Jestr

Journal of Engineering Science and Technology Review 6 (1) (2013) 68-71

JOURNAL OF
Engineering Science and Technology Review

www.jestr.org

Research Article

Cloud Detection with MATLAB

P. Shrivastava

Sagar Institute of Research & Technology Excellence (SIRTE), Bhopal

Received 7 August 2012; Accepted 19 March 2013

Abstract

The accurate detection of clouds in satellite imagery is important in research and operational applications. Cloud cover influences the distribution of solar radiation reaching the ground where it is absorbed. Resulting fluxes of sensible and latent heat are critical to the accurate characterization of boundary layer behavior and mesoscale circulations that often lead to convective development. Therefore the spatial and temporal variation in cloud cover can greatly affect regional and localized weather processes. In this paper, techniques to detect cloud via satellite images is presented.

Keywords: MatLab, Detection, Cloud

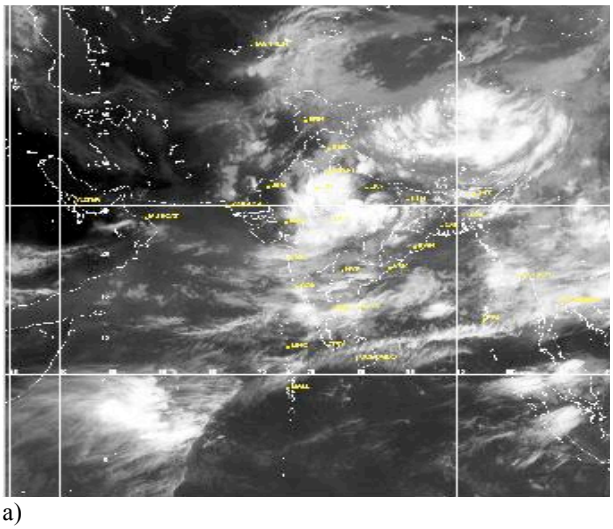
1. Introduction

Clouds are a commonly encountered phenomena that are difficult to simulate because they are complex in shape and interact with light in a complex fashion. In addition, failures in simulation are easy to detect, as everyone knows what clouds look like. They are also an important phenomena in weather, and therefore important for meteorological nowcasting, forecasting and understanding. There is a wealth of information available from the visible clouds and satellite remote sensing makes galleons of bytes data available [11]. Satellite imagery already became a common tool for both meteorologists and scientists.[6]. Clouds can be classified as:

Cloud type	Description	Elevation(feet)
Currus	Fibrous like or silky sheen	26000-35000
Cirrocumulus	Thin white patch	26000-31000
Cirrostratus	Transparent clouds that make halos of sun or moon	20000-26000
Altocumulus	Bumpy rounded masses like wool	12000-20000
Altostratus	Transparent blue/gray clouds with no halo	7000-15000
Nimbostratus	Storm cloud, dark, covers sun	0-6000
Stratocumulus	Gray or whitish layer with dark parts	5000-10000
Stratus	Low clouds with drizzle or snow, no halo	0-5000
Cumulus	Rising mounds of cauliflower white	2000-10000
cumulonimbus	Huge towers, storm clouds, hail, lightning	2000-26000

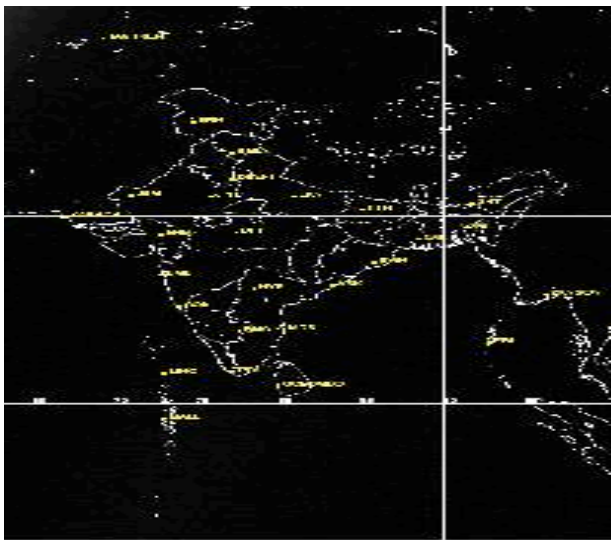
MATLAB (MATrix LABoratory) integrates computation, visualization, and programming in an easy-to-use environment where problems and solutions are expressed in familiar mathematical notation. MATLAB features a family of application-specific solutions called toolboxes. Toolboxes are comprehensive collections of MATLAB functions (M-files) that extend the MATLAB environment to solve particular classes of problems. Areas in which toolboxes are available include signal processing, control systems, neural networks, fuzzyr5r5 logic, wavelets, simulation, image processing and many others. Image processing tool box has extensive functions for many operations for image restoration, enhancement and information extraction. Here, some of the basic image processing tools are used with the images of Kalpana-1 and INSAT satellites.

For satellite imagery different channels are used as infrared channel, visible channel, water vapour channel etc.. Some of the images of those channels are shown in the figure.

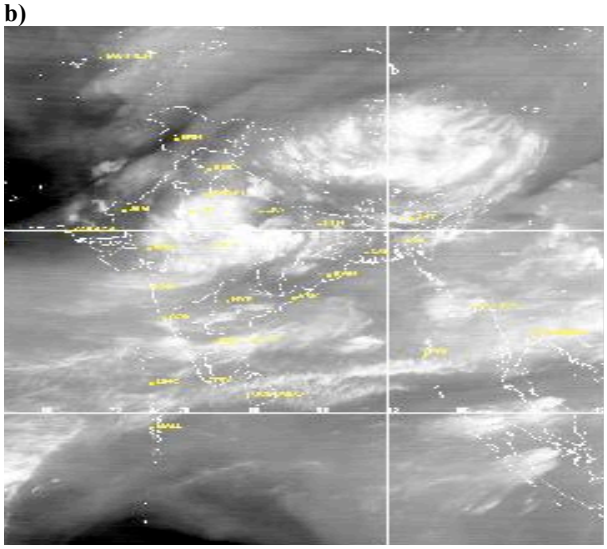


a)

* E-mail address: pavansirt@gmail.com



b)



c)

Fig.1 (a) Infrared image (b) Visible image (c) Water vapor image

2. Problems regarding Cloud Detection

Physically, we are able to discriminate between the clear and cloudy sky conditions. Such a procedure is very consolidated and robust, especially when the type of clouds present in the atmosphere matches the clouds that were simulated in the radiative models. Physical methodologies suffer from three main drawbacks: The variability of clouds in the sky is much larger than that resulting from simulations by radiative transfer models; The dependence of radiance on the emissivity of the surface, which is very difficult to estimate accurately over land; The increase of the number of spectral channels of sensors, that makes more difficult the choice of suitable bands for the decision rules.[2]

The identification of clouds in visible satellite images is relatively straight forward for a trained scientist during the day, but this process is substantially more difficult at night when only thermal channels are available.[5]. The automatically detection of clouds in GOES satellite imagery is not a simple task. Poor spatial resolution, changing solar incidence and instrument viewing angles, limited spectral channels, instrument noise and varying surface properties often limit the success of traditional cloud detection schemes.[5]

Polar nocturnal clouds are often warmer or at the same temperature as the background snow surface, complicating cloud detection. Also, there clouds tend to be thin, with lower emittances than clouds occurring during the summer.[7]. Sun glitter may interfere with cloud detection in the tropics due to spatially unresolved water bodies or recent rainfall. Over volcanic areas clouds and volcanic ash may appear similar in the visible wavelength. Over the desert, clouds and dust may appear similar. Over forests, clouds and fire may appear similar,. Terrain shadows may also interfere with cloud detection in the tropics.[8]

3. Proposed Technique for Cloud Detection

Clouds detection methods can be divided into several classes. The simplest is the so-called “radiance” threshold method where all pixels over a fixed value of illumination are considered to be cloud. [11].Clouds and radiation products can be found on the web page, http://www-pm.larc.nasa.gov. [7]. As well as, INSAT satellite images also available in many web-pages. For cloud detection, image processing toolbox of MATLAB is used.

The basic data structure in MATLAB is the array of an ordered set of real or complex elements. This object is naturally suited to the representation of images, real-valued, ordered sets of color or intensity data. MATLAB stores most images as two-dimensional arrays, in which each element of the matrix corresponds to a single pixel in the displayed image. For example, an image composed of 200 rows and 300 columns of different colored dots would be stored in MATLAB as a 200-by-300 matrix. Some images, such as RGB, require a three-dimensional array, where the first plane in the third dimension represents the red pixel intensities, the second plane represents the red and green pixel intensities, and the third plane represents the blue pixel intensities.

This convention makes working with images in MATLAB similar to working with any other type of matrix data, and renders the full power of MATLAB available for image processing applications.

To read and display the images ‘imread’ and ‘imshow’ command are used. Image is displayed as shown in Fig 1.

```
I = imread(‘cloud.jpg’);

Imshow(I);
```

uint8 and *uint16* data can be converted to double precision using the MATLAB function, double. However, converting between storage classes changes the way MATLAB and the toolbox interpret the image data. If it is desired to interpret the resulting array properly as image data, the original data should be rescaled or offset to suit the conversion. For easier conversion of storage classes, use one of these toolbox functions: *im2double*, *im2uint8*, and *im2uint16*. These functions automatically handle the rescaling and offsetting of the original data. For example, the following command converts a double precision RGB (Red Green Blue) image with data in the range [0,1] to a uint8 RGB image with data in the range [0,255].

```
RGB2 = im2uint8(RGB1);
```

To convert any image of any colour to gray picture ‘rgb2gray’ command is used:

I1 = rgb2gray(I);

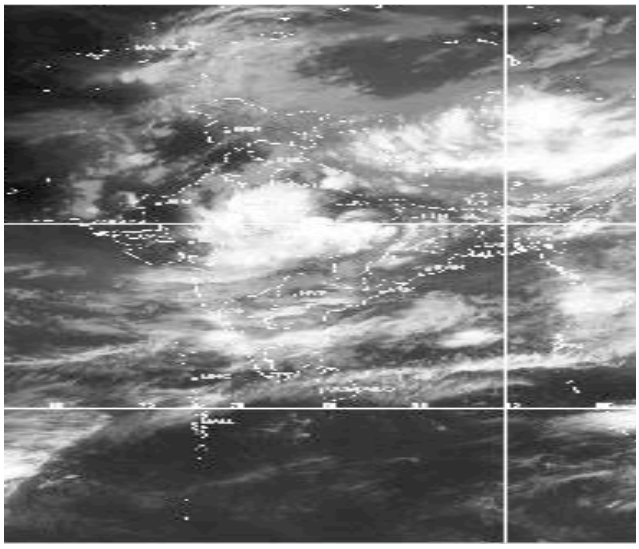


Fig.2 Gray Image

4. Histogram Equalization

As can be seen, image i.e., although pixels can be in the intensity range of 0-255 they are distributed in a narrow range. To see the distribution of intensities in image in its current state, a histogram can be created.

We can also get the histogram of image by the command 'imhist'l. And, then we can improve the intensity value over the full image by 'histeq' command.

H = figure, imhist(I1);

I2 = histeq(H);

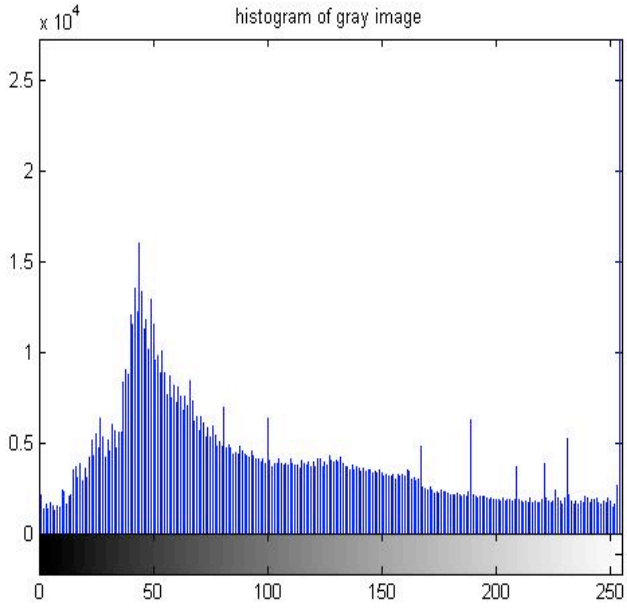


Fig.3 Histogram of image

For image enhancement, we can also subtract two images.

X= imread('cloud.jpg');

Y= imread('nocloud.jpg');

Z= imsubtract(x,y);

Imshow(z);

As well as, we can also crop the image by 'imcrop'. Also, we can add two images,

Z = imadd(x,y);

imadd adds the value of each pixel in one of the input images with the corresponding pixel in the other input image and returns the sum in the corresponding pixel of the output image. Image addition has many uses in image processing. One can also use addition to brighten an image by adding a constant value to each pixel. For example, the following code brightens image4.JPG. To improve the brightness of the image, we can also add a constant in an image.

I = imread('image4.JPG');

Z1= imadd(x,120);

Image Resizing

To change the size of an image, use the *imresize* function. imresize accepts two primary arguments viz., (i) The image to be resized and (ii) The magnification factor. The command below decreases the size of the image by 0.5 times.
F = imread('image5.JPG'); J = imresize(F,0.5); Using imresize, one can also specify the actual size of the output image. The command below creates an output image of size 100-by-150.

Y = imresize(X,[100 150])

Image Rotation

To rotate an image, the *imrotate* function can be used. *imrotate* accepts two primary arguments viz., (i) The image to be rotated and (ii) The rotation angle. The rotation angle should be specified in degrees. For a positive value, *imrotate* rotates the image counterclockwise; and for a negative value, *imrotate* rotates the image clockwise. For example, these commands rotate an image 35 degrees counterclockwise (Fig. 8).

F = imread('image5.JPG');

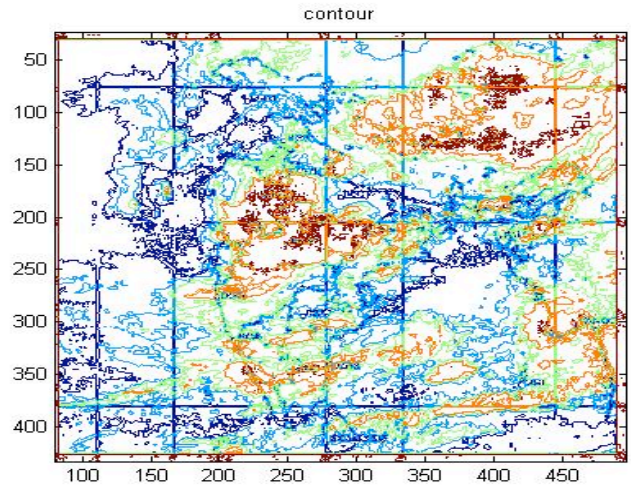
J = imrotate(I,35,'bilinear');

figure, imshow(J)

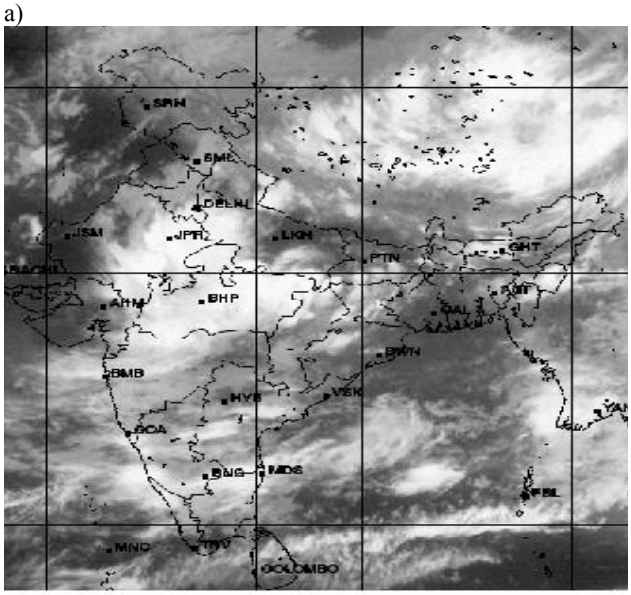
One can use the toolbox function *imcontour* to display a contour plot of the data in an intensity image. This function is similar to the contour function in MATLAB, but it automatically sets up the axes so their orientation and aspect ratio match the image. This example displays a contour plot of the image5.JPG as shown in Figure 9.

I = imread('image5.JPG');

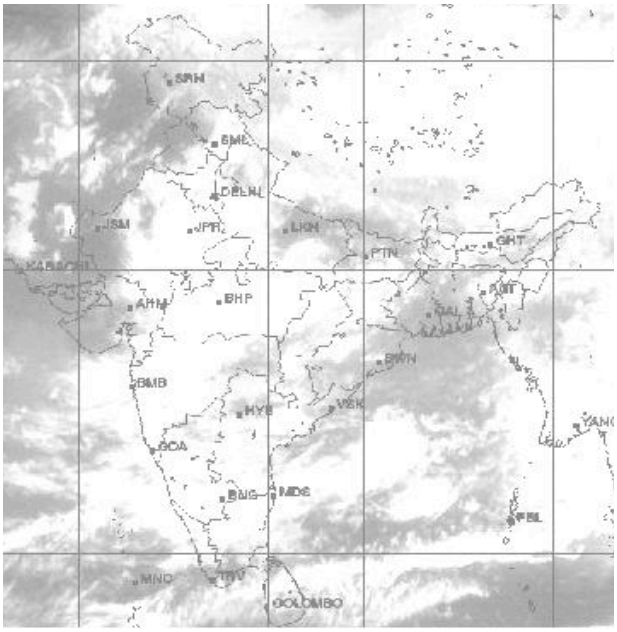
figure, imcontour(I)



a)



b)



c)

Fig. 4 (a) Image contour (b) subtraction of infrared and visible images (c) addition of infrared and visible images

5. Conclusion

Satellite imagery is very important field specially for cloud detection and weather forecasting. With the help of image processing we can enhance the satellite images and make easy the analysis of satellite images.

References

1. Gary J. Jedlovec : 'Operational Cloud Detection in GOES Imagery'.
2. Luisa Cutillo, Umberto Amato, Anestis Antoniadis, Vincenzo Cuomo, Carmine Serio: 'Cloud Detection from Multispectral Satellite Images'.
3. Dan Lubin, Richard Chamberlin, D. A. Harpar: 'Cloud detection in satellite imagery over the South Pole.
4. Ashley Davies: 'Learning Classifier for science event Detection in Remote Sensing Imagery'.
5. Vincenzo Cuomo, Carmine Serio: 'Cloud Detection from Multispectral Satellite Images'
6. Remus BRAD, Ioan Alfred Letia: 'Cloud Motion Detection from Infrared Satellite Images'.
7. D. A. Spangenberg, D. R. Doelling and V. Chakrapani: 'Nighttime Cloud Detection Over the Arctic Using AVHRR Data'.
8. Smadar Shiffman, NASA Ames Research Center: 'Cloud Detection from Satellite Imagery: a Comparison of Expert-Generated and Automatically-Generated Decision Trees'.
9. D. Nagesh Kumar, IISC Bangalore: 'Satellite Image Processing with Matlab'.
10. Rebecca Castano, Dominic Mazzoni, Nghia Tang, Thomas Doggett, Steve Chien, Ronald Greely, Ben Cichy, Ashley Davies: 'Learning Classifier for science event Detection in Remote Sensing Imagery'.
11. Craig M. Wittenbrink, Glen Langdon, California University: 'Feature extraction of clouds from GOES satellite data for integrated model measurement visualization'.
12. MathWorks Inc., *Image Processing Tool Box Users Guide*, 2001.