



# **Fake Land Deed Detection**

Team Members:

Palesha Basumatary - 20BCE1229

Nilashma Saha - 20BCE1514

Bacham Sai Venkata Teja - 20BCE1551

Review 3

CSE3002 - Internet Web Programming

Slot: A2

Professor: Maheswari S

## **Acknowledgement**

We sincerely thank Dr Maheswari S of VIT University, for allowing us to do this project.

## **Declaration**

We hereby declare that the report entitled “**FAKE LAND DETECTION SYSTEM**” submitted by us, for CSE3002 - Internet Web Programming to VIT is a record of bonafidework carried out by us under the supervision of Dr Maheswari.

We further declare that the work reported in this report has not been submitted and will not be submitted, either in part or in full, for any other courses in this institute or any other institute or university.

PLACE: CHENNAI

DATE: 14<sup>th</sup> November 2022.

**SIGN. OF FACULTY**

(Professor: Maheswari S)

## **Abstract**

The occurrence of counterfeit land documents can be detrimental to the effective running of the monetary systems of any nation. These counterfeit documents can be distributed by local goons and other criminal organizations and be used to fund illegal activities. The forgery of official Land deeds becomes familiar which causes a lot of problems and difficulties for official institutions. This paper examines methods to enhance the security features of Indian Land Deeds. The introduction of the INR 100 and 500 document notes in November 1989 resulted in some discourse concerning the security features of these new Indian Land Deeds. The wide occurrence of counterfeit Indian Land Deeds in the denominations of INR 10, 20, 50, 100, and 500 have been reported over recent years. There is an expectation that these newly introduced INR 100 and 500 may also be counterfeited and distributed illegally by criminal gangs and terrorist groups. This paper proposes a novel Deed recognition system where the counterfeit document is automatically recognized without any human intervention. The proposed system provides an interface to recognize the Indian Land Deeds and authenticate their validity. The system also identifies counterfeit documents by using a scanner and various image-processing methods for shape recognition. Various image features were used to distinguish between counterfeit and noncounterfeit Deeds. Also, manually verifying the authenticity of these documents physically increases administrative overhead and takes a lot of time. The study used an image database of Indian Deeds to access the accuracy of the proposed system. The experimental results show that the accuracy of the system proposed is close to 90% with a satisfactory level of sample processing. The accuracy of this identification was found to be diminished for sample Deeds which were damaged.

**Keywords:** ITSegmenter, Unsharp Masking, FAST Algorithm, K-D Tree, DBSCAN, Gaussian Blur, OCR, Cluster Determination, Authenticity Verification.

## **Motivation:**

In the modern world, documents can now be altered and manipulated easily. Hence, the trustworthiness of documents has never been more in demand. Many people resort to these techniques to reach the end of their means.

Thus, to counter the danger posed by fake legal documents, we need to come up with new improved methods to minimize the threat as much as possible. Many preventive measures have been taken by the government to curb these forgery activities and nip them in the bud. But unfortunately, it still has not affected the rapidly growing rate of these crimes.

On the other hand, the usage of fake land deeds on the internet is also increasing rapidly. About 5,000 people in the United States are victims of 36 fake Land deeds every year, with a negative financial crisis of approximately \$50 million. Detecting the fake scanned Land deed is a challenging task. Because, a deed issued by the government or private sector will have several pieces of information like customer images, seals, holograms, texts, stamps, etc. Validating all the security features in a scanned document is important because sometimes a small part of the document might have been forged and all together might show genuineness. For example, in any deed, (a) holograms and seal are genuine, but the image (four lions) present in the document is faked (b) the image is genuine, but the seal and holograms are forged (c) image, seal, hologram are genuine, but textual contents are forged (d) image, seal, hologram, textual contents are genuine, but the signature is forged.

The use of optical recognition systems for the identification of counterfeit Land deeds is one possible technique to assist government agencies and organisations to remove these deeds from records in registration offices. This study proposes a system which uses image

processing techniques and various optical features of Indian Land deeds to detect analogies in these deeds. This paper will evaluate the accuracy of this system and outlines the implications of using this system in reducing the circulation of fake Land deeds in India.

In this research work, we have created a Cluster network-based automatic document verification system to verify all the security features of the Land deed copy. The proposed system allows users/customers/employees to upload the digital land deed directly to the verification system. Later, the FAST algorithm is programmed to check for discrepancies and discrepancies in the document. The verification system then takes over the outcomes to DBSCAN to perform further validations for each object found from the deed uploaded by the user automatically and extracts the important information and verifies the authenticity of the document.

The main motive behind this project is to come up with a concept and develop an idea around it, aimed at social and consumer welfare.

We thus, wish to build a Fake Land Deed Detection System to validate the authenticity and transparency of Recorded Documents regarding lands issued under the jurisdiction of the Indian Government.

## **Literature Review Summary**

**Paper 1: Dr. P. Mangayarkarasi, Akhilendu, Anakha A S, Meghashree K and Faris A B**

As money transactions in India are increasing day by day, verification and recognition of currency are much needed. Since normal citizens are the ones who suffer the most from fake currency, it is important to have a simple and efficient method to identify fake banknotes. In this paper, the Brute Force Classification algorithm is used. Methods mentioned in this paper are image segmentation, edge information of image and characteristics, and feature extraction. The method

proposed is divided into two parts. The first module is Currency Recognition which includes image acquisition, Pre-processing concepts, Edge Detection, Image Segmentation, Feature Extraction and Comparison. Currency Verification is the second module, in which the feature is extracted and verified with the existing features, to produce an output. This method is a cost and time-efficient technique.

**Paper 2: G. Chandra Praba, E. Jeevitha, A. Abitha, A. Shalini and B. Swetha**

Forgery of any official documents is a serious crime and it is punishable by fine or imprisonment, or both. Also, it is difficult for official institutions to identify forged documents as they can easily be edited. There are two concepts used in this paper:-image processing techniques using neural networks and another is scanning the QR code of the document which detects a forgery if not found.

In the case of a QR code scanner, the code contains an encrypted code that determines if the document is fake.

Another method is image processing using neural networks where the image is converted to a grey scale and normalized and the removal of noise is done. The image is then segmented into 3 parts logo, stamp, and signature and a feature extraction method is applied.

**Paper 3: Hiba Benhamza, Abdelhamid Djeffal, Abbas Cheddad**

Due to the increasing advancement of technology, It is easier to forge document images. This paper aims to compare the proposed methods for image forgery detection and discuss their pros and cons. Image forgery detection is divided into two categories: active and passive methods.

Active methods compare the digital signature and watermark with previous information taken from the image. Only pre-processed images can be identified in active methods.

In the case of passive methods, it can detect copy-move forgery and image slicing. It analyses the statistical distortion that is left behind after forgery. Copy move forgery detection starts with pre-processing steps and is converted to grayscale. Then feature extraction is used to collect information from the image and then each block is compared to

determine the manipulated area. CMFD is divided into two types: - Block-based method and Keypoint-based method.

**Paper 4: Kaushik, Manish Shankar, Rabul Saikia, and Aditya Bihar Kandali.**

The editing tools like photoshop, beautify app, etc. have widely increased making it easier to manipulate images. Many proposed different methods. Farid and Popescu proposed a statistical method to determine the block of image resampling and to use discrete cosine transformation. While few performed convolutional neural networks that showed excellent results in detecting various image manipulation. But these techniques are restricted to only some file formats and to detecting manipulation in an uncompressed manner.

The proposed methodology for this paper is using Local Binary Pattern (LBP), features are acquired efficiently, and combine the texture in a feature vector. After feature extraction, 5 steps of a Histogram of Oriented gradient are applied to depict the object's shape and appearance. A support vector machine is applied to classify authentic images and tampered images.

**Paper 5: Shang, Shize, Xiangwei Kong, and Xingang You**

Document authentication is a vital step in this technologically advanced world. Earlier, the detection was done by trained workers, chemicals, and other such systems which were expensive and inefficient. This paper describes a method to analyze and detect tampered documents produced by laser printers, inkjet printers, and electrostatic copiers. Characteristics of documents each produced by different devices are analyzed in this paper. First, the scanned document's image is pre-processed, and based on the analysis, the feature of noise energy, average gradient, and contour roughness are computed. Contour roughness is drawn from character and reflects the degree of roughness. The average gradient is used to differentiate inkjet printers from laser printers. It classifies each letter that can be used to detect a forged document.

## Literature review summary table

Authors and year of references	Title (Study) & no. of pages	Concept/Theoretical model/Framework	Methodology used/ Implementation	Data set Details/Analysis	Relevant Finding	Limitation / Future Research/ Gaps identified
<b>Dr. P. Mangayarkarasi, Akhilendu, Anakha A S, Meghashree K and Faris A B</b>  (May 2020)	Fake Indian Currency Note Recognition  [5 pages]	The paper proposes a system that can verify Currency notes using an easy processing algorithm.	1. Image Acquisition 2. Image Processing 3. Image segmentation 4. Feature extraction	For this project, the dataset is taken from the user.	In this system, the analysis of the image is accurate and the method is cost and time effective.	There is a need for relative comparisons between these techniques over the same data sets.
<b>G. Chandra Praba, E. Jeevitha, A. Abitha, A. Shalini and B. Swetha</b>  (2021)	Fake Education Document Detection using Image Processing and Deep Learning  [3 pages]	This paper proposes two methods to distinguish between the original and the forged one. 1. One is a QR-code scanner which scans the QR- code of the document and detects whether the document is original or forged. 2. Another is the image processing technique to detect fake documents. This	1. QR-code scanner module (checks if the document has an encrypted code) 2. Image processing techniques include - Error Level Analysis and Neural Network.	NA	We use the minimum distance classifier to classify the document. The proposed system is efficient and the results obtained are accurate.	There is a need for relative comparisons between the time taken by these two separate processes to determine the more



		system should be implemented on the web portal at the time of submission of the document.				efficient way.
<p><b><i>Hiba Benhamza, Abdelhamid Djeflal, Abbas Cheddad</i></b></p> <p><b><i>(2021)</i></b></p>	<p>Image Forgery Detection</p> <p><b>[8 pages]</b></p>	<p>In this paper, the authors section image forgery detection into two main types.</p> <ul style="list-style-type: none"> <li>- The first one is an active method that depends on the previous information collected from the original image. It focuses on watermark and signature comparison.</li> <li>- Another is a passive method used to detect forgery without previous information.</li> </ul>	<p>Passive methods are:</p> <ol style="list-style-type: none"> <li>1. Copy-move tampering</li> <li>2. Image slicing.</li> </ol> <p>(Various other main existent works are described.)</p> <p>Active methods are:</p> <ol style="list-style-type: none"> <li>1. Steganography, a technique modified from an existing technique using wavelet transform and many existing works.</li> </ol>	Various Datasets	Various image forgery techniques and a sum up of their evaluation metrics and results.	One limitation of active methods is that only pre-processed images can be identified.
<p><b><u>Kaushik, Manish Shankar, Rabul Saikia, and Aditya Bihar Kandali.</u></b></p> <p><b><i>(2019)</i></b></p>	<p>Digital Image Forgery Detection using Local Binary Patterns (LBP) and Histogram of Oriented Gradients (HOG)</p>	<p>This paper proposed a fusion method of Local Binary Pattern and Histogram of Oriented Gradient to detect authentic and tampered images.</p>	<p>Using a Linear Binary pattern, extract the succeeding function block and then apply a Histogram of Oriented Gradients</p>	CASIA 1 and CASIA 2	Compare this methodology with other methods like SVM + extreme learning (ELM), CNN+ Markov	The future objective is to improve the accuracy rate for high efficiency

	[4 pages]		on the extracted feature. SVM classifier is used for training and classification.		characteristics, etc.	
<u>Shang, Shize, Xiangwei Kong, and Xingang You.</u>  (2015)	Document forgery detection using distortion mutation of geometric parameters in characters.  [13 pages]	This paper describes a method to distinguish tampered documents produced by laser printers, inkjet printers, and electrostatic copies.	Characteristics of documents each produced by different devices are analyzed in this paper. First, the scanned document's image is pre-processed, and based on the analysis, the feature of noise energy, average gradient, and contour roughness are computed. It will check individual characters to classify which can be used to classify the forged documents	NA	Classification accuracy for all the devices. The effects of JPEG compression it has on accuracy. Forgery detection.	Improving the robustness of the proposed method and improving its accuracy at lower resolution.

## Proposed Methodology

Our project verifies land deeds priced at ₹100 & 500. The verification is done in either of the 2 methods. But, for both of them, the first clustering points are determined in the uploaded document. A maximum of 16 clustering points are identified while the minimum number of clustering points to be identified is 4. The 4 major clustering points to be identified are – the four lions, the quotation – “Indian Non-Judicial”, the rupee symbol, and the amount specifying text.

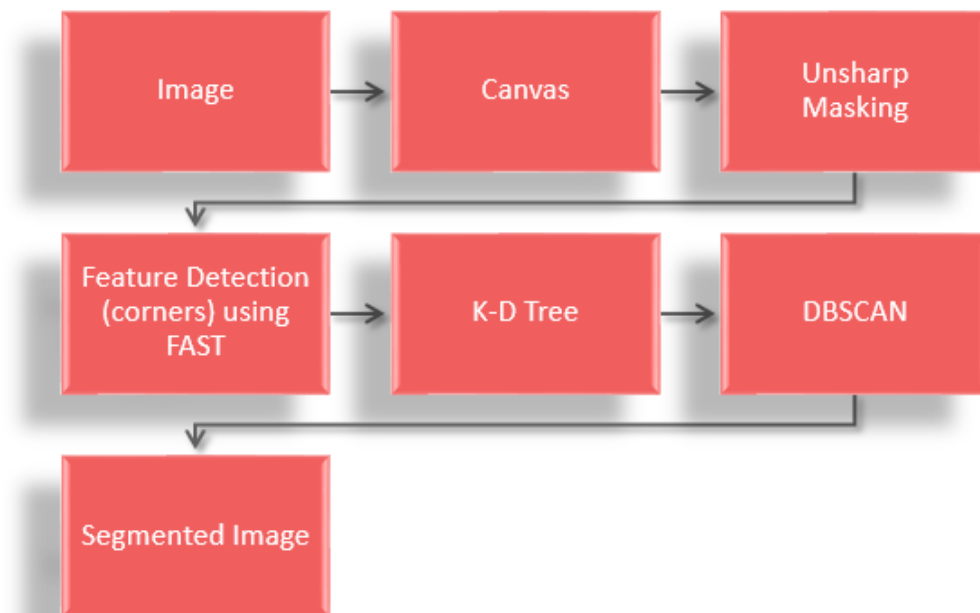
In method 1 the distance between two ascertained clustering points is determined and then compared to the distance recorded for the original document. If matched i.e., the verification is a success for all such pairs of clustering points then the document is authenticated else is proven to be fake.

In the 2<sup>nd</sup> method the objects that are identified, aka the clustering points are examined for their contents as mentioned before such as the four lions’ emblem, the money amount etc. If the contents are the same as that found in the original document, then the document is verified and authenticated, else it’s proven to be a counterfeit.

The proposed system implements a Pattern Recognition Technique for matching the pattern of imprint obtained from the implementation of the Image Processing Technique. Thus, we use:

1. Image Acquisition
2. Edge Detection
3. Image Segmentation
4. Feature Extraction
5. Comparison
6. Output

The above-mentioned techniques will take place as described in Fig 1.



*Fig. 1. Figure showing the sequential flow of methods and functions used.*

### ***Step 1:***

Image acquisition is the creation of digital images, typically from a physical scene. The image considered here is an Indian currency note and is generally acquired by using a digital camera. The image is then stored for further processing. Depending on the field of work, a major factor involved in image acquisition in image processing sometimes is the initial setup and long-term maintenance of the hardware used to capture the images. One of the forms of image acquisition in image processing is known as real-time image acquisition. This usually involves retrieving images from a source that is automatically capturing images.

## Step 2:

Features from Accelerated Segment Test (FAST) is an efficient edge detection algorithm used to extract feature points and map them to computer vision tasks. It has a high computational efficiency compared to other image feature detection algorithms. In this proposed method, the corners of the image such as the seals, hologram, stamps and signature are detected by drawing a Bresenham circle of radius 3 for every pixel. It also uses 12point segment detection on the Bresenham circle to detect the exact corners of the image. The algorithm of the FAST corner detection algorithm is explained in Algorithm 1 and Fig 2. shows the FAST corner detection of the hologram image in the passport.

---

**Algorithm 1:** FAST corner detection Algorithm

---

**Step: 1:** Select a pixel ' $p$ ' from the document

**Step: 2:** identify the interest point (intensifying pixel) and fix threshold value ' $t$ '

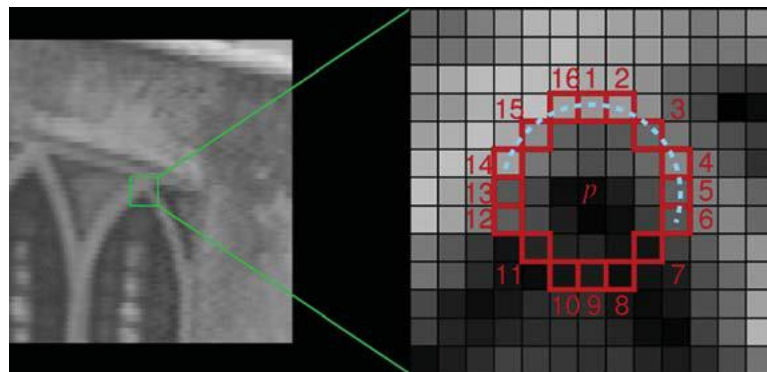
**Step: 3:** Take Bresenham circle of radius 3 around the intensifying pixel.

**Step: 4:** Pixel ' $p$ ' is identified as corner, if it has ' $n$ ' brighter neighboring pixel than threshold from the circle

**Step 5:** If four or more pixels values are stays above the threshold value, then start drawing a boundary lines.

**Step 6:** Continue the process in all the pixels.

---



*Fig. 2. Corner detection of the images in the scanned document using the FAST algorithm.*

ITSegmenter.js uses the implementation of FAST from tracking.js.

```
var corners = Fast.findCorners(pixels, width, height);
```

### ***Step 3:***

We will be using ITSegmenter to segment the image into various objects. ITSegmenter.js aims to detect objects and isolate text regions in an image.

“text segment” is the main function to call, it has 1 required parameter and 9 optional parameters. The simplest usage would be to just specify the image path and use the default value for the optional parameters.

```
textSegment(imgPath);
```

Using default optional parameters the original image remains unaffected, and the coordinates of the bounding rectangle for an object area will be stored in the output and reacts asynchronously.

```
textSegment(imgPath, fThreshold, eps, minPts, dia, amt, drawRects, splitRects, convertToImage, canvasId);
```

**fThreshold:** Fast Threshold; Default:100, higher = less corners

**eps:** Maximum distance between two points to be considered neighbours; Default:15

**minPts:** Minimum number of points required to form a cluster; Default:5

**amt:** Scalar of Unsharp Mask

**drawRects:** Option to draw bounding boxes on the image; Default:0

**splitRects:** Option to split the text segments into individual images; Default:0

**convertToImage:** Option to convert canvas to image

**canvasId:** Option to segment the image on a specific canvas

#### ***Step 4:***

Unsharp masking followed by Gaussian blur is used.

***Unsharp Masking*** is an image sharpening technique, which takes in an image and blurs it, inverts the blurred image and adds it back to the original image. Blurring an image reduces the high-frequency components, thus the inverted blurred image contains only the high-frequency component. Adding that back to the original image amplifies the high-frequency components of the image. This is desirable because the text on an image is generally high-frequency.

To apply Unsharp Masking to an image, call the sharpen function with the parameters being: canvas context of the image, width of the canvas, the height of the canvas, the diameter of the blur, and scalar for the Unsharp Masking effect.

```
sharpen(ctx, w, h, dia, amt);
```

***Gaussian Blur*** is a convolution operation with the Gaussian kernel.

It is very computationally expensive to convolve with a Gaussian kernel. Thus an approximate Gaussian blur by passing the image through 3 box blur is used. It requires 3 parameters: the source buffer, width, height, and diameter of a blur.

```
var blurred = gaussBlur(srcBuff, w, h, dia);
```

First, the ideal kernel size of each box blur is calculated, and then the image buffer is passed through 3 box blur. The sigma parameter corresponds to the diameter of the blur and n is the number of passes.

```
boxesForGauss(sigma, n);
```

Box blur is a simple averaging of surrounding k-cells for each pixel, where k is the kernel size. Furthermore, box blur is separable; convolving the image horizontally and then vertically yields the same result as convolving with the entire kernel.

```
boxBlur(srcBuff, w, h, kernelWidth);
```

### ***Step 5:***

For feature extraction, we need pixel data of each object found from ITSegmenter. To access the pixel data, the image is drawn on a canvas element, the image data can then be accessed by the getImageData method. From there, Unsharp Masking is applied to the image data to sharpen it. This improves the results of the next step, feature detection. After sharpening the image, Features from the Accelerated Segment Test (FAST) are used to find the corners in the image.

In this step, we have used K-D Tree, Range Query & DBSCAN.

## ***K-D Tree***

K-D tree is a binary space partitioning tree, where points are partitioned into 2 parts at each node based on alternating dimensions. It is a very efficient data structure for nearest neighbour and range search problems. This is desirable because the time complexity of DBSCAN is governed by the number of range searches it invokes.

### **Construction:**

In this implementation, the K-D tree is constructed by recursively populating each node's children. The position of the node for



partitioning is selected by using Quick Select. By selecting the median as the position of the node for partitioning, this method results in a balanced tree, which is essential for fast querying on the tree. To create a K-D tree, initialize a new instance of `kdTree` with a points array storing the coordinates. Where the points array is in this format: `[[x1, y1], [x2, y2], ...]`.

```
var tree = new kdTree(points);
    tree.nodes;
    tree.rootNode;
```

Once the tree is created the structure can be accessed via the `tree.nodes` or `tree.rootNode`. ‘`tree.nodes`’ is an array that stores all the individual nodes of the tree and `tree.rootNode` stores the root node of the tree which contains all other nodes in its descendants. The node class contains 4 properties: `position`, `leftChild`, `rightChild`, and `depth`. The `position` is the coordinate of the node, `leftChild` and `rightChild` are the children node of this node, and `depth` is how deep the node is on the tree. The descendants of a node can be accessed via the `getDescendants` method. It takes in 2 parameters, the node to be traversed and the descendants array to store the output.

```
var descendants = [];
getDescendants(node, descendants);
```

## *Range Query*

The range query implemented here is circular, it is very similar to the K-NN search and also orthogonal range query. The circular range query returns all points that are within radius `r` from point `[x,y]`.

```
var Neighbours = tree.rangeSearch(x, y, r);
```

`rangeSearch` traverse down the tree recursively visiting nodes with the

region that either intersects the query circle or is fully contained in the query circle. If the node's region intersects with the query circle, the square distance between the node. position and the query point is compared to  $rSquare$ . If it is less than  $rSquare$  the point is within the query region. If a node's region is completely contained in the query circle then the node and all its descendants are within the query region.

## DBSCAN

DBSCAN (Density-based Spatial Clustering of Applications with Noise) is a data clustering algorithm. DBSCAN is very versatile, it can find clusters of arbitrary shape and even internal clusters. It is used for clustering groups of objects together to draw bounding boxes around them or crop them out. It takes in an array of points and the parameter  $eps$  and  $minPts$  and output clusters of points.

```
var clusters = DBSCAN(arr, eps, minPts);
```

Points form a cluster if they are close together. Every point in the input array is visited, and its neighbouring points are found using RangeQuery.  $Pt$  is the query point,  $eps$  is the maximum distance between two points to be considered neighbours and  $index$  is the K-D tree structure.

```
var Neighbours = RangeQuery(Pt, eps, index);
```

RangeQuery calls the rangeSearch method from the kdTree class and returns all neighbouring points within radius r of point Pt.

## Step 6:

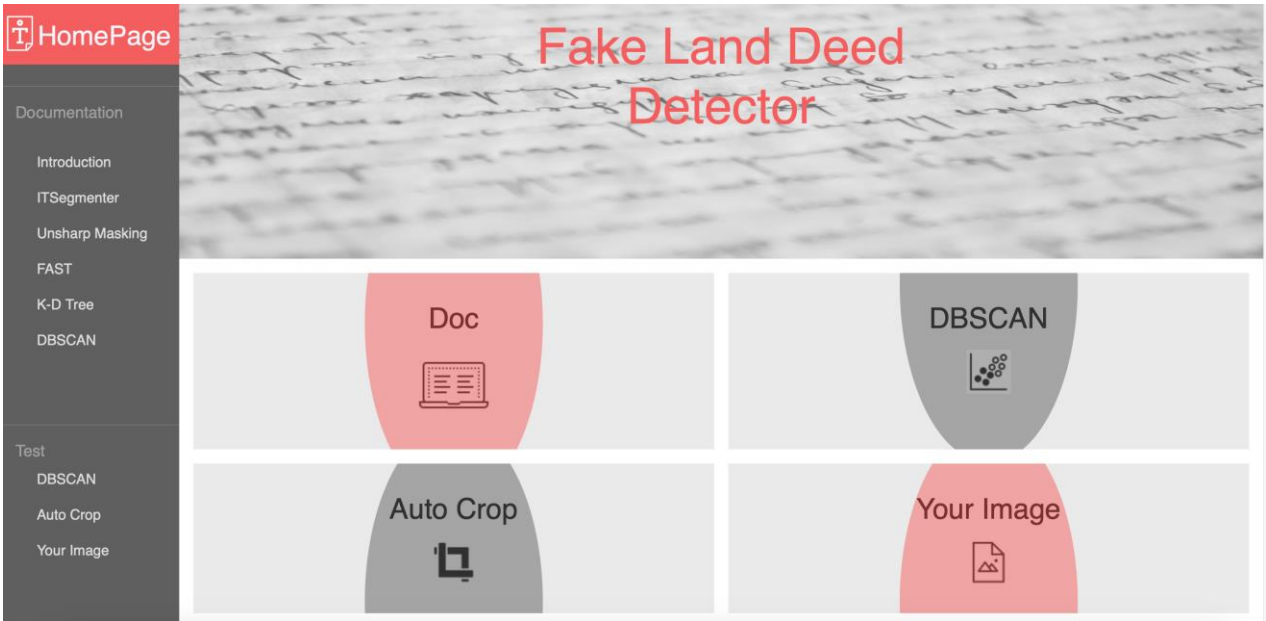
When texts exist in an image alongside other objects, OCR engines often return random characters and carriage returns. To combat this problem ITSegmenter.js detects and isolates the text regions in an image.

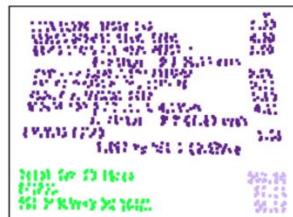
From the above-mentioned methods in step 5 firstly the coordinates of the corners are stored in an array and used to construct a k-d tree. From there, the Density-based Spatial Clustering of Applications with Noise (DBSCAN) algorithm is applied to corners, which groups them into clusters. Once the clusters are formed, bounding boxes can be drawn based on the maximum and minimum x/y values of each cluster.

These clustering distances will be mapped with the values which are stored in a database. Every cluster distance calculated will be checked whether it matches the original land deed document records.

Accordingly, to the results we get, the conclusion will be made stating whether the land deed is fake or original.

# Developed Website:





Parameters

Submit

HomePage

Documentation

Introduction

ITSegmenter

Unsharp Masking

FAST

K-D Tree

DBSCAN

Test

DBSCAN

Auto Crop

Your Image

Your Image

This demo allows you to upload your own image and experiment with the parameters.

Parameters

Choose Image

Browse

☐ Auto Crop

Submit

HomePage

Documentation


- Introduction
- ITSegmenter
- Unsharp Masking
- FAST
- K-D Tree
- DBSCAN

Test

- DBSCAN
- Auto Crop
- Your Image

Your Image

This demo allows you to upload your own image and experiment with the parameters.



498x244

Default parameters are tuned for images with 1000px by 1000px

Adjust slider to change image resolution

1

HomePage

Documentation


- Introduction
- ITSegmenter
- Unsharp Masking
- FAST
- K-D Tree
- DBSCAN


Test

- DBSCAN
- Auto Crop
- Your Image

Your Image

This demo allows you to upload your own image and experiment with the parameters.





**Congrats!**  
Your document is original!

OK

Default parameters are tuned for images with 1000px by 1000px

Adjust slider to change image resolution

1

demo1.jpg

Browse

☒ Auto Crop

Submit

correct

रिती

अ

यिनी

एक सी रु

मये

रु.

अ

भारत INDIA  
INDIA NON JUDIC

रु.

रु.

Your Image

This demo allows you to upload your own image and experiment with the parameters.



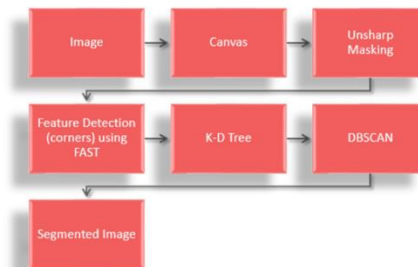
Oops!

Your document is not original!

OK

## Introduction

The performance of popular Optical Character Recognition (OCR) engines such as Tesseract can be greatly improved by pre-processing of the image. When texts exist in an image alongside with other objects, OCR engines often return random characters and carriage returns. To combat this problem ITSegmenter.js detect and isolate text regions in an image.



To access the pixel data, the image is drawn on a canvas element, the image data can then be accessed by the `getImageData` method. From there an Unsharp Masking is applied to the image data to sharpen it. This improves the results of the next step, feature detection. After sharpening the image, Features from Accelerated Segment Test (FAST) is used to find the corners in the image. Then the corners coordinates stored in an array are used to construct a k-d tree. From there, the Density-based Spatial Clustering of Applications with Noise (DBSCAN) algorithm is applied to corners, which groups them into clusters. Once the clusters are formed, bounding boxes can be drawn based on the maximum and minimum x/y values of each cluster.

## ITSegmenter.js

ITSegmenter.js aims to detect and isolate text regions in an image.

`textSegment` is the main function to call, it has 1 required parameter and 9 optional parameters.

The simplest usage would be to just specify the image path and use the default value for the optional parameters.

```
textSegment(imgPath);
```

Using default optional parameters the original image remains unaffected, and the coordinates of the bounding rectangle for text area will be stored in `outputRects` asynchronously.

```
textSegment(imgPath, fThreshold, eps, minPts, dia, amt, drawRects, splitRects, convertToImage, canvasId);
```

**fThreshold:** Fast Threshold; Default:100, higher = less corners

**eps:** Maximum distance between two points to be considered neighbours; Default:15

**minPts:** Minimum number of points required to form a cluster; Default:5

**amt:** Scalar of Unsharp Mask

**drawRects:** Option to draw bounding boxes on the image; Default:0

**splitRects:** Option to split the text segments into individual images; Default:0

**convertToImage:** Option to convert canvas to image

**canvasId:** Option to segment the image on a specific canvas



## Unsharp Masking

Unsharp Masking is an image sharpening technique, which takes in an image and blurs it, invert the blurred image and add it back to the original image. Blurring an image reduces the high frequency components, thus the inverted blurred image contain only the high frequency component.

Adding that back to the original image amplifies the high frequency components of the image.

This is desirable because text on an image are generally high frequency.

To apply Unsharp Masking to an image, call the sharpen function with the parameters being: canvas context of the image, width of the canvas, height of the canvas, diameter of the blur, and scalar for the Unsharp Masking effect.

```
sharpen(ctx, w, h, dia, amt);
```

### Gaussian Blur

The key process here is the Gaussian blur, which is a convolution operation with the Gaussian kernel.

It is very computationally expensive to convolve with a Gaussian kernel.

Thus an approximate Gaussian blur by passing image through 3 box blur is used.

It requires 3 parameters: the source buffer, width, height, and diameter of blur.

```
var blurred = gaussBlur(srcBuff, w, h, dia);
```

First the ideal kernel size of each box blur is calculated, then the image buffer is passed through 3 box blur.

The sigma parameter correspond to the diameter of blur and n is the number of passes.

```
boxesForGauss(sigma, n);
```

Box blur is a simple averaging of surrounding k-cells for each pixel, where k is the kernel size.

Furthermore box blur is separable; convolving the image horizontally then vertically yields the same result as convolving with the entire kernel.

```
boxBlur(srcBuff, w, h, kernelWidth);
```

## FAST

FAST (Features from Accelerated Segment Test) is a corner detection method.

It examines the 16 surrounding pixels of a pixel to determine whether or not that pixel is a corner.

If a set of N contiguous pixels in the 16 pixels are all brighter than the pixel plus a threshold or darker than the pixel minus a threshold then the pixel is classified as a corner.

ITSegmenter.js uses the implementation of FAST from [tracking.js](#).

```
var corners = Fast.findCorners(pixels, width, height);
```

## K-D Tree

K-D tree is a binary space partitioning tree, where points are partitioned 2 parts at each node based on alternating dimensions.

It is a very efficient data structure for nearest neighbour and range search problems.

This is desirable because the time complexity of DBSCAN is governed by the number of range search it invokes.

### Construction

In this implementation, the K-D tree is constructed by recursively populating each node's children.

The position of the node for partitioning is selected by using Quick Select.

By selecting the median as the position of the node for partitioning, this method results in a balanced tree, which is essential for fast querying on the tree.

To create a K-D tree, initialize a new instance of kdTree with a points array storing the coordinates.

Where the points array is in this format: `[[x1, y1], [x2, y2], ...]`.

```
var tree = new kdTree(points);
tree.nodes;
tree.rootNode;
```

Once the tree is created the structure can be accessed via `tree.nodes` or `tree.rootNode`.  
`tree.nodes` is an array that stores all the individual nodes of the tree and `tree.rootNode` stores the root node of the tree which contains all other nodes in its descendants.  
The node class contain 4 properties: `position`, `leftChild`, `rightChild`, and `depth`.  
`Position` is the coordinate of the node, `leftChild` and `rightChild` are children node of this node, and `depth` is how deep the node is on the tree.  
The descendants of a node can be accessed via the `getDescendants` method.  
It takes in 2 parameters, the node to be traversed and the descendants array to store the output.

```
var descendants = [];
getDescendants(node, descendants);
```

## Range Query

The range query implemented here is a circular range query, it is very similar to K-NN search and also orthogonal range query.  
Circular range query returns all points that are within radius `r` from point `[x,y]`.

```
var Neighbours = tree.rangeSearch(x, y, r);
```

`rangeSearch` traverse down the tree recursively visiting nodes with region that either intersects the query circle or is fully contained in the query circle.  
If the node's region intersects with the query circle, the square distance between `node.position` and the query point is compared to `rSquare`.  
If it is less than `rSquare` the point is within the query region.  
If a node's region is completely contained in the query circle then the node and all its descendants are within the query region.

## DBSCAN

DBSCAN (Density-based Spatial Clustering of Applications with Noise) is a data clustering algorithm.  
DBSCAN is very versatile, it can find clusters of arbitrary shape and even internal clusters.  
It is used for clustering group of text together in order to draw bounding boxes around them or crop them out.  
It takes in an array of points and the parameter `eps` and `minPts` and output clusters of points.

```
var clusters = DBSCAN(arr, eps, minPts);
```

Points form a cluster if they are close together.  
Every point in the input array is visited, and its neighbouring points are found using `RangeQuery`.  
`Pt` is the query point, `eps` is the maximum distance between two points to be considered neighbours and `index` is the K-D tree structure.

```
var Neighbours = RangeQuery(Pt, eps, index);
```

`RangeQuery` calls the `rangeSearch` method from the `kdTree` class and returns all neighbouring points within radius `r` of point `Pt`.

## Conclusion

From the results obtained from our project, we can conclude that with further expansion of our database or code to include other types of official documents' verification, our project will provide us full-proof verification in minimum time with 100% precision and accuracy. Thus, making it a viable technique for future implementation on a large-scale basis as a solution to detect a forgery in official documents.

## References

- ANAKHA AMAL, S. I. D. D. I. Q. U. E., K. MEGHASHREE, and AB FARIS. "Fake Indian Currency Note Recognition." (2020).
- G. Chandra Praba, E. Jeevitha, A. Abitha, A. Shalini, B. Swetha, 2021, Fake Education Document Detection using Image Processing and Deep Learning, INTERNATIONAL JOURNAL OF ENGINEERING RESEARCH & TECHNOLOGY (IJERT) ICRADL – 2021 (Volume 09 – Issue 05),
- Benhamza, Hiba, Abdelhamid Djeflal, and Abbas Cheddad. "Image forgery detection review." In *2021 International Conference on Information Systems and Advanced Technologies (ICISAT)*, pp. 1-7. IEEE, 2021.
- Kaushik, Manish Shankar, Rabul Saikia, and Aditya Bihar Kandali. "Digital Image Forgery Detection using Local Binary Patterns (LBP) and Histogram of Oriented Gradients (HOG)." (*IRJET*) 6, no. 02 (2019): 1844-47.
- Shang, Shize, Xiangwei Kong, and Xingang You. "Document forgery detection using distortion mutation of geometric parameters in characters." *Journal of Electronic Imaging* 24, no. 2 (2015): 023008.