DEVELOPING A GENERIC FRAMEWORK TO SUPPORT MULTI-DIMENSIONAL EARTH OBSERVING SYSTEM DATA IN GIS APPLICATIONS

by

Yunfeng Jiang

A Thesis
Submitted to the
Graduate Faculty
of
George Mason University
in Partial Fulfillment of
The Requirements for the Degree
of
Master of Science
Geographic and Cartographic Sciences

Committee:

_____	Dr. Chaowei Yang, Thesis Director
_____	Dr. George Taylor, Committee Member
_____	Dr. Ruixin Yang, Committee Member
_____	Dr. Anthony Stefanidis, Department Chairperson
_____	Dr. Donna M. Fox, Associate Dean, Office of Student Affairs & Special Programs, College of Science
_____	Dr. Peggy Agouris, Dean, College of Science
Date: _____	Fall Semester 2015 George Mason University Fairfax, VA

Developing a Generic Framework to Support Multi-dimensional Earth Observing System Data in GIS Applications

A thesis submitted in partial fulfillment of the requirements for the degree of Master of Science at George Mason University

By

Yunfeng Jiang
Bachelor of Science
The China University of Mining and Technology, 2004

Director: Chaowei Yang, Professor
Department of Geography and Geoinformation Science

Fall Semester 2015
George Mason University
Fairfax, VA

DEDICATION

This thesis is dedicated to my family including my parents, who inspire and encourage me to set high goals, gives me the confidence to achieve them, leave me more free space, and who are always my spiritual support anytime. It is also dedicated to my wife, who always believes me, and get along with me all those years, especially during the hard time.

ACKNOWLEDGEMENTS

I would like to thank Dr. Chaowei Yang, who has been the thesis supervisor and my committee chairman. He provides me the opportunities and gives me tremendous help during my graduate studies at George Mason University.

I would like to thank all committee members for their guidance in the thesis, Dr.George Taylor, Dr.Ruixin Yang, who were more than generous with their expertise and previous time in commenting, reading, encouraging, and being patience throughout the entire process.

I would like to thank all members of the Center for Intelligent Spatial Computing for Water/Energy Science (CISC) at George Mason University, Min Sun, Jing Li, Kai Liu, Zhipeng Gui, Zhenglong Li, Chen Xu, Manzhu Yu, Qin Han, Fei Hu, Yongyao Jiang, Mengcao Xu.

Finally, and most importantly, I would like to thank my parents and wife. They have been always supporting me in my life.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS

Earth Observing System ..... EOS
Geospatial Data Abstraction Library ..... GDAL
Hierarchical Data Format ..... HDF
Atmospheric Scientific Data Center ..... ASDC
Geographic Information System ..... GIS

ABSTRACT

DEVELOPING A GENERIC FRAMEWORK TO SUPPORT MULTI-DIMENSIONAL EARTH OBSERVING SYSTEM DATA IN GIS APPLICATIONS

Yunfeng Jiang, M.S.

George Mason University, 2015

Thesis Director: Dr. Chaowei Yang

Earth Observing System (EOS) data are expanding at an unprecedented rate due to the fast development of advanced data acquisition technologies. These data provide valuable, long-term record of change and dynamics about our Earth, and therefore are paramount in addressing key national and global challenges in climate change, water use and quality, natural disasters, weather forecasting and warnings, renewable energy, agriculture, forestry and natural ecosystems, coasts and oceans, and national security. As a result, they have been increasingly used in various GIS applications by both government and science communities. However, many varied formats and standards have been defined to organize and store the EOS data that are highly tailored for different applications by different organizations over the past decades. Many of these data are in old formats, and specialized geospatial tools are required to interpret and use them. This makes it difficult to incorporate EOS into GIS and it very ineffectively to analyze in either commercial or open-source GIS tools. On the other hand, most GIS systems cannot comprehensively

process and utilize all types of EOS data, and there are always unexpected issues and errors while importing and manipulating EOS data. To reconcile the conflicts between EOS data and GIS systems, initiatives have been made for developing a general methodology to solve EOS data compatibility in GIS using common standards. However, no solutions are currently available to support the processing of all types of EOS data products. The objective of this research is to explore the barriers and strategies of integrating various types of EOS data in GIS applications. Specifically, the research investigates and solves three key technical problems including: (i) designing a generic and heuristic plug-in framework for consuming different types of EOS data; (ii) developing a series of functions to fix the problem occurring when using EOS data in GIS applications; (iii) optimizing HDF4/HDF5 data drivers of GDAL for enhancing its capability of handle EOS data; and (iv) developing an open source GIS extension to enhance the capability of GIS systems in accessing EOS data.

One research result of this thesis is optimized source code of Geospatial Data Abstraction Library (GDAL) commonly used in most GIS systems for handling geospatial raster and vector data, without impacting the original function on reading non-EOS data products. The optimized GDAL fixes the issues of HDF4 and HDF5 data drivers used to access HDF datasets and overcome limitations in processing multiple dimensional datasets posed by the current GDAL version. Finally based on the optimized GDAL, an open source extension is developed to support the access of more EOS data of different types and fill in the gap between GDAL and commercial GIS software (e.g. ArcGIS) or open source GIS projects (e.g. QGIS). A series of EOS data products collected from NASA's

Atmospheric Scientific Data Center (ASDC) are selected as study cases for demonstrating the effectiveness and applicability of the proposed framework and tools. The enhanced GDAL and GIS extension enable and encourage more GIS users to use EOS data in GIS software for different research or applications. Additionally, a series of Application Programming Interfaces (APIs) are provided to allow other developers in GIS communities to integrate these interfaces into their GIS applications. It is concluded that the proposed plug-in framework can be effectively applied to different domains for handling the current problems or limitations of interpreting multi-dimensional dataset, without compromising their original functions.

CHAPTER 1 INTRODUCTION

1.1 Research problem

NASA's Earth Observing System (EOS) produces a high volume of remote sensing data which record and capture long-term global observations of land surface, solid earth, atmosphere and oceans. With the fast development of remote sensing technology, EOS data are expanding at an unprecedented rate through multiple space-based instruments and satellite platforms. For example, some NASA missions (e.g. Soil Moisture Mission) provide more than 1 terabyte of data per day (Board 2007). In recent years, the importance of EOS data has been widely acknowledged by both government and science communities. They are not only useful for Earth system science research but also benefit a variety of applications, such as disaster response, environmental planning, global change, insurance, and private investment (Farr 2011).

Meanwhile, many 21st century challenges require the timely integration of vast amount of geospatial information through Geographic Information System (GIS) to address global and regional challenges such as emergency response and planning. GIS has been widely used to interrelate multiple types of information assembled from a variety of EOS data to visualize, query, overlay, and analyze these data for understanding relationship, patterns and trends, making it valuable to a wide range of scientific, academic and private entities (Bagwell 2011). Therefore, NASA has increasingly used

GIS to process and analyze the EOS data to discover patterns and knowledge about the Earth's surface. Currently we have many commercial and open source GIS tools for mapping, analyzing and visualizing geospatial data, such as ESRI ArcGIS, one of the most popular commercial GIS products, and QGIS, a widely used open-source GIS package.

In as early as 2010, fifty remote sensing scientists and GIS experts participated in a joint NASA-ESRI workshop at the headquarters of ESRI to explore benefits and barriers of integrating EOS data into GIS (Farr 2011). Despite of the fact that remote sensing and GIS work well in some applications (Lo et cl. 1997; Shalaby and Tateishi 2007; Petropoulos et cl. 2015), there are only limited communication among the remote sensing and GIS communities (Goodchild 1994). As a result, many different data models, formats, standards, tools, services, and terminology are defined and designed to organize and store remote sensing data by GIS and remote sensing communities through using specific tools to solve particular issues. Therefore, different data sets with generic GIS tools are made only compatible with each other. For example, Atmospheric Scientific Data Center (ASDC) of NASA has a diverse set of EOS data products, such as Advanced Spaceborne Thermal Emission and Reflection Radiometer (ASTER), Clouds and Earth's Radiant Energy System (CERES), Multi-angle Imaging SpectroRadiometer (MISR), Moderate Resolution Imaging Spectroadiometer (MODIS), and Measurement of Polltution in the Troposphere (MOPPIT). However, over the past decades, many varied formats and standards have been defined to organize and store these EOS data that are highly tailored for different applications by different organizations. Many of these data

are in old formats and specialized geospatial tools are required to parse and use them. This makes it difficult to incorporate EOS data into GIS, and therefore cannot be efficiently and effectively visualized or analyzed in either commercial or open-source GIS tools. On the other hand, most main-stream GIS systems cannot process and utilize all types of EOS data, and there are always some unexpected issues or errors while importing EOS data files. To reconcile the conflictions between EOS data and GIS systems, some initiatives have been made for developing a generic methodology to solve EOS data compatibility problems in GIS. For example, HDF group is trying to propose a comprehensive methodology to better support EOS data with HDF formats. Unfortunately, no solutions are available to get out of the dilemma yet.

Among various open source software or tools handling EOS data, GDAL is a widely used one that supports GIS applications for accessing and processing both raster and vector data (GDAL 2015). For example, ArcGIS relies on GDAL to read EOS data in Hierarchical Data Format (HDF), the primary data format of EOS data. Unfortunately, GDAL still has some problems in dealing with HDF data, especially multiple dimensional, e.g. 3D, 4D, and 5D. Therefore, ArcGIS supports only a part of NASA Hierarchical Data Format-Earth Observing System (HDF-EOS; the standard format to store data collected from three EOS satellites, including Terra, Aqua and Aura) datasets. In addition, various problems often occur while importing EOS data in GIS systems (ESRI 2015). For example, missing spatial reference, failing to retrieve NoData value (the absence of a recorded value), and misunderstanding multi-dimensional variables. In the past years, some GIS software vendors have been working on fixing these problems

by developing their specific version of GDAL to address these problems for consuming more EOS data. For example, ESRI developed ESRI-specific GDAL to enhance its capability of interpreting EOS data in ArcGIS Desktop 10.3. However, due to the heterogeneity of different data products from different data centers, most problems are still remaining thus impede the usage of NASA EOS data products. Furthermore, integrating the fixes in a particular branch of GDAL (e.g. the ESRI-specific version) into the main GDAL open-source trunk is time-consuming and labor intensive. To sum up, it is quite a challenge to fill in the gap between GDAL and most of GIS software and seamlessly integrate them.

1.2 Proposed research

Facing these challenging problems, it is urgent to develop a generic framework for GDAL to enhance the interoperability of EOS data in GIS. In this research, an xml-base plug-in framework is proposed to fix these corresponding problems according to different data products by using an xml file that is used to heuristically invoke a series of related functions. Specifically, rotate an image by 90 degree, invert an image upside down, or interpret 3D/4D/5D variables, and so on. In order to demonstrate its feasibility and the functionality, GIS extensions are developed on the basis of the proposed framework.Unlike traditional EOS geospatial tools that can only consume partial EOS data, the proposed framework enables GIS to support more EOS data products. Additionally, the proposed framework is very flexible and extendable that users can manually add a new function for copying with new emerging problems as well as new EOS data products. In GDAL, HDF4/HDF5 data drivers are optimized for addressing the

problems mentioned above and improving its capability of processing multi-dimensional variables at the level of source code. The enhanced GDAL also allows GIS developers to invent new modules for consuming different EOS data in their GIS applications.

1.3 Thesis outline

The thesis consists of seven sections. This section introduces the study by analyzing challenges, potential solutions, and proposed research. Section 2 reviews related work, including EOS data and its major formats, tools and libraries for parsing these data, describes the problems while using GIS to process them, and relevant work on addressing these problems. Based on the existing research efforts, Section 3 proposes a generic xml-based plug-in framework to address the remaining problems on incorporating GIS and EOS data. In Section 4, the details of system implementations and optimizations are presented. Using various case studies, Section 5 demonstrates the effectiveness and efficiency of the proposed framework and corresponding tools. Section 6 discusses the contributions of the study. Section 7 summarizes research and discusses future research.

CHAPTER 2 RELATED WORK AND CHALLENGES

This chapter introduces EOS data and its major formats, tools and libraries for parsing EOS data, and describes the problems while using GIS to process them. It also presents relevant work in the past decades to provide a good understanding of the context and background of the proposed framework.

2.1 EOS data

NASA had launched many satellites to help us develop a scientific understanding of the Earth system and its response to natural and human-induced changes to enable improved prediction of climate, weather, and natural hazards for present and future generations. As a major component of NASA Earth Observatories (**Error! Reference source not found.**), EOS was conceived in the 1980s and began to take shape in the early 1990s. It includes a series of polar-orbiting and low inclination satellites for long-term global observation of land surface, biosphere, solid Earth, atmosphere, and ocean, which improved the understanding of the Earth as an integrated system.

Figure 1. NASA Earth science division operating missions [1]

Satellite instruments or sensors are accommodated on NASA eighteen flagship EOS satellites, including Aqua, Terra and Aura. These instruments are named according to 1) the satellite or platform, and 2) the capabilities of the sensor or instrument (Table 1).

Table 1. Selected EOS satellites, related sensors (data products) and applications

Satellites	Sensors	Applications
Terra	CERES (Clouds and Earth's Radiant Energy System)	Study clouds, and its impact on Earth's energy
	MISR (Multi-angle Imaging SpectroRadiometer)	Cloud heights, the heights of plumes of smoke, aerosols and atmospheric particles
	MOPITT (Measurements of Pollution in the Troposphere)	Air pollution levels in the lower atmosphere, e.g. C02
	MODIS (the Moderate Resolution Imaging SpectroRadiometer)	Sea surface temperature, snow cover, fire, vegetation
	ASTER (Advance Spaceborne Thermal Emission and Reflection Radiometer)	Land Surface temperature, emissivity, reflectance, and elevation
Aqua	CERES	Clouds , and its impact on Earth's

[1] http://eospso.nasa.gov/

		energy
	MODIS	Sea surface temperature, snow cover, fire, vegetation
	AMSU-A (Advanced Microwave Sounding Unit)	Temperature , precipitation, atmospheric water vapor
	HSB(Humidity Sounder for Brazil)	Atmospheric water vapor (humidity)
	AIRS (Advanced Infrared Sounder)	Temperature and humidity, clouds
	AMSR-E (the Advanced Microwave Scanning Radiometer for EOS)	Sea ice levels, water vapor, wind speed, Global rainfall, temperature
Aura	HIRDLS (High Resolution Dynamic Limb Sounder)	Temperature
	MLS (Microwave Limb Sounder)	Carbon monoxide and ozone
	TES (Tropospheric Emission Spectrometer)	Ozone and carbon monoxide
	OMI (Ozone Monitoring Instrument)	Total ozone and aerosol index

For example, Terra (formerly EOS AM-1), the satellite launched in 1999 to observe and monitor Earth system, collects multiple types of data on the state of the atmosphere, land, and oceans, as well as their interactions with solar radiation and with one another. This satellite has five sensors on board, including ASTER, CERES, MISR, MODIS, and MOPPIT (Figure 2). Each sensor has its own specialized capabilities and purposes, and produces different types of data products for various applications (Table 1). For example, MOPPIT captures the lower atmosphere, especially focusing on distribution, transport, source, and sinks of carbon monoxide in the troposphere and enables us to observe how it interacts with land and ocean biospheres.

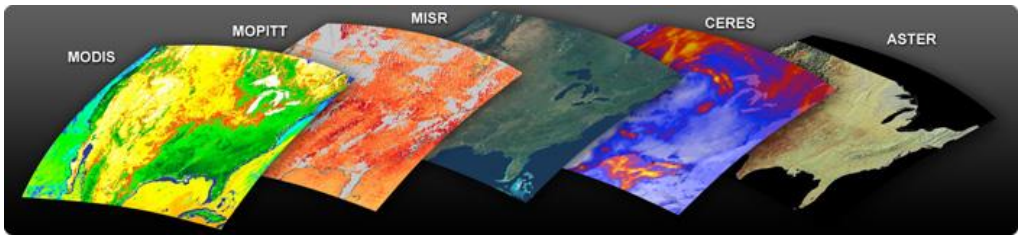


Figure 2. Five Terra onboard sensors

2.2 EOS data formats

Different instruments or sensors produced different types of EOS data, and they are usually consistent in data format. The standard data format for all NASA EOS data products is HDF. It is a multi-objects -based data format which are originated from the National Center for Supercomputing Applications (NCSA) and developed and updated by the non-profit organization, HDF Group (Bagwell 2011). Currently, there are two distinct varieties of HDF, known as HDF4 (version 4 and earlier) and the newer HDF5 (HDF Group 2015).

- HDF4: HDF4 is an old version of HDF format that supports a variety of data models, including multi-dimension arrays, raster images, and tables. It is very flexible for developers and users to quickly add new data models. Nevertheless, many problems of HDF4 format remain. For example, it is not a very clear object model, which makes it difficult for continued supports and improvements.
- HDF5: HDF5 format is designed to address some limitations of HDF4 and make it easy to be utilized by modern systems and applications. It simplifies the file structure by using two major types of objects: dataset, which is a multidimensional array of records, and group, which is a structure for grouping objects. Therefore, it is in a truly hierarchical, file system-like data format.

Different from traditional file systems, the data user does not need to know how a data object is stored in the file and where its physical location is, while accessing data in HDF (HDF 2015). Furthermore, The API supporting HDF5 is also object-oriented with respect to datasets, groups, attributes, types, data spaces and property lists, which make it simple to do further development.

Some of the features of HDF that make it a widely used data format to store scientific data are (Di 2000):

- HDF makes it possible for programs to obtain information about the data from the data file itself, rather than from another source.
- HDF standardizes the format and descriptions of many types of commonly used data sets, such as , scientific data arrays, tables, and text annotations, as well as several types of raster images and their associated color palettes (HDF, 2015)
- HDF is a platform independent file format. It can be used on many different computers, regardless of the operating system that a machine is running.
- Each data object in HDF file can have a user-defined name, which makes it easy to retrieve or port data to many different platforms.
- It is easy to use and implement, due to open source libraries and good documentation. New data models may be added to HDF by either the development team or HDF users.

There are many open source tools for manipulating and visualizing HDF data, such as HDF Group data products and OPeNDAP. Because many earth science data structures need to be geo-located, the HDF group developed the HDF-EOS format with

additional conventions and data types for HDF files based on HDF format. HDF-EOS (Hierarchical Data Format-Earth Observing System) is a self-describing file format for transferring various types of data between different machines based upon HDF. For example, NASA uses it as the primary data format of EOS data. It is a standard format to store data collected from EOS satellites: Terra, Aqua and Aura support three geospatial data types: grid, point, zonal, and swath, providing uniform access to diverse data types in geospatial context. HDF-EOS is designed to store and organize large amount of numeric data and supported by many commercial and non-commercial software platforms (Folk 1999).

2.3 EOS data tools

To make EOS data usable to Earth Science research communities and the general public, a variety of tools handling EOS data have been developed to easily compare, analyze, and visualize data from a variety of EOS data sets. Some of them are general tools that allow user to browse and edit general HDF-EOS data files no matter where they are from, e.g. HDF View that is developed by HDF Group for supporting HDF4 and HDF5 format (HDF group 2015). Others are designed for specific EOS data product, e.g. AIRS tool that is only available to manipulate AIRS data files. More than often, both types of tools work with EOS data very well and allow the user to manipulate and visualize EOS data. But, only the limited number of functions are provided by the two types of tools, such as, reformatting, re-projection, mosaicking, data quality assessment, image processing, and multispectral analysis. In addition, with the increasingly growing demands of big data, single pure EOS data tool is not enough for many research that

require integrating analysis of multi-disciplinary and multi-source data. Therefore, scientists and engineers resort to GIS tools that are widely used in both science communities and industry.

The past decades witnessed the tremendous development of GIS. It enables to provide more powerful functions and capabilities in visualizing and analyzing EOS data than these traditional HDF data tools. Users prefer to use GIS rather than HDF tools for the following reasons (e.g. ArcGIS): 1) GIS can interrelate multiple types of information assembled a diversity of data source (e.g. GIS format) to visualize, query, overlay, and analyze data; 2) GIS provides more advanced functions for spatial data in managing, mapping, modeling and making decisions. However, the potential value of EOS in GIS applications has not been fully explored, due to a series of problems occurring when integrating these data. Fortunately, some initiatives are underway to develop a common methodology to solve parts of the problems. For example, in order to reduce the difficulty of accessing and increasing the public use of EOS data. NASA funded a project that was to develop a NASA HDF-EOS Web GIS Software Suite (NWGISS) for providing standards-based access and services to NASA EOS data for the GIS user community according to OGC specifications (McDonald and Di 2003). The project integrates with Grid technology for the purpose of sharing data and computing resources among NASA data centers, and implements geospatial web services.

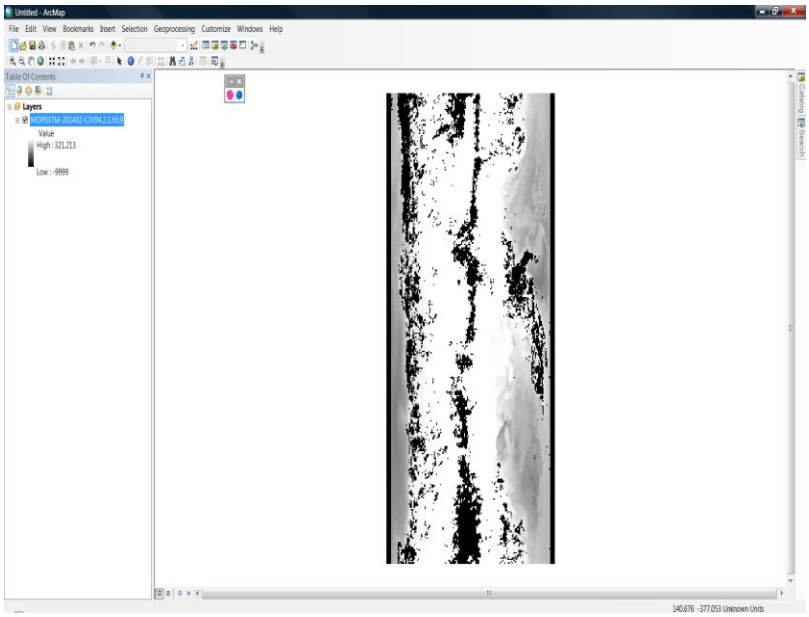
2.4 Common challenges when using EOS data in GIS

ASDC data products such as MOPITT, TES, CERES, and MISR, are stored in HDF4, HDF5 or NetCDF format. MOPITT level 3 version 5 (in HDF4), MOPIT level 3

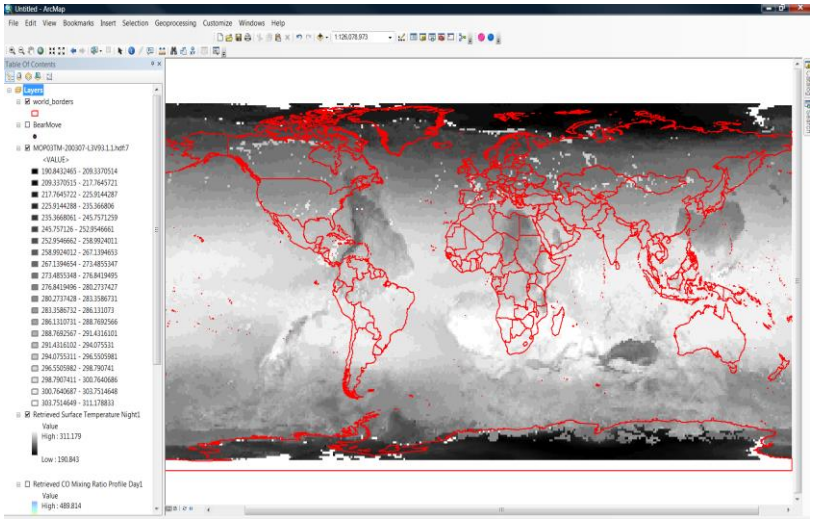
version 6 (in HDF5) and TES level 3 (in HDF5) data products are selected as study cases in the current development. The conducted investigation reveals that there are a variety of problems occurring when opening these datasets in GIS applications. The major problems of accessing EOS data are summarized and categorized into four groups:

- Problem 1: 90 degrees rotated image and inverted upside down image

Both two problems happened at the pixel level when displaying or visualizing selected data files. Specifically, the images generated from EOS data would be rotated by 90 degrees or inverted upside down in a GIS system (Figure 3.A and 3.B). The reason why the image is rotated by 90 degrees including: 1) Dimension size in metadata was wrong. For example, width and height should be (360,180), but was written as (180, 360). 2) Pixel values in the image were also written incorrectly. 3) Sometimes GDAL returns null value of the dimension size, because the field in metadata that GDAL reads is "band", but the field in HDF file store those information is actually "nprs,", "nlon,", and "nlat,". As for image-inverted problem, it is found that pixel value in the image was written incorrectly.



A. 90 degree rotated image



B. Upside down inverted image

Figure 3. Rotated images generated by EOS datasets

- Problem 2: Missing geo-reference information

Geo-reference or spatial reference, related information to geographic location, is commonly used in GIS fields for describing the process of associating a raster/vector of a map with spatial location in physical space. EOS data ought to have its own geo-reference information and are expected to be extracted by GIS software. However, sometimes, such information is missing when opening the targeted datasets in GIS. Our investigation reveals that GDAL actually does not support all HDF data products very well in obtaining correct geo-reference information. Therefore, sometimes, GIS systems fail to recognize these HDF spatial information (Figure 4).

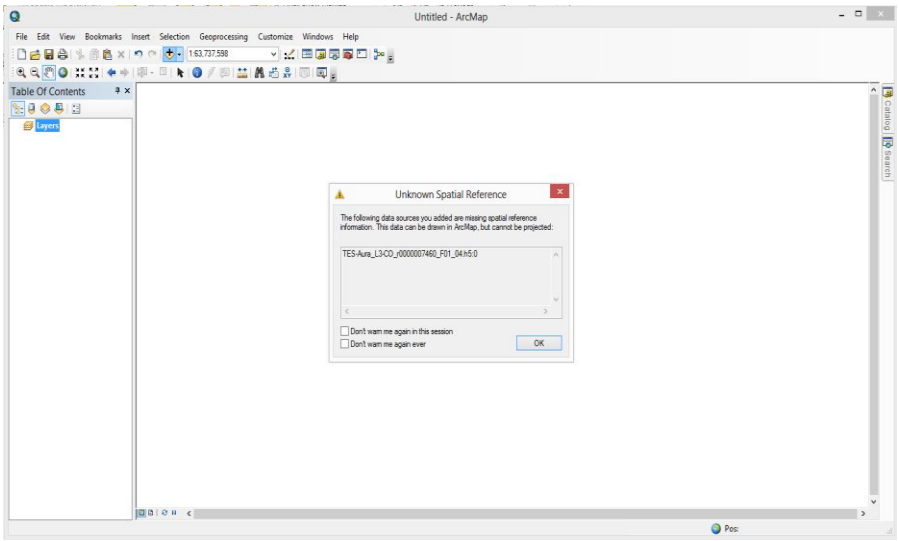


Figure 4. Missing georeference information

- Problem 3: 3D subsets cannot be interpreted correctly

Most GIS tools are unable to correctly display 3D or 3D+ sub-datasets of EOS data. There are several factors that may cause GIS to be unable to appropriately display 3D HDF datasets including 1) In HDF data, the dimension size is written in a wrong order. Take MOPPIT level 3 as example, one subset should have 9 bands and each band

is in 360*180 (width*height). However, the data record the dimension information as 360*180*9, instead, which means 360 180*9 (width*height) images. That's why that GIS often displays a very narrow rectangle image with gray color. 2) Sometime GDAL returns null value of the dimension size, because the field in metadata that GDAL reads is "band", but the field in HDF file store those information is actually "nprs", "nlon", and "nlat". 3) HDF driver does not support reading 5D subsets. For 4D subsets, driver cannot recognize the size of each dimension correctly. 4) Another reason is from GIS tools. Currently, the mainstream GIS systems do not support displaying multiple-dimensional HDF datasets yet, like 4D and 5D. Some of them use the first three bands as color channels (R, G, B) (Figure 5). In addition, as the third party software used in GIS, GDAL has not supported 4D and 5D datasets.

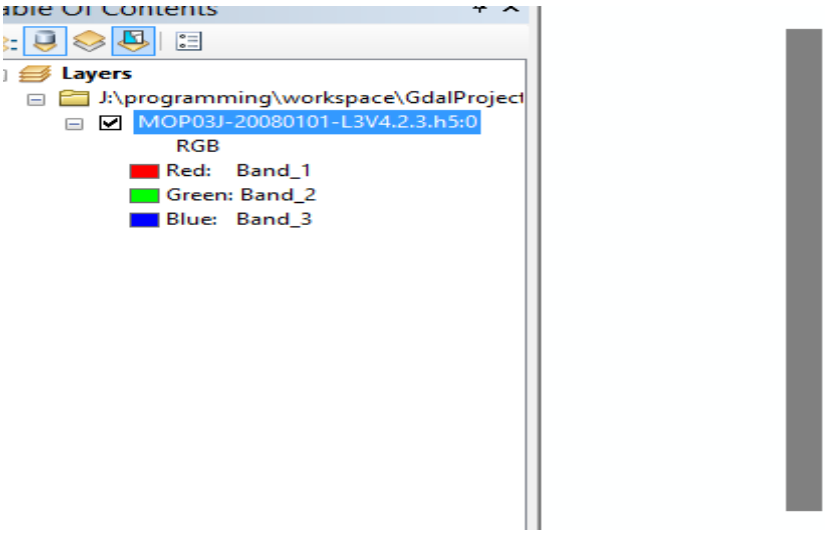


Figure 5. 3D subsets of HDF dataset in a GIS

- Problem 4: Missing metadata

Missing some useful metadata (e.g. NoData value, a mask used for representing the absence of data) is more likely to produce inappropriate image or lead to crash without a warning when displaying HDF datasets in GIS systems (Figure 6 and Figure 7). There are two aspects leading to the problem. On one hand, it is indeed that the metadata is not filled in at the initial stage. On the other hand, the metadata exist in the physical data file, but GIS or GDAL fails to interpret or find probably.

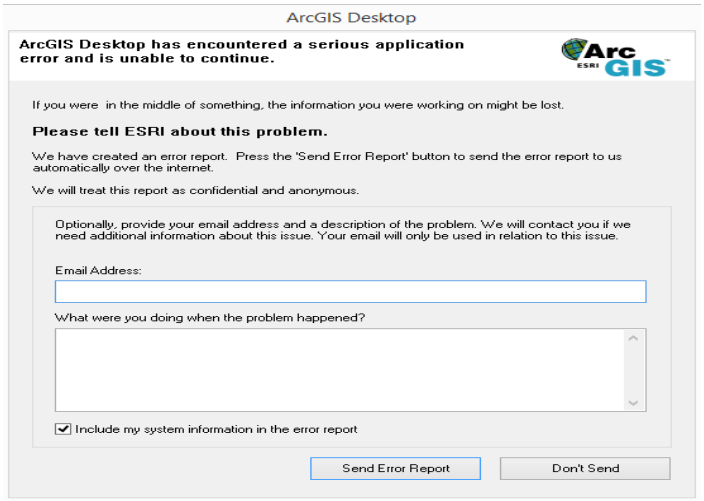


Figure 6. GIS crashes without a warning

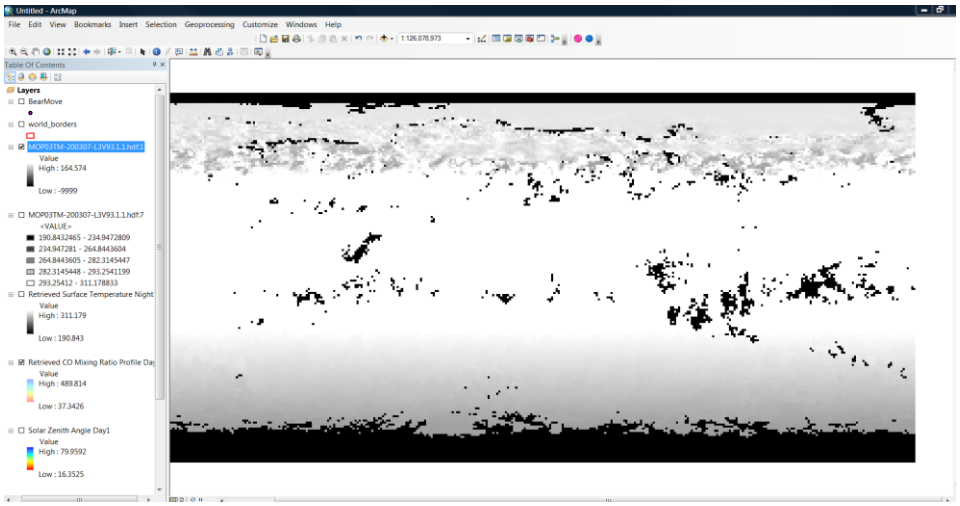


Figure 7. Displaying image without NoData value

CHAPTER 3 A GENERIC XML-BASED PLUG-IN FRAMEWORK

In order to fill the gap between EOS data and GIS software, this study presents a generic xml-based plug-in framework to facilitate the consumption of EOS data in GIS applications. The proposed framework allows user to detect the existing problems, intelligently repairs these problems, effectively interpret the variables and bands of EOS dataset, and finally correctly use images in GIS. Through studying the existed non-compliant data, we develop a number of functions to fix the data access problems appearing in GIS. It is essential to develop more functions to solve the increasingly emerging unknown problems, and support more types of EOS data products with similar problems. Hence, the framework should be designed for being easily integrated into both commercial GIS and open source GIS as an extension or a plug-in. Figure 8 is the overview of the plug-in framework that is composed of four primary component layers. The following section gives a detailed explanation about them one by one..

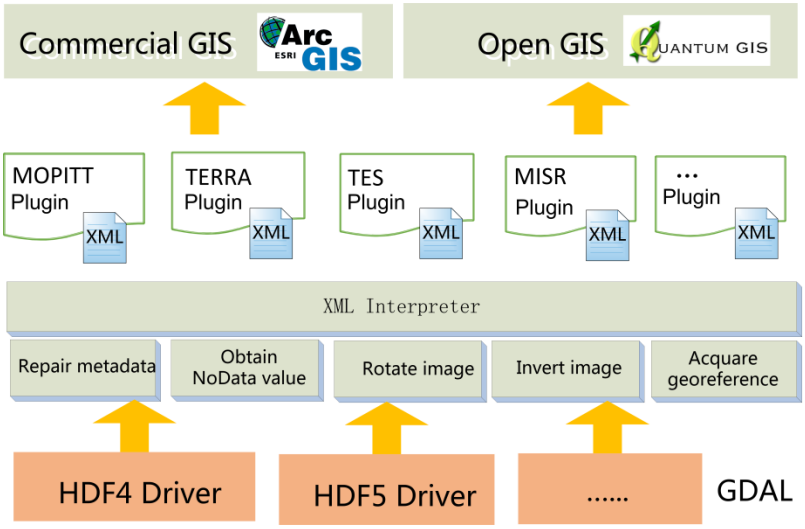


Figure 8. A generic xml-based plugin framework

3.1 GDAL/HDF Layer

Most EOS data products from NASA missions are stored in two formats: HDF and NetCDF. The former was invented by HDF group, while the latter was introduced by NetCDF group. The two data formats are also widely used in scientific communities. In the research, we aim to solve problems caused by HDF data in GIS, thus focus on HDF drivers in GDAL. This layer is placed at the low level of the framework but plays a critical role in accessing EOS data, since most GIS tools use GDAL as the third part software to deal with raster and vector dataset, and its HDF data drivers are specifically designed and developed to handle various sets of HDF data. In this layer, we can address most technique problems met at this current stage through enhancing the exiting HDF data driver, e.g. rotated or inverted image, and wrongly interpret multi-dimensional variables. Therefore, we made some changes on the source code of HDF4 and HDF5 drivers to enhance their capability of parsing HDF dataset, especially for multi-dimensional dataset.

3.2 Function layer

In order to address the various problems occurred in GIS applications when opening EOS data, we design a series of functions in the function layer that is on the top of GDAL layer. Generally, one function has been corresponded to one problem. For example, it is observed that the output 2D image from MOPPIT level 3 (HDF4) data is rotated by 90 degree. Therefore, we design Rotate_90_degrees function for rotating one 2D image as input by 90 degrees. Each function may be repeatedly applied on multiple data products that have the same problem. At this stage, a number of functions have been developed including: Rotate90Degree, InvertUpsideDown, OpenXmlFile, Get3DDimension, Get4DDimension, Get5DDimension, GetNoDataValue, GetGeoreference, and so on. With new problems from more data products, more new functions can be developed to address the problems. In the study, we categorize all functions into four groups according to their functionality, including metadata functions, image functions, interpreting functions, and georeference functions.

3.2.1 Metadata functions

Metadata is a type of data describing data itself and provides some basic information about the data. Incomplete metadata or missing metadata is more likely to lead to some unexpected results. For example, GIS probably fails to open data files, if it cannot recognize data type. To avoid the similar problem, a series of metadata functions are developed for retrieving useful user-defined information from metadata, including recognizing data type of image pixel value, getting image dimension, obtaining NoData values, or acquiring the time period of producing the data.

3.2.2 Image displaying functions

Occasionally, GIS displays EOS data as raster image inappropriately, due to the wrong way of organizing the pixel value. The investigation of all test EOS data products revealed the two most popular the problems: the image is rotated by 90 degrees or it is inverted upside down. Such problems probably were caused by the developers who writing the image into the HDF files. Therefore, we develop two corresponding functions to fix them: Rotate90Degree and InvertUpsideDown.

3.2.3 Interpreting functions

Most EOS data products often store a number of multiple dimensional variables that are comprised of more than three bands. Nevertheless, many GIS systems fail to process such datasets very well and display it inappropriately at most times, due to the third party library, GDAL. It is unable to fully support these data. For example, when opening a MOPPIT level 3 data that has one variable with 9 raster bands, e.g. temperature, ArcGIS treats the first three raster bands as color bands (RGB) and ignores other six raster bands. It would give scientists a misunderstanding of the dataset. The potential solution to this problem is to develop related functions that are targeted for correctly interpreting multiple dimensional EOS datasets, including 3D, 4D, and 5D.

3.2.4 Spatial reference functions

Different from traditional scientific data, EOS data are featured with spatial reference that helps users to position the geo-location of data. GIS often uses it to map different layers together from different data sources, make a thematic map. Without spatial reference information, GIS is unable to integrate, combine or map EOS raster

datasets in a united geographical coordinate system. Examining the output EOS data in GIS, we found that reference information is easy to lose, even though it exists in them. In order to better process EOS datasets in GIS, spatial reference function is developed to acquire spatial reference of each EOS raster dataset.

3.3 Plug-in layer

The Plug-in layer is the most important layer of the generic framework. It is designed as a bridge connecting function layer and GIS extension layer. When accessing and displaying EOS data in GIS, each type of data product may have one more problems listed in section 2.3. To solve these problems, the plug-in may will call their corresponding types of functions in function layer presented in section 3.2. The plugin help user assign the repairing functions that data products need, according to its problems in GIS. In this layer, we define each data product as a plugin associated with an xml file that defines a series of required functions. For example, the xml of MOPPIT data plugin includes three functions: rotate 90 degree, interpret multiple dimensional datasets, and obtain NoData value. Through parsing the xml, GIS enables to call the user-defined functions to fix the problems of corresponding data product files in GIS. Since EOS data provider is very familiar with the data model, they ought to create the xml plug-ins for EOS data user. The plug-in mode makes it easy for user to add one new data product into the GIS applications by adding a plug-in, rather than extra development, if the existing functions meet the demands.

3.4 GIS extension layer

The top layer in the generic framework is GIS extension layer that directly serves to the end user who intends to use EOS data in GIS. The layer is developed on the basis of the plug-in layer comprised of a variety of EOS data plug-ins. GIS extension includes specialized GIS tools for enhanced productivity and advanced analysis. Many GIS software offer a wide range of optional extension that can dramatically expand the capabilities of GIS, e.g. ArcGIS and QGIS (ESRI 2015). The GIS extension developed in the study better supports raster data in the forms of HDF files that is the most common format for storing EOS data. Different from GIS's build-in functions of loading HDF data format, the developed GIS extension enables to appropriately display EOS data associated with an xml file as plug-in. In the research, we customized two major GIS extensions based on enhanced GDAL to support HDF data, including ArcGIS and QGIS, the two most popular GIS systems.

CHAPTER 4 IMPLEMENTATION

Chapter 3 gives an overview of the generic xml-base plugin framework. This chapter introduces the developing environment and illustrates how to implement each component of the framework.

4.1 Development Environment

Taking into account that the framework is expected to be applied in both commercial and open GIS running cross-platforms, it is necessary to consider development compatibility and extensibility. Since GDAL is an open source software running on cross-platform, we adopt the standard programming API rather than Operation System (OS) level. The whole framework is implemented by using C++ programming language in Microsoft Studio environment on the basis of the optimized GDAL mentioned in section 4.1, and thereby easy to be translated into other programming language like Java. GIS users can utilize the interfaces to build their own extensions or directly integrate it into their own GIS system. The ArcGIS and QGIS extensions are developed by using C# programming language with Microsoft Studio on the basis of ArcGIS developing toolkit and the presented library above. It can be installed on machines equipped with ArcGIS software.

4.2 GDAL optimization

Section 2.3 describes the technical problems between EOS and GIS, some of them are more likely to be caused by the GIS's third party library, GDAL, e.g. missing NoData value. In order to overcome this kind of problems, we made changes on GDAL's source code on the basis of its the latest version (2.0.0) and compiled it again based on the official released HDF4 (4.2.6) and HDF5 (1.8.7) libraries from HDF group. Optimizing HDF driver of GDAL can greatly enhance the GDAL capability of accessing HDF4/HDF5 data, including gaining correct dimension information of HDF variables and bands and extracting geo-reference information. In addition, we also enhance the performance of opening and closing HDF datasets by fixing some GDAL internal defects or bugs. For example, it is observed that the procedure of freeing sub-dataset is time consuming, due to releasing a large number of Ground Control Points (GCPs) of each dataset in an unreasonable way. In comparison to the old version, the new GDAL's HDF driver is more stable and powerful in accessing of HDF data.

4.3 Function implementation

Section 3.2 introduces four groups of functions that are usually used for fixing EOS data problems in GIS applications. Targeting image function and interpreting functions, we developed a set of algorithms for the functions. Two cases are presented in this section, including display raster image probably and interpret 3D dataset correctly.

4.3.1 Displaying raster image

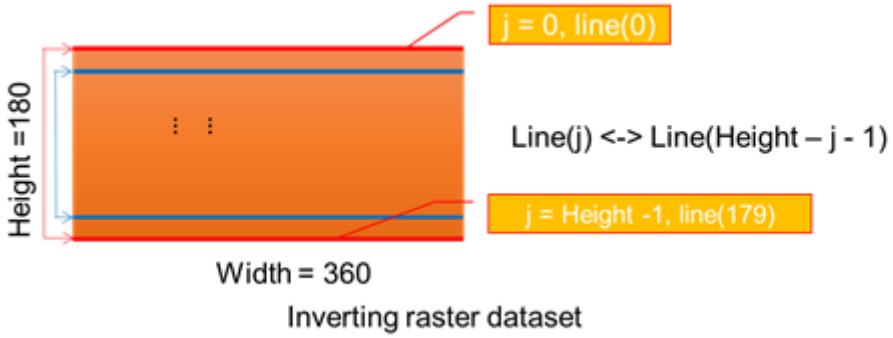
In the section 2.3, EOS raster image sometimes is inverted upside down or rotated by 90 degree. The former's solution is straightforward: exchange line (j) with line (height - j - 1) where j is the line number and height is the height of the raster image.

- Algorithm 1. presents the details of fixing image upside down problem as following:

```
Algorithm 1:
Requiring: width, height, buffer
1:for(int i = 0; i < width; i++)
2:      for(int j = 0; j < height / 2; j++)
3:              int line = height - j - 1;
                //exchange line(j) and line(height – j - 1)
4:              temp = buffer[j * width + i]
5:              buffer[j * width + i] = buffer[line * width + i];
6:              buffer[line * width + i] = temp
7:      end
8:end
```

- Algorithm 2. presents the method of fixing 90 degree rotation as following

```
Algorithm 2:
Requiring: old_width, old_height, old_buffer
Output: width, height, buffer
1:width = old_height;
2:height = old_width;
3:Create buffer with length (width * height * sizeof(DataType))
4:for(int i = 0; i < height; i++)
5:      for(int j = 0; j < width; j++)
6:              int idx_old = j * height + i;
7:              int idx_new = (height - i - 1) * width + j;
8:              buffer(idx_new) = old_buffer(idx_old);
9:      end
10:end
```



Inverting raster dataset

Figure 9. Invert raster image

4.3.2 Interpreting multi-dimensional image

Interpreting 3D dataset is one of the essential functions in the function layer. Almost all EOS data from ASDC has similar problems in GIS applications that treat the first three bands as color channel (RGB) and ignore other bands. Figure 10 shows the process of interpreting a 3D variable with multiple bands. It is observed that the variable is composed of 360 bands with width (9) and height (180). Actually, the variable is supposed to be represented as 9 bands which are featured with 360 width and 180 height. Hence, we develop algorithm 3 to re-organize the dataset structure.

- Algorithm 3. presents the method of correctly interpret 3D variables (Figure 10):

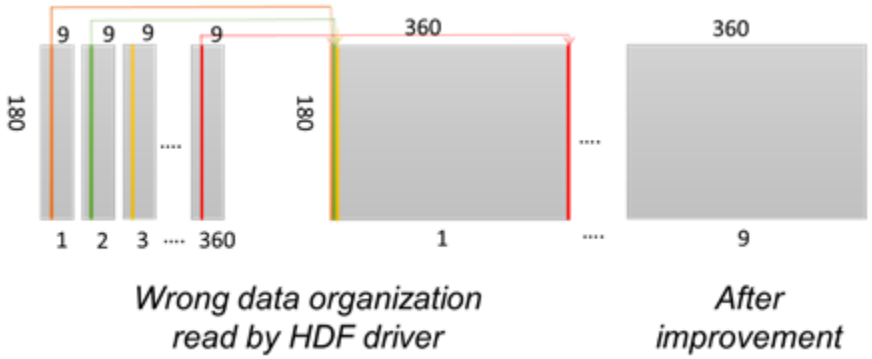


Figure 10. Re-organizing 3D dataset

Algorithm 3:
Requiring: rasterSizeX, rasterSizeY, rasterCount, bandIndex.
1:int pixelIndex = 0;
2:int nXBlocks, nYBlocks, nXBlockSize, nYBlockSize;
3:for(int r = 1; r <= rasterCount; r++)
4: Band band = dataset.getRasterBand(r);
5: band.GetBlockSize(nXBlockSize, nYBlockSize);

```
6:
7:      nXBlocks = (band.XSize + nXBlockSize - 1) / nXBlockSize;
8:      nYBlocks = (band.YSize + nYBlockSize - 1) / nYBlockSize;
9:      for(int iYBlock = 0 to nYBlocks)
10:             for(int iXBlock = 0 to nXBlocks)
11:                     create temp buffer[nXBlockSize * nYBlockSize];
                        band.ReadRaster(iXBlocks * nXBlockSize, iYBlocks * nYBlockSize, temp buffer,
                nXBlockSize, nYBlockSize, 0, 0);
12:                     for (int sy = 0; sy < nYBlockSize; sy++)
13:                             for (int sx = 0; sx < nXBlockSize; sx++)
14:                                     if (sx == bandIndex)
15:                                             int sidx = sy * nXBlockSize + sx;
16:                                             buffer[pixelIndex ] = temp buffer[sidx];
17:                                             pixelIndex ++;
18:                                     end
19:                             end
20:                     end
21:             end
22:     end
23:end
```

4.4 XML-based plug-in

As mentioned above, we developed a number of functions to address different problems occurring when using EOS data in GIS. Since different EOS data product has different usage problems in GIS, it is hard to organize and manage functions to repair these specific problems. One data product may need fewer functions to fix their problems. Therefore, the proposed strategy is a good solution to address the dilemma by using XML-base plug-in enables to intelligently choosing appropriate functions for EOS data product. The plug-in xml files store the type and format of EOS data product, and the names of required functions needed. An xml parser is specially designed for reading the information from this kind of xml files. Thereby, GIS enables to exactly call the pre-defined functions to fix the issues of the selected data product. Table 2 shows the content of the xml and the constraints of defining the xml file, which help user understand the xml. An example xml of configuring functions:

```
<?xml version="1.0" encoding="utf-8"?>
<ProductPlugin productType="MOPPIT3" productFormat="HDF4">
   <DataAccessProblem>Description Problems</DataAccessProblem>
   <Correction>
         <Function>InvertImageUpsideDown</Function>
   </Correction>
   <Correction>
         <Function>AddFilledValue</Function>
   </Correction>
   <Correction>
         <Function>Display3Dimension</Function>
         <RasterXDim>2</RasterXDim>
         <RasterYDim>1</RasterYDim>
         <RasterBandDim>0</RasterBandDim>
   </Correction>
   <Correction>
         <Function>Display4Dimension</Function>
         <RasterXDim>3</RasterXDim>
         <RasterYDim>2</RasterYDim>
         <RasterBandDim>1</RasterBandDim>
         <Raster4Dim >0</Raster4Dim >
   </Correction>
   <Correction>
         <Function>Display5Dimension</Function>
         <RasterXDim>4</RasterXDim>
         <RasterYDim>3</RasterYDim>
         <RasterBandDim>2</RasterBandDim>
         <Raster4Dim >1</Raster4Dim >
         <Raster5Dim >0</Raster5Dim >
</Correction>
</ProductPlugin>
```

Table 2. The content of the xml and the constraints of defining the xml file

Tag	Attribute	Required	Purpose	Allowed Value(s) as of date Oct 22, 2015
Product Plugin		yes	The root tag	N/A
	Product Type	yes	indicate the HDF data product	MOPPIT, TES, MISR or FLASH
	Product Format	yes	Defines the HDF version	HDF4 or HDF5
Problem		no	Describe the problems of the	e.g. All sub datasets of this HDF files are

			corresponding HDF data to give a good understanding of the problems to users	inverted upside down. Meanwhile, it does not support multiple-dimensions (e.g. 4D/5D)
Correction		yes	indicate the correction that will be applied to the data product	
Function		yes	Indicate the function that needs to be be invoked in the GDAL, Usually, there are more than one Function tag. The number of the Function tag in the xml depends on the number of problems the data have	InvertImageUpsideDown: Invert the image upside down. RotateImageBy90Degree: Rotate the image by 90 degrees clockwise. AddNoData: Obtain NoData value from metadata or set it manually. LoadMissingGeoreferenc e: read georeference data from Display3Dimension: Support 3D subdataset. Display4Dimension: Support 4D subdataset Display5Dimension: Support 5D subdataset
NoData		optional	add this tag when the data need the function "AddFilledValue", the missing value will be set as the input value. If no tag, the program will search missing	a missing data value, e.g. -9999

			value in data array and fill up the missing value automatically	
RasterXDim		optional	give the flexibility to users to define the sequence of dimensions, only appear along with the Display3/4/5Dimension correction functions	default value is 2 when handling 3D subset
RasterYDim		optional	give the flexibility to users to define the sequence of dimensions, only appear along with the Display3/4/5Dimension correction functions	default value is 1 when handling 3D subset
Raster3Dim		optional	give the flexibility to users to define the sequence of dimensions, only appear along with the Display3/4/5Dimension correction functions	default value is 0 when handling 3D subset
Raster4Dim		optional	give the flexibility to users to define the sequence of dimensions, only appear along with the Display4/5Dimension correction functions	

Raster5Dim		optional	give the flexibility to users to define the sequence of dimensions, only appear along with the Display5Dimension correction functions	

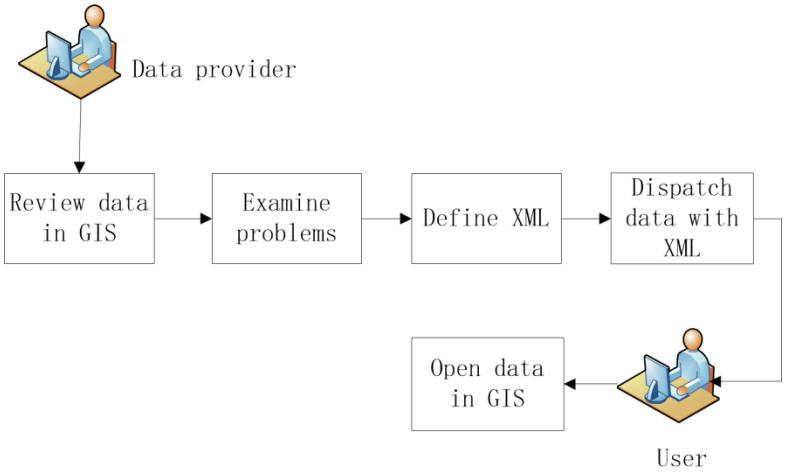


Figure 11. Plugin information flow

Figure 11 is the information flow of generating a plugin xml file for one data product. First, a data provider need to reviews EOS data files and examines the specific problems through GIS software. Due to these problems, they would choose the specific functions designed for solving the problems and write the names of pre-defined and implemented functions into the plugin xml file. It should be dispatched with HDF data files and put together for enabling EOS data user to successfully open them in GIS applications. If EOS data users have a good understanding of the data product they are

using, they can also define the xml file by themselves, according to the xml instruction document.

4.5 Workflow of processing HDF dataset

Figure 12 is the workflow of processing an EOS HDF dataset that has one or more multiple dimensional variables by using ArcGIS extension. First, the EOS data product's type and format can be established by the selected file's full name or its metadata. Meanwhile, system will automatically scan all files under the current working folder where the selected data file locates, and search the associated plugin xml file. If it is available, parsing the xml file enables to get a series of predefined functions used to solve the problems of the current data product. In processing the data in ArcGIS, these functions work on all raster bands of the selection before displaying these raster images. The customized graphic user interface (GUI) of the developed extension allows users to freely choose the variables and their sub bands of. Finally, ArcGIS would display raster image layers generated from the selected raster bands by using ArcGIS APIs.

Figure 13 is the Graphic User Interface (GUI) of selecting HDF sub-datasets when opening HDF data file in ArcGIS. It enables to recognize HDF data product type and format from the plugin xml file dispatched with the type of data product, and retrieve the user-predefined functions that are shared by all variables and bands of the data. The newly designed GUI allows user to conveniently select multiple bands from the same or different variables for integrating them into a same workspace and creating multiple corresponding raster image layers at the same time.

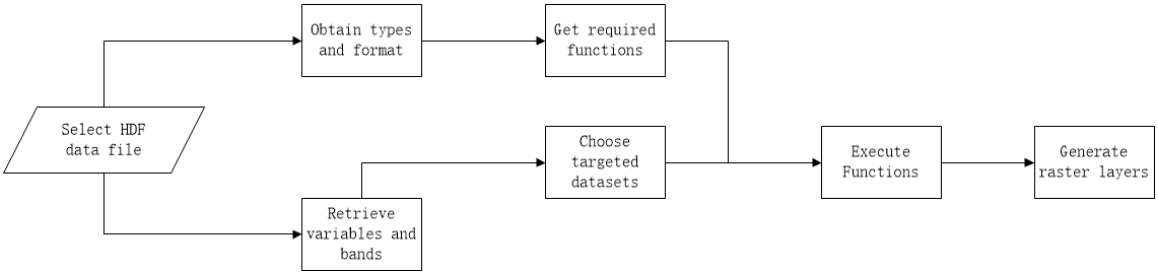


Figure 12. Processing workflow

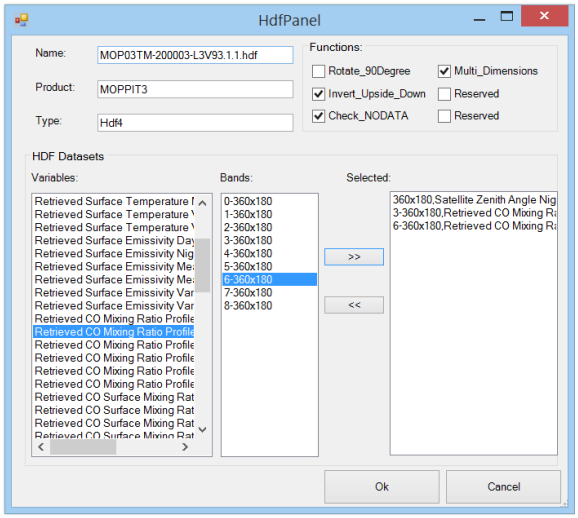


Figure 13. Extension GUI of selecting HDF data

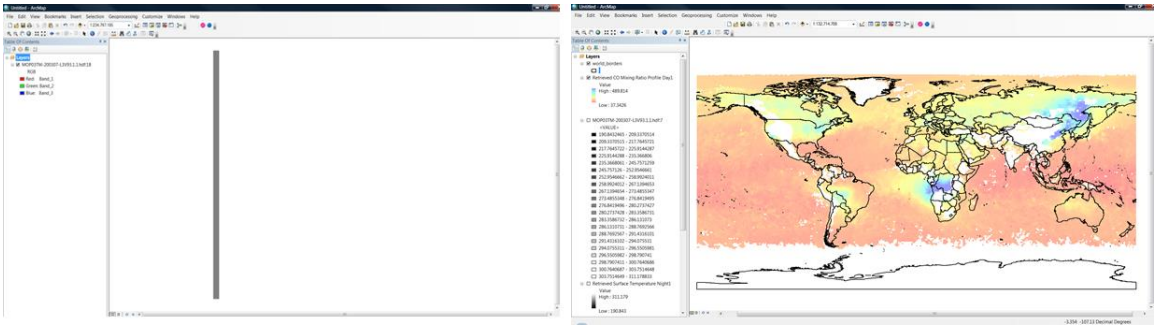
CHAPTER 5 EXPERIMENT & RESULT

To validate the feasibility and functionality of the proposed plug-in framework, we developed an ArcGIS extension to test a diversity of EOS data products that have been collected from NASA's ASDC. In the study, a set of experiments were conducted in ArcGIS (10.2) to demonstrate the availability of the developed extension, according to the problems presented in the section 2.3. To conveniently compare the difference before and after the improvements, some GIS data were included and assembled onto the generated raster layer, e.g. world boundary shape files.

5.1 Case 1: Interpret multiple dimension HDF dataset

When opening EOS data in ArcGIS, the most serious problem is that GIS cannot interpret and display 3D/4D/5D HDF sub-datasets probably. These types of problems occur to most EOS data products. In this experiment, we use a MOPPIT HDF data file that has a 3D sub-dataset ("Retrieved CO Mixing Ratio Profile Day") as the testing sample data to show the improvements of the extension developed. Figure 14.A shows the displaying result of a selected 3D sub-dataset that is generated in primitive ArcGIS. It is observed that the image is abnormally displayed and its bands are wrongly treated as color band (RGB). Therefore, we developed the functions of interpreting and displaying multiple dimensional HDF datasets to solve the problem. Figure 14.B shows the appropriate displaying of an image that is generated from the first band of the selected 3D

HDF sub-dataset after the improvements. The extension has greatly enhanced the capability of ArcGIS in accessing and processing multiple dimensional HDF datasets.

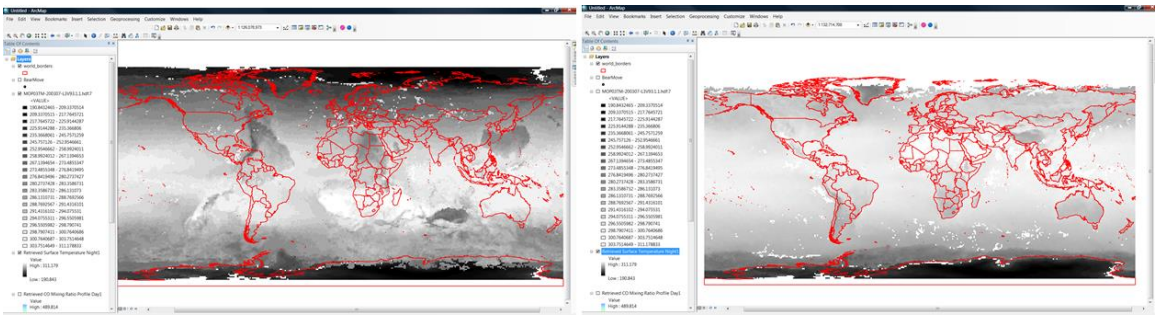


A B

Figure 14. Interpret 3D variables

5.2 Case 2: Rectify image inverted problem

When opening some EOS HDF4 data in ArcGIS, e.g. MOPPIT data, it is found that the raster image generated from one of its 2D sub-dataset was inverted upside down. The problem occurs to EOS data products in HDF4. For example, we use "MOP03TM-200307-L3V93.1.1.hdf " as the sample data to show the problem. Figure 15.A shows the displaying result of one selected sub-dataset, "Retrieved Surface Temprature Night" in ArcGIS. Comparing with the overlay of world map boundary, it is clear to see that the raster image is inverted while displaying in ArcGIS. Using the image functions that we implemented enables these issues to be addressed. Figure 15.B shows the correct raster image through opening it in the developed ArcGIS extension after improvements.

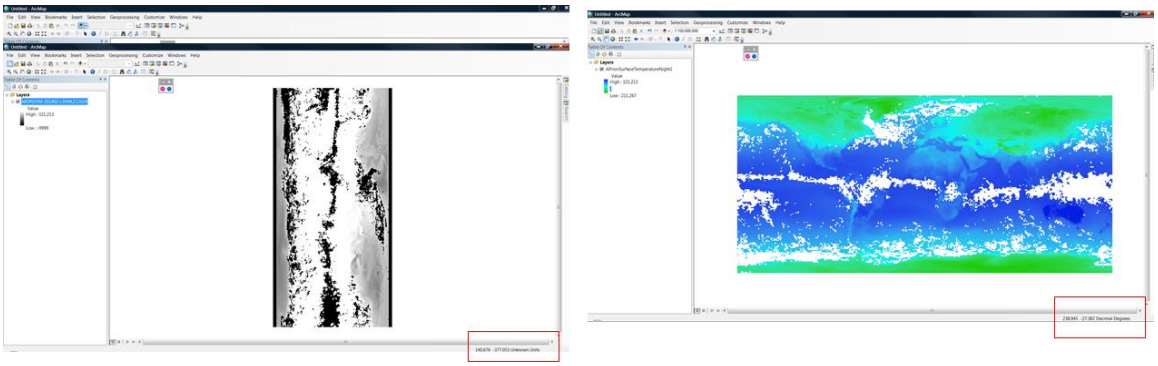


A B

Figure 15. Fix the problem of inverted image

5.3 Case 3: Repair images rotated by 90 degrees

The problem that raster image is rotated by 90 degree in clockwise direction usually occurs to EOS HDF 5 data files when opening in GIS applications. In order to confirm the functions of repairing the problem, we use MOPPIT HDF5 data (MOP03TM-201402-L3V94.2.1.he5) as the sample data and open in ArcGIS. Figure 16.A shows the result of one selected 2D sub-dataset ("A Priori Surface Temperature Night") of the selected HDF data file in ArcGIS without any improvement. The whole raster image is inappropriately displayed as rotated 90 degrees. To solve this kind of problem, it is necessary to call image functions to process the data. Figure 16.B is the correct displaying result of the same sub-dataset through the ArcGIS extension with improvements.

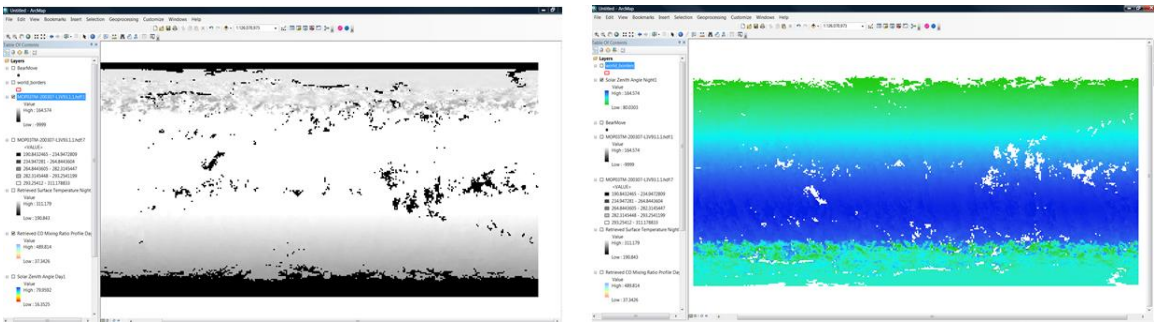


A B

Figure 16. Fixing 90 Degree rotated

5.4 Case 4: Assign missed NoData value

Sometimes, developers and users who created HDF data at the first time overlooked putting NoData value into HDF metadata or retrieve it from HDF data file, which leads to that GIS are unable to properly display raster images from HDF datasets. It would make it hard to understand the displaying results, while missing NoData value in GIS. We use MOPPIT HDF data (MOP03TM-201402-L3V94.2.1.he5) as the sample data to show the importance of setting NoData value. Figure 17.A shows the generated raster image without assigning NoData value in ArcGIS, which poses a risk of interpreting it in a wrong way. From the figure, it is hard to clearly find the distribution of the feature. On the contrary, Figure 17.B is the correct output image that gives us a clear distribution map of the targeted feature, if using NoData value as -9999.



A B

Figure 17. Color scale including No-data value -9999

5.5 Case 5: Adjust image mismatched coordinate system

Spatial reference defines a specific map projection of EOS data. However, when opening EOS HDF data in GIS, its spatial reference is missed, which leads to that the coordinate system is mismatched between two data sources, e.g. EOS data and GIS data. In this study, we use MOPPIT data as sample data to investigate the problem. We select two bands of one sub-dataset “Solar Zenith Angle Day”, but find the two raster image cannot be overlapped in ArcGIS. Figure 18 shows that two mismatched raster images that are supposed to have the same spatial reference.

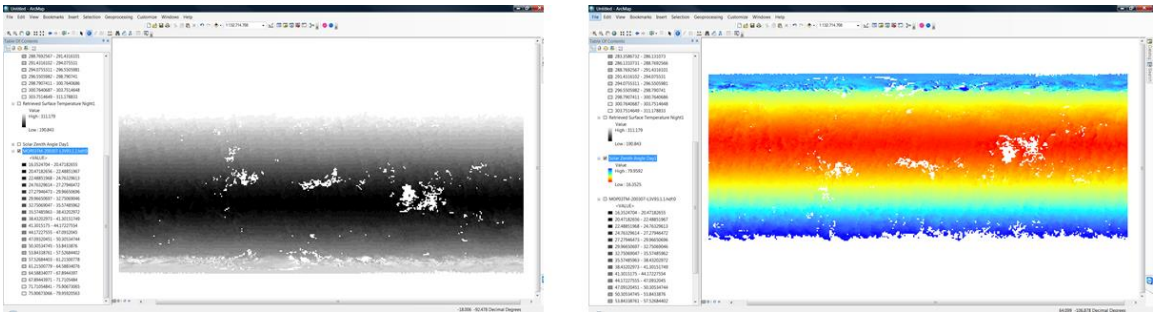


Figure 18. Coordinate system mismatch

CHAPTER 6 CONTRIBUTION TO GDAL COMMUNITY

The initial purpose of developing such a generic framework is to allow GIS developers to integrate it into GIS applications for enhancing their capabilities of processing EOS data, especially for these data having problems when they are used in GIS. Ever since, more and more people would like to process these EOS data by using GIS tools. On the other hand, it is equally important to those users who would like to directly use GDAL to process the EOS data with problems, e.g. transferring the data into different image formats. Therefore, capsulizing the work into GDAL source code is significant to expanding the usage of the EOS data. In order to contribute the framework to GDAL, it is necessary to put the implementation into GDAL at a source code level. Figure 19 shows the primary method in which functions are implemented in HDF4 and HDF5 data drivers in GDAL.

From Figure 19, the plug-in framework is incorporated into the source code of original GDAL, specifically in HDF4 and HDF5 data drivers. Within the framework, there are five major components for overcoming the problems that are introduced above. During the implementation, five general components into four classes: HDF4ImageDataset, HDF4RasterBand, HDF5ImageDataset, and HDF5RasterBand. Different from the conventional method of opening a HDF data file, the new GDAL edition requires an xml file to be related with the targeted HDF data file that need to be

fixed. As a matter of fact, the xml file determines the functions to be invoked when opening the HDF data file using GDAL.

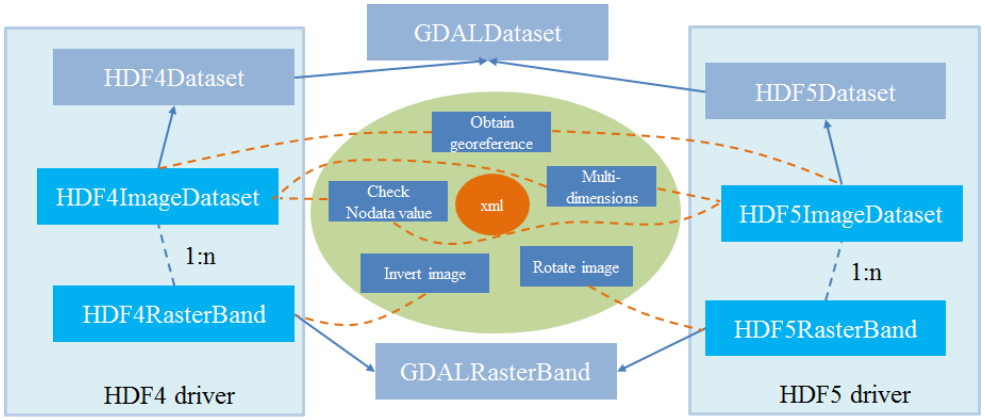


Figure 19. Plug-in framework incorporated into GDAL source code

Therefore, there are two ways to implement the framework. One is to develop a middleware on the basis of GDAL without changing its source code. GIS developer can integrate the library into their applications or produce an extension based on the library. Another is to capsulate the framework into GDAL source code, which most of effort have been done in GDAL. Both are effective solutions but contribute to different communities. Table 3 shows the advantages and disadvantages between independent library and enhanced GDAL respectively.

Table 3. Compare Middleware and enhanced GDAL

	Middleware	Enhanced GDAL
Pros	Development: Flexible in development, not limited by	Development: May be limited by GDAL interface structure when expanding any new function.

	GDAL interface Easy to expand interfaces Integration with GIS software and GDAL widgets: no obvious advantage compared to the other approach Contribution and maintenance: Because modification outside GDAL, maintenance is relatively independent from GDAL versions. This makes maintenance easier.	GDAL has been compiled on different platforms (Linux, Windows, or Mac). Integration with GIS software and GDAL widgets: be able to use GDAL widgets (e.g. gdal_translate) Contribution and maintenance: Contributing to GDAL community is more straightforward
Cons	Development: Need more effort to compile cross-platform versions Integration with GIS software and GDAL widgets: Need development from the GIS software side. Unable to use GDAL commands Require new understanding of the framework for GIS software developers Contribution and maintenance: no obvious disadvantage compared to the other approach	Development: Hard to add new interfaces or public functions, constrained by GDAL code structure. Hard to expand HDF data driver's capabilities. Integration with GIS software and GDAL widgets: may still need modification from GIS software side (probably not only simply replace GDAL library with the new one). Contribution and maintenance: the major change on GDAL is in HDF driver. If the official new release does not make too much change in HDF driver, then update the HDF driver with our changes should be easy. Otherwise, we need to redevelop the HDF driver.

CHAPTER 7 CONCLUSION AND FUTURE WORK

As an integral part of a NASA project, the research presented a significant solution of integrating EOS data into GIS. It mainly focuses on upgrading the delivery of ASDC data products for consumption by GIS applications. The proposed XML-based plug-in framework enables GIS application to efficiently and effectively access and interpret HDF-EOS data, especially for multiple dimensional datasets. In order to enhance GDAL's capability in processing EOS data formats, we optimized GDAL source code in interpreting multi-dimensional variables and overcoming the limitation of accessing complicated HDF datasets. Based on the enhanced GDAL and the generic plug-in framework, we developed GIS extensions cross-platforms to facilitate GIS users to better use EOS data in GIS, especially from ASDC. A series of conducted experiments demonstrated that ArcGIS extension enables to solve a series of practical problems when integrating these datasets in GIS applications, especially for multiple dimensional datasets. Furthermore, all work has been capsulated in GDAL source code and contributed into the GDAL community, which tremendously encourages both researchers and developers to make full use of EOS data. We also expect that the methodology in the research can be applied into other domains. The next research plan is to test more EOS data products from NASA's other data centers, and make the tool become more elastic, scalable, and robust in manipulating and interpreting EOS data.

APPENDIX

REFERENCES

Bagwell R, Lindsay F, Lynnes C, and Yang M 2011 Using NASA Remote Sensing Data in a Geographical Information System. AGU Fall Meeting Abstract, 1: 11423.

Board S 2007 Earth Science and Applications from Space:: National Imperatives for the Next Decade and Beyond. National Academies Press.

Di L, and Kobler B 2000 NASA Standards for Earth Remote Sensing Data. INTERNATIONAL ARCHIVES OF PHOTOGRAMMETRY AND REMOTE. SENSING, 33(2): 147-155.

Di L, Yang W, Deng M, Deng D, and McDonald K 2002. Interoperable access of remote sensing data through NWGISS. In Proceedings of Geoscience and Remote Sensing Symposium, IEEE International, 1: 255-257.

ESRI 2015 ArcGIS online. WWW document, http://doc.arcgis.com/en/arcgis-online/create-maps/display-imagery.htm. [accessed 15 September 2015]

Farr T, Granger S, and Kopp S 2011 Integrating Remote Sensing Data Into Geographic Information Systems. Transactions American Geophysical Union, 92(18): 154-154.

Folk M, McGrath R, and Yeager N 1999 HDF: an update and future directions. In proceedings of Geoscience and Remote Sensing Symposium, IEEE 1999 International, 1: 273-275.

GDAL 2015 GDAL online document. WWW document, http://www.gdal.org/.[accessed Dec 05 2015]

Goodchild M 1994 Integrating GIS and remote sensing for vegetation analysis and modeling: methodological issues. Journal of Vegetation Science, 5(5):615-626.

HDF 2015 HDF-EOS tools and information center. WWW document, http://hdfeos.org/software/arcgis.php.[accessed 5 October 2015]

Hill L 2009 Georeferencing: The geographic associations of information. Mit Press.

Lo C, Quattrochi D, and Luvall J 1997 Application of high-resolution thermal infrared remote sensing and GIS to assess the urban heat island effect. International Journal of Remote Sensing, 18(2):287-304.

McDonald K 2003 Serving NASA EOS data to the GIS community through the OGC-standard based NWGISS system. In Proceedings of 2003 Asia GIS Conference. Asia GIS Society. Oct 16-18, Wuhan, China.

NSIDC 2015 National Snow& Ice Data Center. WWW document, http://nsidc.org/data/hdfeos/intro.html[accessed 10 August 2015]

Petropoulos G, Kalivas D, Griffiths, H, and Dimou P 2015 Remote sensing and GIS analysis for mapping spatio-temporal changes of erosion and deposition of two Mediterranean river deltas: The case of the Axios and Aliakmonas rivers, Greece. International Journal of Applied Earth Observation and Geoinformation, 35: 217-228.

Raskin R, Pan M, and Mattmann C 2004 Enabling semantic interoperability for earth science data. In proceedings of 4th NASA Earth Science Technology Conference.

Shalaby A, and Tateishi R 2007 Remote sensing and GIS for mapping and monitoring land cover and land-use changes in the Northwestern coastal zone of Egypt. Applied Geography, 27(1):28-41.

Tsou M 2004 Integrating Web-based GIS and image processing tools for environmental monitoring and natural resource management. Journal of Geographical Systems, 6(2):155-174.

van de Vegte J, de Cerff W, van den Oord G, Sluiter R, van der Neut I, Plieger M, and Wilhelmi O V 2007 Atmospheric data access for the geospatial user community.

In Remote Sensing. International Society for Optics and Photonics: 674910-674910.

Wilkinson G 1996. A review of current issues in the integration of GIS and remote sensing data. International Journal of Geographical Information Science, 10(1):85-101.

Wright D, and Goodchild M 1997. Data from the deep: implications for the GIS community. International Journal of Geographical Information Science, 11(6):523-528.

CURRICULUM VITAE

Yunfeng Jiang received his Bachelor of Science in Geographic Information Science from China University of Mining and Technology in 2004.