A new approach for Terrain Analysis of Lunar Surface by Chandrayaan-1 Data using Open Source Libraries

Hardik Acharya
IISM, Survey of India
Hyderabad, India

Amitabh
SAC, ISRO
Ahmedabad, India

T. P. Srinivasan
SAC, ISRO
Ahmedabad, India

B. GopalaKrishna
SAC, ISRO
Ahmedabad, India

Abstract — **Chandrayaan-1, India's first moon mission was launched by ISRO in October 2008. SAC (Space Applications centre) is responsible for development of software for processing data from HySI (Hyper Spectral Imager) and TMC (Terrain Mapping Camera). The present work discusses the technique and methodology for generating terrain parameters i.e. slope, aspect, relief-shade, contour etc. using Digital Elevation Model (DEM) generated from Chandrayaan-1 TMC datasets. In this paper, an algorithm and corresponding desktop application software has been developed and implemented. Preliminary testing of application using Chandrayaan-1 DEM data indicate promising results. Environment creation for execution of the code using open source technology is the challenging task, as it includes the building of open source libraries with visual studio. This paper describes the Slope, Aspect, Relief-Shade, Painted slope, Painted aspect and Painted DEM generation method and discusses the results achieved for the good evaluation of terrain.**

Keywords — Chandrayaan-1; DEM; TMC; GDAL; Qt

I. INTRODUCTION

Elevation models provide vital information for environmental, planetary and earth science studies. DEMs are important and are valuable tools for scientific analysis (e.g., large scale geomorphology). They are also required for flood modeling, land-use studies, geological applications etc. Digital Elevation Models are data files that contain the elevation of the terrain over a specified area, usually at a fixed grid interval over the surface. The intervals between each of the grid points will always be referenced to some selenographic coordinate system. DEM image for testing desktop application contains Transverse Mercator as projection system and Moon Spheroid as datum.

II. METHODOLOGY

A schematic of the workflow for Terrain Visualization is shown in figure-1. Terrain parameters like slope, aspect, relief-shade, contour is derived from DEM and further operation can be done on derived parameters to obtain slope less than 20^0, painted slope, painted aspect and painted DEM.

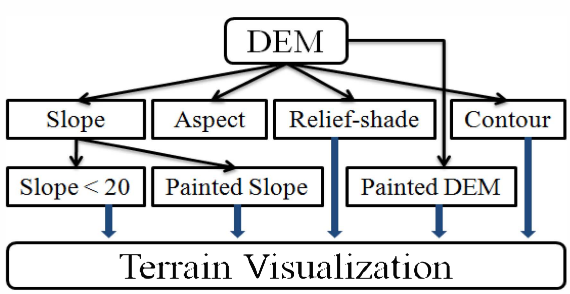


Figure - 1 : Flowchart for Terrain Visualization

A. Slope

Slope measures the rate of change of elevation at surface location and is calculated for each pixel in raster. Slope can also be said to be the minimum rate of change of elevation over each pixel and its eight neighbors. The higher the slope value, steeper the terrain, lower the slope value, flatter the terrain. Slope may be expressed as percent slope or degree slope. Percent slope is 100 times the ratio of rise (vertical distance) over run (horizontal distance), where as degree slope is an arc tangent of the ratio of rise over run [1].

Terrain derivatives can be obtained by passing a local window 3 * 3 on the DEM and estimating the relationship between the central pixel and its neighbors [2]. Slope used to be derived manually from a contour map. This practice has become rare with the use of GIS. It computes slope and aspect for discrete units such as pixels of an elevation raster.

Figure-2 shows the 3 * 3 kernel, having elevation value from e_0 to e_8. Centre pixel value i.e. e_4 is computed using its neighboring pixels. Figure-3 represents the resultant slope obtained from slope in x and slope in y direction.

e_0	e_1	e_2
e_3	e_4	e_5
e_6	e_7	e_8

Figure -2: 3*3 kernel

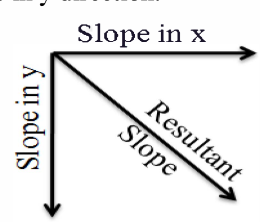


Figure-3: Resultant slope

Different methods have been proposed for calculating slope and aspect from an elevation raster. The method used in this paper is Horn's Algorithm. This method use a 3 * 3 moving window to estimate the slope and aspect of the centre pixel [1]. The slope at e_4 in figure-2 can be computed by :

$$S = \frac{\sqrt{n_x^2 + n_y^2}}{8d} \qquad (1)$$

$$D = \text{arc tan}\left(\frac{[n_y]}{[n_x]}\right) \qquad (2)$$

Where, e_i = pixel value, d = distance between adjacent pixels, S = Slope, D = Slope's directional angle

$n_x = (e_0 + 2e_3 + e_6) - (e_2 + 2e_5 + e_8)$

$n_y = (e_6 + 2e_7 + e_8) - (e_0 + 2e_1 + e_2)$

TABLE I

ALGORITHM FOR SLOPE GENERATION

Input: DEM raster image
Output: Slope /Aspect raster image

1. Read raster image into memory using GDAL library.
2. Get rasterband and transformation parameter of an image.
3. Create a buffer for
   a. for output slope raster image.
   b. for 3 * 3 window mask(kernel).
4. Run loop over an entire image
   a. Set border pixel values to zero.
   b. create sub-loop for kernel pixels.
   c. computation of centre pixel of kernel by Mathematical algorithm using neighborhood pixels.
   d. Go to TABLE -II (Only in case of aspect calculation)
   e. Assign computed central pixel value to slope/Aspect buffer.
5. Fetch rasterband and transformation parameter from input image and assign it to output image.
6. Write an image using GDAL library to get final Slope /Aspect image.

Above algorithm is used for calculating slope and aspect in this study and GUI is developed respectively.

B. Aspect

Aspect is directional measure of slope. It starts with 0^0 at the north, moves clockwise and ends with 360^0 also at the north. A common method is to classify aspects into the four principal directions (north, south, east and west) or eight principal directions (north, north-east, east, south-east, south, south-west, west and north-west) and to treat aspects as categorical data [1].

TABLE II

ALGORITHM FOR ASPECT GENERATION

Algorithm for converting D to aspect

```
aspect = D * Radian_to_Degrees
 if(nx == 0)
        if(ny < 0)
              aspect = 180
        else
              aspect = 360
        else if (nx > 0)
              aspect = 90 - aspect
        else
              aspect = 270 - aspect
   aspect buffer [j]  =  aspect
```

C. Relief-Shade

The Relief-shade function obtains the hypothetical illumination of a surface by determining illumination values for each pixel in a raster. It does this by setting a position for a hypothetical light source and calculating an illumination value of each pixel in relation to neighboring pixels. It can greatly enhance the visualization of a surface for analysis or graphical display, especially when using transparency [3].

There are four factors which controls the visual effect of relief-shading:

1. Sun's Altitude
2. Sun's Azimuth
3. Slope
4. Aspect

1) Computing Sun's Altitude:

Altitude of celestial body is an angular distance above the horizon, measured on the vertical circle passing through the body. To calculate sun's altitude, convert an altitude angle to the zenith angle by considering 90^0 complement to altitude.

$$Z\_DEG = 90^0 - Altitude$$

2) Computing Sun's Azimuth:

Azimuth of sun is an angle at zenith between observer's meridian and vertical circle passing through the sun, measured in clockwise direction from north point of horizon. Sun's altitude and azimuth are defined in degrees. The Relief-shade formula requires this angle to be in units of radians.

$$AZ\_DEG = 360^0 - Azimuth + 90^0$$

The Relief-shade Algorithm:

To calculate the shade value, first the altitude and azimuth of the sun is needed. These values will be processed along with calculations for slope and aspect to determine the final relief-shade value for each pixel in the output raster.

The algorithm for calculating relief-shade value is :

Relief-Shade = 255 * (sin(Z_DEG * deg_to_radians) * sin(slope * deg_to_radians) +cos(Z_DEG * deg_to_radians) *cos(slope * deg_to_radians) * cos((AZ_DEG - 90) * deg_to_radians - aspect)

TABLE III

ALGORITHM FOR RELIEF-SHADE GENERATION

Input: DEM raster image
Output: Relief-shade raster image

1. Read raster image into memory using GDAL library.
2. Get rasterband and transformation parameter of an image.
3. Provide Azimuth and altitude angle of illumination source.
4. Create a buffer for
   a. for output slope raster image.
   b. for 3 * 3 window mask(kernel).
5. Run loop over an entire image
   a. create sub-loop for kernel pixels.
   b. computation of centre pixel of kernel by Mathematical algorithm using neighborhood pixels.(slope and aspect value is required in mathematical algorithm of Relief-Shade)
   c. Assign computed central pixel value to relief-shade buffer.

6. Fetch rasterband and transformation parameter from input image and assign it to output image
7. Write an image using GDAL library to get final relief-shade image.

D. *Painted DEM, Painted Slope and Painted Aspect*

Painted DEM depict elevation as bands of colour, to enhance elevation zones so map readers can better see differences in relief. The colours selected for the tints are assumed to relate to the ground cover typically found at various elevations in the area being mapped. A typical colour scheme progresses from dark red for lower elevations up through blue, sky-blue, green, yellow and white for upper elevations. To create Painted-DEM, we first have to create a text-based colour configuration file (painted_dem.txt), containing the association between elevation values and colours. The file generally contains 4 columns per line: an elevation value and the corresponding red, green and blue value between 0 and 255 (RGB). The numbers in the text file define a gradient that will blend 5 colours over 2600 meters of elevation as shown in figure-4(b) [4].

The same applies for Painted-slope as its value ranges from 0 to 90 degrees. Lower slope 0^0 as white and colour-scheme progresses according to text file (painted_slope.txt) up to red being the highest slope 90^0.

In painted aspect, green, black, blue and red colour is used to represent north, east, south and west directions respectively as shown in figure-4(f).

E. *GDAL and Qt with Visual Studio 2010*

1) *GDAL (Geospatial Data Abstraction Library):*

It is a library for reading and writing raster geospatial data formats. As a library, it presents a single abstract data model to the calling application for all supported formats. The related OGR library (which is part of the GDAL source tree provides a similar capability for simple features vector data [5]. C++ open source libraries providing read and write access to wide variety of raster and vector file formats. All images were read using GDAL libraries and then processed according to mathematical algorithm. Finally output image is again written using GDAL libraries.

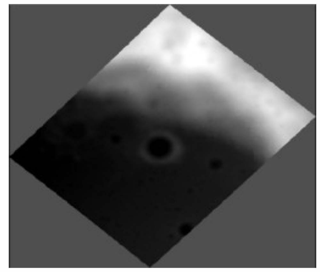
2) *Qt:*

It is a cross-platform application and UI(User Interface) framework for development based on C++. A solid understanding of the basics of C++ is essential for those planning on using Qt for C++. The overall development effort is minimal since Qt API are easy to understand and application functionality can be implemented with a smaller amount of code [6]. GUI (Graphical user Interface) is developed for all the tools described above. User can fetch any of the DEM file and save to particular desired location.

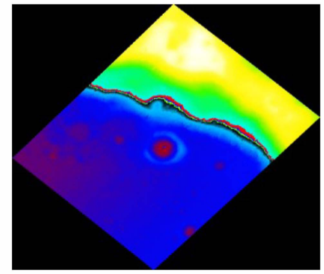
The library is multiplatform and works under different operating systems (Linux, Unix, Windows) and different architectures (32 or 64 bits) to get rid of hardware constraints. The multiplatform aspect is also very important because it imposes to follow strict design and coding rules thus leading to a robust system less sensitive to particular platform specific [7].

III. RESULTS AND DISCUSSIONS

Results shows the parameters derived from the Digital Elevation Model using VC++.

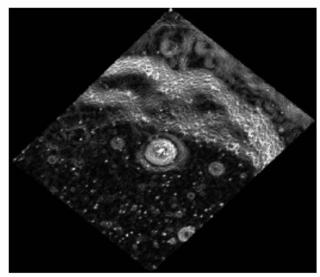


4(a)Chandrayaan-1 test DEM input

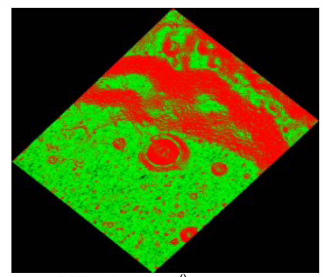


4(b) Painted DEM

Figure-4(a) is an input DEM from which the parameters are derived using desktop application. Figure-4(b) Painted DEM applies color to different elevation zones as shown in color bar. The use of well-chosen color palate can help viewers to see the progression in elevation.

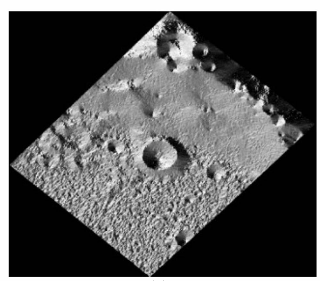


4(c) slope

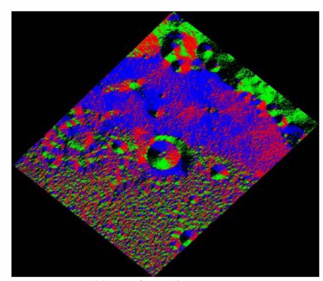


4(d) slope < 5^0 → green

Figure-4(c) is slope image as per an algorithm described in Table I. Figure-4(d) shows slope < 5^0 which is painted in green describes the flatter terrain.

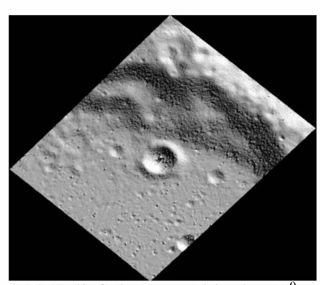


4(e) Aspect

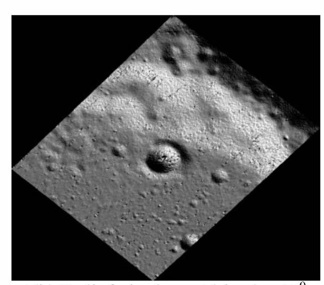


4(f) Painted Aspect

Figure-4(e) shows the aspect image as per an algorithm described in Table II. Figure-4(f) Classify aspects into the four principal directions (north, east, south and west which are painted in green, black, blue and red respectively)

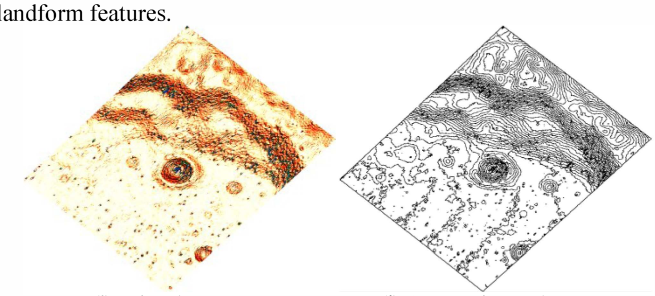


4(g) Relief-shade→ Altitude=45^0, Azimuth = 315^0



4(h) Relief-shade→ Altitude=45^0, Azimuth = 135^0

Figure-4(g) and 4(h) simulates how the terrain looks with the interaction between sunlight and surface features. Image shows variations in elevation based on a user-specified position of the sun. Areas that would be in sunlight are highlighted and areas that would be in shadow are shaded. It helps viewers recognize the shape of landform features.



4(i) Painted-DEM   4(j) Contour interval = 50 m

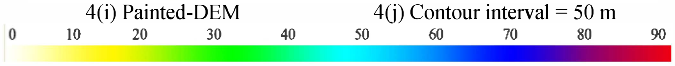
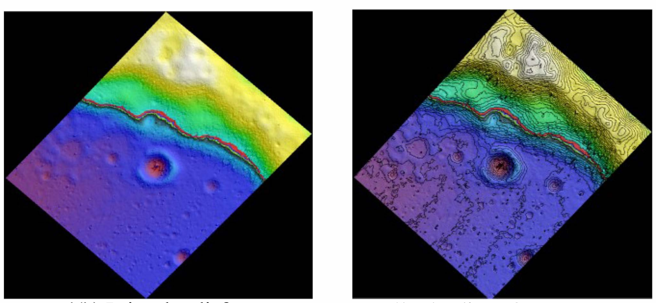


Figure-4(i) represents slope in degree with different colors as shown in color bar. Figure-4(j) contour lines connect point of equal elevation and are derived at an interval of 50m. Contour lines are closely spaced in steep terrain and are curved in upstream direction.



4(k) Painted- relief   4(l) Blending of parameters

Figure-4(k) represents the relief-shade with color coded elevation value. Figure-4(l) Blending of parameters for Terrain Visualization.

Figure - 4 Derived terrain parameters from DEM

IV. GRAPHICAL USER INTERFACE

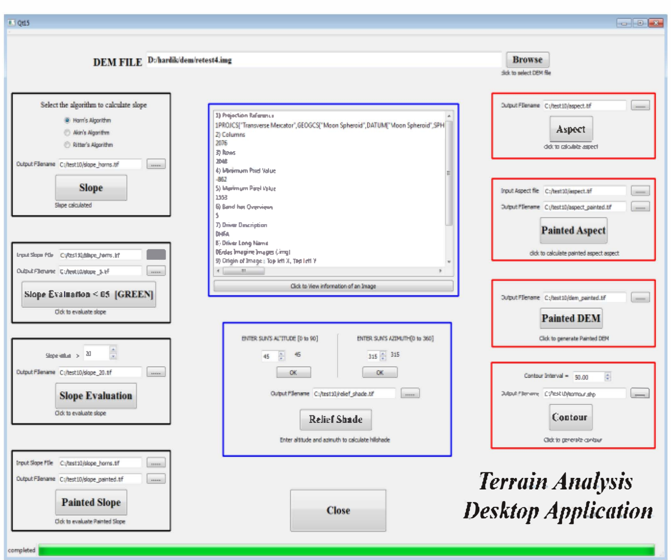


Figure - 5

Figure - 5: Graphical User Interface as desktop application for DEM derived parameters using Qt as GUI.

V. CONCLUSION

Terrain parameters from Chandrayaan-1 DEM are obtained using this desktop application. For future lunar missions, this software can be used for deriving terrain parameters of lunar surface.

The output of slope calculation will be ranging from 0^0 to 90^0. Slope greater than 15° is considered as a hazard for any landing mission. A tool is also developed in GUI for direct evaluation of slope, in which slope less than 5^0 is painted in green color and above 5^0 is red as shown in figure-4(d). It can be edited to 10^0, 15^0 or 20^0 as per user requirement. Similarly relief-shade can also be generated by providing azimuth and altitude angle according to user's requirement. It is a cost free desktop application, and more tools can be developed and integrated into GUI. Experiments can be done using newly developed as well as available algorithms.

REFERENCES

[1] Kang-Tsung(Karl) Chang, Introduction to geographic Information Systems, pp - 286-294

[2] T. Hengl, S. Gruber, and D. P. Shrestha, ―*Digital Terrain Analysis in ILWIS*, International Institute For Geo-Information Science and Earth Observation, Enschede, The Netherlands, Jul-2003.

[3] "How Hill shade works," ArcGIS9.2 Desktop Help, http://webhelp.esri.com/arcgis desktop/9.2/index

[4] Thematic mapping blog : http://blog.thematicmapping.org

[5] "GDAL - Geospatial Data Abstraction Library," http://www.gdal.org

[6] "Qt - Cross platform application and UI framework for developers using C++" : http://qt-project.org

[7] Emmanuel Christophe and Jordi Inglada," *Open Source Remote Sensing: Increasing the usability of cutting - edge Algorithm*, " in IEEE Geosciences and Remote Sensing Society Newsletter, pp - 9-15, March 2009

[8] Sannidhi K S, M. Z. Kurian. "Creation *of Digital Elevated Model using lunar images of Chandrayan-1*," in International Journal of Electrical and Computer Engineering (IJECE), Vol.1, No.2, December 2011, pp.182 - 186

[9] B. Gopala Krishna, Amitabh, Sanjay Singh, P. K. Srivastava and A. S. and Kiran Kumar, "*Digital Elevation Models of the Lunar Surface from Chandrayaan-1 Terrain Mapping Camera (Tmc) Imagery – Initial Results*", in 40th Lunar and Planetary Science Conference (2009)