Abstract—Counting and locating active wells is deemed important while implementing rural development projects involving irrigation. Irrigation needs to be planned more thoroughly as it is a quite expensive thing to do. Government institutions like NBSS & LUP manually detect the wells on Google Earth Pro, which is a quite cumbersome process. This paper aims to present an overview of a project developed to introduce an element of automation to this hectic procedure and explores the open source library Scikitimage which is an image processing and image recognition library that implements the basic image manipulation and processing using the core scientific modules NumPy and SciPy.

Keywords—well counting, object detection, automation, irrigation planning, rural development, Scikit-image.

1. Introduction

In computer Science, Image Processing is a significant field of study which is related to manipulation and processing of images by applying various algorithms. Different techniques can be applied on images to smoothen and sharpen them. Using digital image processing, noise and distortion in an image can be removed effectively (Vander Walt, 2014)[1-4]. The main objective of this paper is to detect objects present in an image. Object detection is a category related to Computer Vision which helps in detecting objects in an image of a particular class. Once the image is processed, it becomes easier to detect an object in them[5-6]. Object Detection is used in many real-life areas such as vehicle number-plate recognition, OCR etc. Scikit-Image which is an opensource image processing library is used mainly in python programming language for image processing. It was developed by Stéfan van der Walt and has BSD license [7-10]. Scikit-image is a field of Scipy which is also an open source python library for machine learning used for scientific and technical computing[11-12]. It mainly uses Numpy array object for processing which includes tools like Matplotlib and pandas. In this paper, every step is elaborated i.e. Data acquisition - image collection, image pre-processing, image filtering, training data collection and detection techniques.

2. Data Collection

Collecting proper data is the very first step for image processing and object detection. To detect an object (in this case wells) in an image we need to provide appropriate high resolution data set of wells. These dataset will work as training samples for the algorithm of detection and thus will act as labeled data for the model. We have to provide with main image of farms on which the wells are supposed to be detected. So, we collect the segments of images of one city and recursively apply processing and detection algorithms to detect all the wells present in every image. Additionally, images of wells is also provided so that it can be used for maximum matching on the segmented images of particular city. Focus will be more on the farms, as more number of wells will be found there. These images of wells and city is cropped and downloaded from the SAS planet which provides high resolution remote sensing images. Images will be fetched from the folder and then processed which is discussed in the later section [13]. The process of well detection starts data collection as shown in Fig. 1.
3. Image Processing for Region of Detected Images

Once the images are collected, they are ready to be processed. Image processing deals with performing operations on a digital image as shown in Fig. 2. PIL (Python Imaging Library) is used to fetch images from file, do manipulations and save in different file formats. It offers per pixel manipulation of an image, image enhancing such as sharpening and color contrasting, masking and transparency handling and image filtering such as smoothing and blurring of an image. Images read by PIL are stored as Numpy array and thus various Numpy functions can be applied on them such as acquisition and rewriting of pixel values, slicing and concatenating can be done. Numpy is self-sufficient to manipulate images and no need to use opencv library of python.(C. Saravanan,2010).

When an image is read through Numpy array, the RGB i.e. color images becomes 3D ndarray (row (height)*column (width)*color (3)) and black and white i.e. grayscale image becomes 2D nd array (row (height)*column (width)).While converting PIL image to ndarray, the color order followed is RGB. Acquisition of pixel value becomes easy due to ndarray. The origin (0,0) is the upper left of an image with min and max functions. Now, the images are imported as numpy array pixels so we can work with color band i.e. RGB bands of an image[10].

Here, we work with one color band. Matplotlib library of python is used throughout the processing to plot different images and results.

4. Scikit-Image Template Matching.

In Scikit-Image, Template matching uses match_template function for matching occurrences of instances or objects (in this case wells) on an image. It is a technique to match sample instance in an image as shown in Fig. 3 and Fig 4. When there exists many instances in an image, then this approach becomes very effective for example - counting of trees in field or counting wells in a farm or detecting all the swimming pools in a city. Fast and normalized cross relation is used by the match_template function to find the instances of the template in an image. Since, Template matching approach may require large amount of training sample data for matching or detection, we can reduce the sample data by reducing search resolution of an image and hence perform on downsized image. Below in the given output image, we return a single match, so as to highlight the maximum value in the match_template result corresponding to wel(Pratt,2007).
5. Image Metadata

A raster image is a grid of pixels of variable intensities which can be converted to an array and other way around. But it may not be just the pixel intensities, it can have additional relevant information stored within it. Image metadata is used for cataloguing and contextualizing visual information. Additional information can be for example Geo-referencing Tags, CRS (coordinate reference system), affine transformation information, Copyrights information, DPI info, ISO number, etc. In this paper we primarily deal with Geo-referencing Tags, CRS (coordinate reference system) and affine transformation information in metadata which essentially is needed for image overlaying in Google Earth Pro. Extensible Metadata Platform (XMP), EXchangeable Image File (Exif) are widely used. Several Python 3.0 libraries like exif and metadata helps to manipulate or work with metadata associated within image. The matched instances are shown in Fig. 5.

The aim is to copy the metadata information, which is lost during conversion of image into Numpy array (conversion causes only pixel intensities to be added to array and everything else is discarded) which is applied with certain image processing techniques, into image which is converted back to spatial domain by use of Python 3.0 libraries like PIL. Final output image when overlaid on map covers the area from which it was initially extracted from.

6. Object Detection

Object Detection is a technique involving detection of an object in a stationary or moving image. In the proposed work, Wells are deemed the objects of concern, which we need to locate them in the input raster. Scikit Image library python is an Image processing library which is essentially used to achieve the purpose. Scikit Image library has a Template Matching function which takes a sample image and detects occurrence of the sample patch in the image [10]. Sample image for template mapping is given in Fig. 7.

The Template Matching adds markers to the region with matched pixels (here the filter criteria is >0.75 accurate) is shown in Fig. 7.

7. Image Overlaying on Google Earth

Overlaying the resultant image on Google Earth adds the Object detection ring markers on top layer of the Earth Map as shown in Fig. 8. Adding Earth placemark on the ring markers of the overlaid image registers the Geographical coordinates of that location.

Placemarks can be counted to calculate density of wells in a region. They can be exported as shape file or KML to use in variety of other projects [16].
8. Proposed work

In proposed work, for the data acquisition part image fragments with geo-referenced metadata are gathered in a folder. One sample image of the concerned object is also extracted. Each fragment is then individually translated to Numpy array and applied Template Matching with the sample image. Match Accuracy is set to 0.75 and resultants are stored in tuples. Only the sample image (if present) is detected by default but due to reduced match filter (>0.75) objects with 25% dissimilarity with sample image is also accepted.

Tuples are then marked on the initial fragment array with markers. It is then again converted to spatial domain. Resultant image is then affixed with metadata from the initial one. This image is then capable of overlaying on Google Earth layer. Overlaid image’s markers are marked with Earth Placemarks which record the coordinates of each Well.

For ease this process can be made to run iteratively over several fragments and resultant images are too generated simultaneously. Then the whole set of images, overlaid on map are easier to work with. The placemarks then can be exported as shape files and used on variety of other platforms like ArcMap, Google Earth Engine, etc.

9. Conclusion

This paper presents the general procedure of image processing and object detection to solve the irrigation problems. To improve the irrigation system, the wells in a region are detected through image processing techniques. Automation of counting wells and locating them not only saves time but also greatly help in designing and planning irrigation system for any particular region.

Acknowledgement

The authors would like to thank government body NBSS & LUP located in Nagpur to share their problem definition with us of automating detection of wells and to give us opportunity to work with them.

References

[1] Van der Walt, S., Schönberger, J.L., Nunez-Iglesias, J., Boulogne, F., Warner, J.D., Yager, N., Gouillart, E. and Yu, T., 2014. scikit-image: image processing in Python. PeerJ, 2, p.e453.
[2] C. Saravanan, "Color Image to Grayscale Image Conversion," 2010 Second International Conference on Computer Engineering and Applications, Bali Island, 2010, pp. 196-199, doi: 10.1109/ICCEA.2010.192.
[3] Pratt, W. K., 2007, “Digital Image Processing”, Fourth Edition, A John Wiley & Sons Inc. Publication.
[4] Irani, M., Peleg, S., 1991, Improving resolution by image Registration, Graphical Models and Image Processing, vol. 53, no. 3, pp.231-239.
[5] Parveen, N. R.S., Sathik, M.M., 2010, Gray Scale Conversion Of X-Ray Image, ICISA 2010, and Chennai, India.
[6] Kumar, T., Verma, K., 2010, Analysis of Soft computing Techniques for face Detection, Thesis, Thaper University, Patiala, India.
[7] https://scikitimage.org/docs/dev/auto_examples/features_detection/plot_template.html
[8] Fonseca, L.M.G., Namikawa, L.M. and Castejon, E.F., 2009, October. Digital image processing in remote sensing. In 2009 Tutorials of the XXII Brazilian Symposium on Computer Graphics and Image Processing (pp. 59 - 71). IEEE.
[9] https://scikit-image.org/docs/dev/stable/auto_examples/features_detection/plot_template.html
[10] degeneratestate.org/posts/2016/Oct/23/imageprocessing-with-numpy/
[11] [https://towardsdatascience.com/image-processingusing-scikit-image-cb57ce4321ed](https://towardsdatascience.com/image-processingusing-scikit-image-cb57ce4321ed)
[12] [https://scikitimage.org/docs/dev/auto_examples/features_detection/plot_template.html](https://scikitimage.org/docs/dev/auto_examples/features_detection/plot_template.html)
[13] [https://opencv-python-tutorials.readthedocs.io/en/latest/py_tutorials/py_imgproc/geometric_transformations/py_geometric_transformations.html](https://opencv-python-tutorials.readthedocs.io/en/latest/py_tutorials/py_imgproc/geometric_transformations/py_geometric_transformations.html)