its techniques that allow for this incredible technology to take place. Ultimately, it will allow to start a research project that will be focused on applying this technology into the field of table tennis. Detecting a table in table tennis comes with its own unique challenges, such as finding the exact location of four corners.

COMPUTER SCIENCE CONCEPT

• **Computer Vision → Classical methods for extracting features and detecting objects**

1 INTRODUCTION

Artificial Intelligence is the 'umbrella' of fields like Computer Vision and Machine Learning. Both fields are interconnected and relied upon each other to push them to their maximum potential, which is indefinite. Humans' visual system is very intelligent to say the least, it is rapid fast, highly accurate and can detect multiple objects with a conscious thought [1]. In the modern age, computers are also very intelligent, with access to large amounts of data, faster GPUs, and intelligent algorithms they can also be trained to classify multiple objects with high accurate in an image. Object detection is a technique that allows us to identify and localise objects through processing of visual information such as an image/video and assigns a label based on the input. The basic structure of object detection model is derived from two parts, an encoder, and a decoder. An input is taken as an image in the encoder, and it is run through a progressive series of blocks and layers that extract statistical features which allow for localisation and labelling of objects. This information is transferred onto the decoder which allows for relatively accurate predictions of bounding boxes and labels for each object [2].

The ground-breaking invention of convolutional neural network (CNN) by deep learning pioneer Yann LeCun in 1980s has completely revolutionised the field of Computer Vision, especially object detection in images. CNN is a type of neural network that can capture patterns with great efficiency in multidimensional spaces. Although object detection has advanced to great heights, there is still a long way to go. Viewpoint variation, objects viewed from different angles may look completely different and therefore, still one of the biggest challenges that is yet to be fully solved [3].

1.2 Motivation

The motivation of pursing object detection, which is a subset of a much larger and widespread topic, Artificial Intelligence lies in its real-world applications. Object detection with deep learning has created cutting-edge technology such autonomous vehicles, once considered

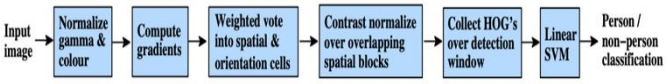
to be impossible. It can be used in video surveillances for safety. Health care settings also utilize this technology for medical imaging. In a nutshell, this technology is truly impressive and is yet to show its full potential. By undertaking this research project focused on table tennis and detecting objects within a sport, allows for visual analytics to take place. It truly enhances the sport for the players to get a visual perspective of their gameplay to improve their performance. And this will kickstart my journey in the field of object detection, and hopefully take it to greater heights.

2 Literature Review

David G. Lowe (2004) conducted a study in which Scale Invariant Feature Transform (SIFT) is presented, a method for extracting distinctive invariant features from images that can be utilised to achieve matching between different views of an object/scene with impressive accuracy. The method at its core achieves distinctiveness by assembling a high-dimensional vector representing the image gradients within a local region of the image. This is very advantages because large quantity of key-points can be extracted, and this allows for robustness amongst all the clutter when extracting smaller objects. Although, the research paper only looks at a monochrome intensity image, which may be a limiting factor as invariant colour features may not be detectable using SIFT. However, this could be a driving force for Lowe to continue his research to a deeper level. Overall, this research paper not only successfully implements SIFT, but it also puts the field of Computer Vision a step ahead by continuing previous research and taking inspiration from Moravec (1981) and many others [4].

In contrast, Dalal and Triggs' (2005) study focused on feature sets for robust visual object detection. By using linear SVM (an algorithm for image classification) based human detection as a test case, the study claims experimentally that by implementing their Histograms of Oriented Gradient (HOG) descriptors, it significantly outperforms existing feature sets for human detection. The method and approach taken to accomplish these exceptional results is quite expensive and therefore, refer to Figure 1 below for a summary of the feature extraction and object detection process.

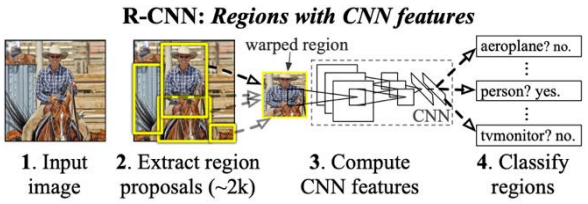
Figure 1: An Overview of feature extraction and object detection process



The results of the study were compared against other existing methods, such as Generalized Haar Wavelets, in which HOG outperformed all of them. According to Dalal and Triggs', the HOG outperformed Haar Wavelets because they utilised 2nd order derivatives and contrast normalise the output vector. Although, the study also discovered that any significant degree of smoothing before calculating the gradient, negatively impacts HOG results. Therefore, based on the results achieved, the gradient should be calculated at the finest available scale. By utilising HOG, they were able to achieve remarkable results against other existing methods, however, it also requires certain factors to play its favour. Such as calculating the gradient at the finest available scale, which may not always work in a real-life scenario as that introduces a lot more factors that may cause unexpected outcomes [5].

In a more recent study, Ross Girshick et al. (2014) use a method they call Regions with Convolutional Neural Networks (R-CNN). This scalable algorithm proved to outperform a previous best result on VOC 2012 dataset, by relatively achieving a mean average precision (mAP) by more than 30%. The method at its core, although involves much more than this, a simplified version of the object detection system is displayed in Figure 2 below.

Figure 2: An Overview of Object Detection System



It also mentions both algorithms, SIFT and HOG discussed above, and how R-CNN is a hierarchical multi-stage process for computing features that is even more informative for visual recognition. This comparison is to be expected as there is a large gap of time between these studies, and this showcases how technology in this sector has improved over the years. A drawback seems to be that labelled data is scarce and the amount available during this study was minimal for training a large CNN and hence, may be a liming factor in determining its full potential [6].

Although these techniques are great and work fine in a real-world scenario with training data, it will be very hard to apply these in this project as there is no data to work with. Therefore, segmentation models in deep learning such as Mask-RCNN is of interest. Mask RCNN is a deep neural network aimed to solve instance segmentation problem in machine learning. Xin Wu at el. (2019) in their study showcase how Mask-RCNN segmentation algorithm has a high time overload and poor accuracy of image segmentation. Therefore, they were able to develop an improved algorithm that was able to deliver increased accuracy of target segmentation and dynamically build semantic maps [7].

2.1 Literature Review - Prototype

The literature review below is focused on creating the prototype, which is detecting the four corners of a table tennis table in a simple image with minimal variation of other factors like angle, blur, colour contrast etc. This image will be shown and further elaborated in the discussion section below.

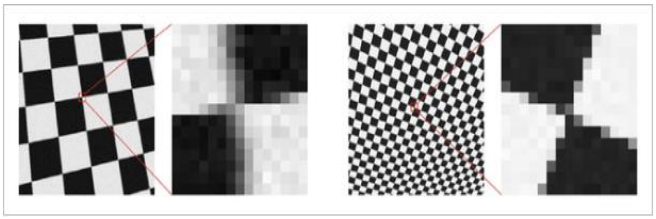
Corners are essentially the primary local feature in an image, they are points usually that have high curvature and appear in brighter intersections of images. Extracting corners in images significantly reduces processing of data. Therefore, it plays a key role in many parts of image processing, including, motion tracking, image matching, and preprocessing stages of capture systems. Herur and Kerur (2018) in their technical review explored the idea of image corner detection in machine vision. They stated the main four problems for highly accurate and precise extraction of corners in images. They are:

- The attitude, position and orientation of the camera with respect to the object
- The interior-orientation of camera
- The fluctuations of the illumination and
- The camera optics.

They also state that the two main conventional techniques for image corner detection involves either to calculate the second derivative of grey-level variation or to apply some sort of convolution-based approach [8].

By using these principles, Wan and Wu (2021) carried out a study on producing a highly accurate and robust deep checkerboard corner detector. Checkerboard corners are routinely used during tasks such as camera calibration, and accurate detection of them is essential. Traditional methods such as those based on the Harris corner detector are usually affected by image artefacts such as noise and blur. The study designed a deep network train it using synthetic data that stimulates the real imaging process. The synthetic checkerboard images training data were produced with exact corner point coordinates under different lightning, noise, blur and camera poses by simulating the real imaging process. These results of the synthetic data can be seen in the figure below:

Figure 3: Two examples from the synthetic training data. The images have different camera poses, resolutions, blur and noise levels



The results of this study were compared to MATLAB camera calibration toolbox, the OpenCV checkerboard corner detectors, and the more recently proposed deep learning-based method and results were favorable relative to these. On average, the results of the study were 14.3% more accurate in detecting corners of the checkboard compared to the other methods [9]. This is a very good example of why training data whether it be, synthetic or real is crucial in object detection as it allows for results to be much more consistent and accurate.

3 Methodology

Harris Corner Detector

Harris Corner Detector is a corner detection operator in OpenCV written as cv.cornerHarris() which uses algorithms to extract corners and infer features of an image. It has been constantly improved and adopted over time in many algorithms to preprocess images, making it of great use in this project as it does not require training data to function. The process of Harris Corner detection algorithm is rather complicated and for the sake of simplicity, it can be split up in 5 distinct steps:

1. Colour to grayscale
2. Spatial derivative calculation
3. Structure tensor setup
4. Harris response calculation
5. Non-maximum suppression

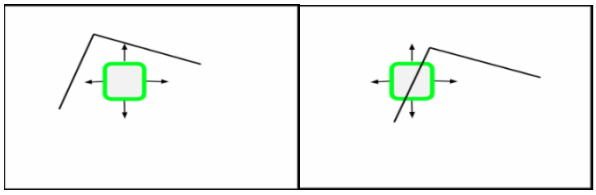
The purpose of colour to grayscale step, which is done is mostly all corner detection algorithm is to simply the algorithm and reduce computation requirements as it is produced from the weighted sums of R, B and G of a coloured image. Conversely, it enhances the processing speed of the image. Steps two to four are complex mathematical steps such as maximizing functions, applying Taylor expansion, finding derivatives and finding the magnitude of eigenvalues to decide whether a region is a corner or not. The last step picks the most optimal values out of these by finding the local maxima as corners within the window which is a 3 by 3 filter.

In order to further improve the quality of results by the Harris Corner Detector, we can also integrate another inbuilt function called Subpixel Corners, or its syntax being cv.cornerSubPix(). It refines the corners detected with sub-pixel accuracy. For example, Harris Corner Detector will detect a pixel of (69, 145), however, by using this function it will allows us to get a much accurate pixel coordinate with high precision such as (69.143, 145.513). It works by using dot product, it solves several of dot product vector equations to estimate the corner position to find a much more precise coordinate.

Shi-Tomasi corner detection

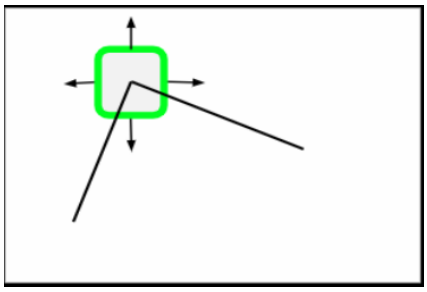
The next methodology used was the Shi-Tomasi corner detection method, which was published by J.Shi and C.Tomasi in their paper “Good Features to Track”. In simpler terms, the algorithm understands that the corners can be detected by looking for significant change in direction. For example, as seen in the figures below, flat region and edges have no major change in direction:

Figure 4: Flat region and edges have no major change in direction



However, if there is a corner, there will be a significant change in direction, as also seen by the figure below:

Figure 5: A corner has a significant change in direction



The mathematical implementation behind this algorithm is a multiple layered process. Firstly, For a window(W) located at (X, Y) with pixel intensity I(X, Y), formula for Shi-Tomasi Corner Detection is:

$$f(X,Y) = \Sigma\,(I(Xk,Yk) - I(Xk + \Delta X, Yk + \Delta Y))2 \;\; where\;(Xk,Yk)\,\epsilon\,W$$

The formula in simpler terms is scanning the image with a window, and if it notices that there is an area where there's a major change any given direction, then it has good intuition of locating a corner. After that, taylor expansion is used to simply the following function, R:

$$R = min(\lambda 1, \lambda 2)$$

$where\ \lambda 1, \lambda 2\ are\ eigenvalues\ of\ resultant\ matrix$

Then the Good Features to Track function or its syntax in python:

goodFeaturesToTrack(gray_img,maxc,Q,maxD).

As seen, the function has four parameters:

- gray_img – Grayscale image with integral values
- maxc – Maximum number of corners we want (give negative value to get all the corners)
- Q – Quality level parameter (preferred value=0.01)
- maxD – Maximum distance (preferred value=10)

Each of these have to be adjusted based on the image being used to produce an optimal output of corners.

Homography

Homography, in mathematical terms is a transformation of a matrix. It is based on linear algebra just like many other important computer vision and computer graphics discoveries. It allows us to shift from one view to another view of the same scene by multiplying the Homography matrix with the points in one view to find their corresponding locations in another view, refer to the figure below.

Figure 6: Homography Transformation Matrix

$$\widehat{P} = HP$$

$$\widehat{P} = \begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & 1 \end{bmatrix} P$$

Homogenous coordinates and project space are interconnect. Homogenous Coordinates are a system of coordinates that are used in the project space. For example, a plane located at Z = 1 in the 3D space in the project space, lines can cut through the origin of the 3D space and intersect at Z = 1 plane form points in the projective space. This provides an advantage over cartesian coordinates because it allows us to combine image transformations like scale and rotate

with translate as one matrix multiplication instead of a longer process with vector addition. The Pin Hole Camera Model is derived from both the homogenous coordinates and the projective space. The model explains how to project a scene in the 3D space onto the image plane (2D image). From here you can do a series of matrix calculations to do achieve and derive the homography transformation matrix H.

4 Experimental Setup

All the code produced for the purpose of this project is done through with the help of online research, problem solving and code snippets. These are all referenced in the reference list below.

The following exact setup was used to create, produce and test all of the code in this research project. All of this was performed a Windows 10 Home 64-bit machine. Firstly, python needs to be installed into the machine through the installer provided on its official website. Then, Visual Studio Code for windows was downloaded as the main and only IDE to write, test and compile python code to keep consistent results through the project without any random or systematic errors. Furthermore, “pip install opencv-python” was used in the command prompt to install the basic package of OpenCV library into the machine as the full package isn’t necessary for the scope of this project. A dedicated folder was created where all the python files were created and stored all the images to be tested upon. If the results of this experiment need to be reproduced, this exact setup described can be used, along with all the references that were used and there will also be a dedicated GitHub Repository for code reference.

Basic Code Explanation for reproduction purposes:

In total, there are three different working algorithms used to produce results, which will be shown later. Firstly, Shi-Tomasi Corner Detector & Good Features to Track with randomly coloured lines and without lines. Secondly, Harris Corner Detector without sub-pixel accuracy and Harris Corner Detector with sub-pixel accuracy. Finally, the last algorithm uses homography to warp any image and produce an output with a top-down view.

Shi-Tomasi Corner Detector & Good Features to Track with/without randomly coloured lines:

Firstly, numpy and cv2 (OpenCV library) were imported. To read the table tennis table image, a variable was created and an “cv2.imread(‘ ‘)” function was assigned to it to be used with a relative defined path of where the image is located at, which was in the same folder as the python file. A “cv2.resize()” was used to resize the image to a desired size. The next step was to use “cv2.cvtColor()” to transfer the image to a gray-scale image. Then in order to detect the corners, goodFeaturesToTrack() function was assigned to a variable and utilized with the parameters explained above. The parameters will vary based on the condition of the image and how the code is structured for each individual case, just like with any parameters that will be discussed. A np.int0() function was used to turn the floating-point corners into integers. In order to draw the corners, a for loop was used with the function cv2.cirlce() define how the corners should be displayed. This is where the code for without lines ends. Furthermore, for the code with lines, a nested for loop was utilized to iterate over the corners to draw a line from each corner to the other with the lines being any random colour.

Harris Corner Detector with/without sub-pixel accuracy:

A similar approach method was used as the Shi-Tomasi Corner Detector above, where numpy and cv2 were imported. The image was read in, converted into gray-scale image. Then the "cv.cornerHarris()" function was used to detect the corners with its built-in algorithm. Then from here, for the code with sub-pixel accuracy, centroids were calculated, which are basically the mean value of each of the input variables in order to calculate the sub-pixels. Then the "cv.cornerSubPix()" function was utilized as centroids being one of the parameters which were converted into floating point values for high precision sub-pixel values that it provides.

Homography:

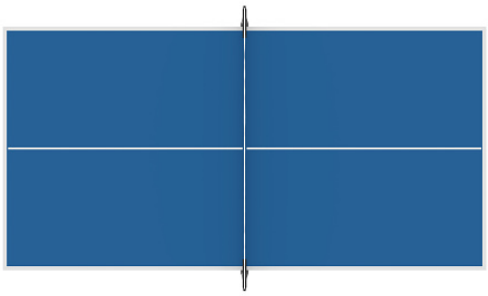
The approach of this method was unique compared to the previous two. This method does not use any built-in algorithm to detect corners. Rather the user has to manually pick the four corners of any table tennis table image and it will produce a top-down view of the table. The code for this method was that Cv2 and numpy were imported. A function was created which created a circle whenever left mouse button was clicked on the image. Then the input image was read and a size configuration was given to the output image. A list of points were created to store and append the four coordinates of the image every time the left mouse button was clicked using "np.append()". Finally, "cv2.findHomography()" and "cv2.warpPerspective" functions were used in order to find the homography transformation matrix based on the in-built algorithm of the function as explained earlier and then the warp perspective produced a top-down view of it.

5 Experimental Setup

All of the three algorithms discussed above in the experimental setup were tested for results. Below are the results found for each of the algorithms.

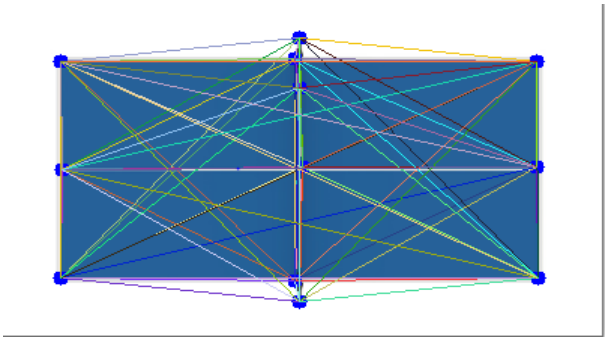
Firstly, the below figure is the image that was used to test the first two algorithms for corner detection, including Shi-Tomasi Corner Detector & Good Features to Track with/without coloured lines and Harris corner detector with/without sub-pixel accuracy.

Figure 7: Test image for Shi-Tomasi Corner Detector & Good Features to Track with/without coloured lines and Harris corner detector with/without sub-pixel accuracy



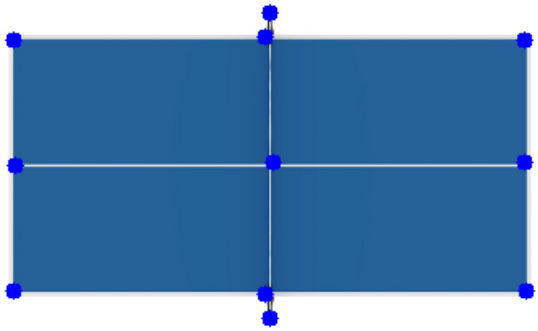
By applying this image onto Shi-Tomasi Corner Detector & Good Features to Track with randomly coloured lines, the following output image was produced:

Figure 8 : Output image for Shi-Tomasi Corner Detector & Good Features to Track with randomly coloured lines.



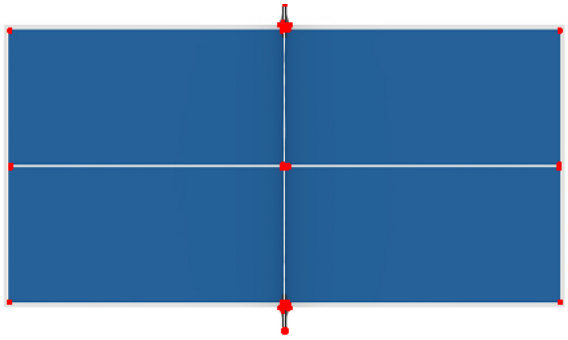
By applying the same algorithm without the randomly coloured lines, the following output was discovered:

Figure 9: Output image for Shi-Tomasi Corner Detector & Good Features to Track without randomly coloured lines.



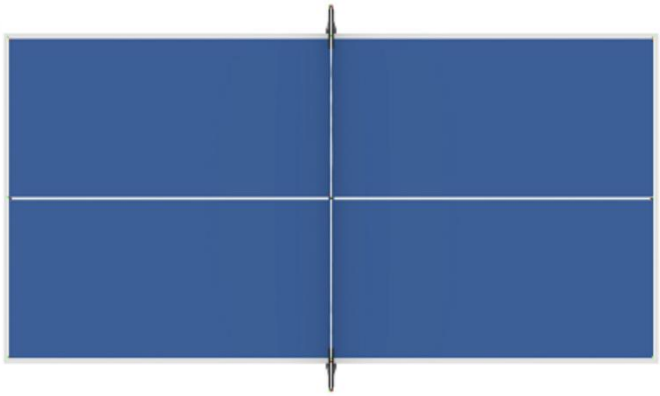
As seen by the two results above, the Shi-Tomasi Corner Detector along with Good Features to Track were able to detect the corners of the table, however, they also detected other points which seem to be intersection points.
Now moving onto Harris Corner Detector without sub-pixel accuracy, the following result was outputted from the designed algorithm:

Figure 9 : Harris Corner Detector without sub-pixel accuracy



And the output from Harris Corner Detector with sub-pixel accuracy:

Figure 10: Harris Corner Detector with sub-pixel accuracy



The output drawings of the corners above are very hard to see as the corners are drawn with red and green colour on sub-pixel with high accuracy, however, they produced the exact same results as the normal Harris Corner Detector, only with a higher accuracy and precision of the pixels.

As seen from the results above from Harris Corner Detector with and without sub-pixel accuracy, the results are almost identical to Shi-Tomasi and Good Features to Track algorithm. Both were unable to detect "only the corners", and they also ended up detecting other points which seemed like corners, such as the middle intersection point of the table. Even if both algorithms were limited to display 4 corners, they did not displayer the outer four edge corners which is what we wanted, as the algorithms aren't designed to only detect the "outer" edge corners. They're designed to detect anything that seems like a corner. Hence, it was decided to produce an algorithm where we could avoid algorithmic errors and interpretations and manually pick the corners to produce a top-down view of the table from any given image. Therefore, the homography algorithm was developed and here are the results:

Figure 11: On the left: Four yellow corners manually picked, On the right: Output of a top-down view from the four corners picked.

As seen by the result of the homography algorithm, you can manually pick the four corners of a table tennis image and you can produce a top-down view of the image just like show in the figure above. This can be applied to mostly any table tennis table image and it should produce that outcome.

5 Project scope, issues and limitations:

The scope of this project was relative, it was based on many factors. The main factor was with what knowledge I started to work on this project. With little to no knowledge about python, the goal was to achieve much as possible on a person level by learning OpenCV and python. There were also few issues and limitations of the outcome of this project. Firstly, the main factor that influenced the outcome of the results is that there was no training data to work with. As seen by a literature review above, synthetic or real data that can be training with produces much more consistent and reliable results. Another limitation was that the level of specific information related to finding "outer corners" of a table was limited online. Therefore, a lot more research and problem solving had to be done.

6 Conclusion and Future Work:

In conclusion, the results from Shi-Tomasi Corner Detector and Harris Corner were satisfactory, as they were able to detect the outer corners, however, they also included other corners as that's what the algorithm was designed to do. The results from the homography and warp perspective algorithm were great as any image could be put through that algorithm to produce such results. Although, a major goal for future work on this project is to create an algorithm much like the homography one, such that it is able to detect the four corners of an image automatically without the need of manual human control as that is not feasible in a long-term environment for productivity and external use.

ACKNOWLEDGMENTS

This work was supported and monitored by Mahdi Kazemi Moghaddam.

REFERENCES

[1] - Brownlee, J., 2021. A Gentle Introduction to Object Recognition With Deep Learning. [online] Machine Learning Mastery. Available at: <https://machinelearningmastery.com/object-recognition-with-deep-learning/> [Accessed 1 April 2022].

[2] - Ji, Y., Zhang, H., Zhang, Z. and Liu, M., 2021. CNN-based encoder-decoder networks for salient object detection: A comprehensive review and recent advances. Information Sciences, 546, pp.835-857.

[3] - Dickson, B., 2021. An introduction to object detection with deep learning. [online] TechTalks. Available at: <https://bdtechtalks.com/2021/06/21/object-detection-deep-learning/> [Accessed 1 April 2022].

[4] - Lowe, D., 2004. Distinctive Image Features from Scale-Invariant Keypoints. International Journal of Computer Vision, 60(2), pp.91-110.

[5] - Dalal, N. and Triggs, B., 2005. [online] Vision.stanford.edu. Available at: <http://vision.stanford.edu/teaching/cs231b_spring1213/papers/CVPR05_DalalTriggs.pdf> [Accessed 2 April 2022].

[6] - Virasova, A., Klimov, D., Khromov, O., Gubaidullin, I. and Oreshko, V., 2021. Rich feature hierarchies for accurate object detection and semantic segmentation. Radioengineering, pp.115-126.

[7] - Virasova, A., Klimov, D., Khromov, O., Gubaidullin, I. and Oreshko, V., 2021. Rich feature hierarchies for accurate object detection and semantic segmentation. Radioengineering, pp.115-126.

[8] - Herur, S. and Kerur, S., 2018. International Journal of Pure and Applied Mathematics, [online] (1314-3395). Available at: <efaidnbmnnnibpcajpcglclefindmkaj/https://acadpubl.eu/hub/2018-120-6/1/53.pdf> [Accessed 11 June 2022].

[9] - Wu, H. and Wan, Y., 2021. A highly accurate and robust deep checkerboard corner detector. Electronics Letters, 57(8), pp.317-320.