**Salient Feature Detection from Planetary Images**

(SFDPI)

**CS496 Senior Design**

**Project Documentation**

Document Prepared by:

Benjamin Evans

Kevin Casado

Adam Berman

Lineli Hairapetian

Bruno Recillas

Andy Tong

5 December 2015

CALIFORNIA STATE UNIVERSITY LOS ANGELES



Los Angeles, California

**Table of Contents**

[1.0 INTRODUCTION](#h.30j0zll)

[1.1 Purpose.........................................................................................](#h.1fob9te)

[1.2 Scope…………………………………………………………………..](#h.3znysh7)

[1.2.1 Relationship to Other Documents…………………………………](#h.2et92p0)

[1.2.2 Hardware and Software Considerations…………………....](#h.3dy6vkm)........

[1.3 Documentation of the Development Process………………..](#h.1t3h5sf).........

[1.4 References…………………………………………………………….](#h.4d34og8)

[1.4.1 Controlling Documents……………………………………………..](#h.2s8eyo1)

[1.4.2 Applicable Documents……………………………………………..](#h.17dp8vu)

[1.4.3 Standards……………………………………………………………](#h.3rdcrjn)

[2.0](#h.26in1rg) [Functional Description of the](#h.1ksv4uv) SFDPI

2.1 Project Origin ……………………………………………………….

2.2.1 Ellipse Fitting improvement …………………………………………

2.2.2 Circular Hough Transform Improvement …………………………..

[2.2.3 Template Matching Improvements ………………………………..](#h.35nkun2)

2.3. System Requirements

[3.0 SFDPI Setup procedures](#h.1ksv4uv)

# **1.0 INTRODUCTION**

# 1.1 Purpose

The purpose of this document is to define a full set of requirements for SFDPI. (These sections correspond to a Software Requirements Document, SRD). The complete definition of all SFDPI requirements provides the requirements to be used in the subsequent software subsystem documents

### 1.2 Scope

This is the documentation developed as part of this CS496A class. Full implementation will be completed in the following CS496B and CS496C classes.

The scope of this document includes the following:

* All functional requirements for SFDPI
* A detailed description of setup procedures for SFDPI

## 1.2.1 Relationship to Other Documents

The **SFDPI** SRD/SDD/STP/SID is a complete self-contained document.

## 1.2.2 Hardware and Software Considerations

The software being written requires the use of open source code. The software must also be executable in the Linux environment.

# 1.3 Documentation of the Development Process

Section 2 is a succinct software description document describing methods of improving implementation of the various crater detection algorithms used in SFDPI as well as the functional requirements for SFDPI.

Section 3 is a succinct software description document detailing with setup procedures for SFDPI

# 1.4 References

All references used in the creation of this document are listed below.

## 1.4.1 Controlling Documents

There is no document controlling this document.

## 1.4.2 Applicable Documents

SFDPI is an extension of LIPFD (Lunar Image Processing - Feature Detection), also referred to as LCDR (Lunar Crater Detection and Recognition), so all documents pertaining to LIPFD or LCDR are directly applicable to this documentation excluding certain functional requirements that will be discussed in Section 2.

## 1.4.3 Standards

No Standard has been used in the creation of this document.

# **2.0 FUNCTIONAL DESCRIPTION OF THE SFDPI**

# 2.1 Project Origin

SFDPI is an extension of LIPFD (also referred to as LCDR). The goal of LIPFD is to create an algorithm or system for detecting craters on the Lunar surface. The LIPFD team accomplished this goal through a pipeline called Ringtoss which was later implemented, and when tested with Lunar Images produced good results. SFDPI aims to provide many of the same functionalities as LIPFD but shall use Mars images as input instead of Lunar images. SFDPI’s main focus will be on crater detection. Diameters of detected craters shall be calculated and checked for accuracy. However, the accuracy of craters’ calculated depth are not paramount for this project, nor is rock detection so these two features will not be included as part of SFDPI.

LIPFD utilizes three different algorithms for detecting craters. Each Algorithm shall be tweaked for SFDPI to yield better results for Mars images.

# 2.2.1 Ellipse Fitting Improvements

The crater detection algorithm using the ellipse fitting method takes lunar images with varying sun angles that produce distinct highlights and shadows. The image is sent through a two-step process; pre-processing and ellipse fitting. The pre-processing stage begins with a lunar image and converted using an RGB to HSV filter. The resulting image is then fed through a thresholding function, converting the image into a binary image. The process continues on to an erosion and dilation phase, removing any background noise. The image now moves on to the centroid detection and matching of light and dark patches, as well as calculation of distance and angles, and saved as crater candidates. This candidate list is then sent on to the ellipse fitting method, where an ellipse is fit to each candidate. The final image, with fitted ellipses, is then displayed and the detected craters are stored in a list. When the input image is converted to the TIFF format by ISIS the image may be oriented in a way that doesn’t line up with the original sun angle. To account for this we must apply similar transformations to the sun angle to ensure that everything is properly aligned. Also due to the nature of this algorithm, images taken at lower elevations will produce better results. To improve the results from this algorithm it is in our best interest to limit detection for images no higher than a specific elevation that will be determined at a later date.

2.2.2 Circular Hough Transform Improvements

The crater detection algorithm using the circular Hough transform takes in a lunar image and detects the crisp new craters within the image. Before the image is fed into the CHT (Circular Hough Transform) some pre-processing steps must be accomplished. First the image is enhanced by applying Single Scale Retinex algorithm followed by Discrete Wavelet Transform algorithm to the image. The retinex DWT algorithms will reveal details normally hidden by shadows. Secondly, the image is passed through Gaussian and Median filters to reduce the noise within the image. Once pre-processing is done the image is ran under the Canny edge detector. The user adjusts a slider to determine the hysteresis lower and upper thresholding values to be used in the canny edge detector. The canny edge detector will output a binary image of edges that will be feed into the circular Hough transform. The circular Hough transform will determine the centers and radii of craters. This data is stored and the centers and radii are overlaid on the lunar image. This algorithm runs the Hough transform with different radii ranges and thresholding values. The results are unioned, stored, and displayed. Because of the nature of Mars surface and the images we get from Mars, Circle Hough algorithm does not produce good results. In order to get better results on Mars images, algorithm needs some improvements. Other than using different variables which will work better on Mars images, we need more technical improvements. One improvement idea is to detect where we have many small circles detected and check to see if they are in a large circle which did not get detected or the algorithm missed it because it is not completely circular. At this point we may need to detect some elliptical shapes.

2.2.3 Template Matching Improvements

In order to improve the results for the template matching algorithm we shall update the template images to include those of Mars images. These templates shall be selected based on the locations with confirmed Mars Craters that are within one kilometer of the MSL Landing Site .

# 2.2.3 Creating Test Cases and Producing Analysis Data

Test cases are produced by downloading CDR(Calibrated Data Record) IMG files of Hirise images from <http://hirise-pds.lpl.arizona.edu/PDS/EDR/>. The IMG files are converted TIFF using USGS ISIS commands. These TIFF images are used as input for Ringtoss. The algorithms will be tested across various images to check for accuracy of crater detection. Each algorithm in the Ringtoss pipeline will run the same images through to compare results and compare accuracy and performance between different methodologies used. Results regarding depth and diameter will be calculated but will not be the primary focus when determining accuracy of results; since SFDPI is primarily focused on improving Ringtoss’ ability to detect craters on Mars images the latitude and longitude of the detected crater shall be used to determine SFDPI’s accuracy.

2.3 System Requirements

SFDPI must use Ringtoss, as it is an extension of LIPFD. The System requirements for SFDPI include the following:

* Unix based operating system
* Java8
* GDAL with python bindings
* OpenCV 2.4.9 or 2.4.10
* MySQL 5.6 or greater
* Python 2.7 (for gdal)
* USGS ISIS

**3.0 SFDPI Setup Procedures**

Once the prerequisite software is installed (see section 2.3 ) the following steps must be taken to ensure that SFDPI runs properly.

1. Download **EDRCUMINDEX.TAB** from http://hirise-pds.lpl.arizona.edu/PDS/INDEX/
2. Open the SFDPI shell script and set the **cd\_metadata\_file** variable equal to the path that the TAB file is in.
3. Change set the **run\_create\_database** equal to “yes”
4. Run the script

Following the above procedure will create a database on the local machine that include metadata for Mars Hirise images. After the database is initialized be sure to switch the **run\_create\_database** parameter to “no”.

#### DATA DICTIONARY

None.

|  |  |
| --- | --- |
| B. | ACRONYMS |
|  |  |
| DEM | Digital Elevation Model |
| GDAL | Geospatial Data Abstraction Library |
| SFDPI | Salient Feature Detection from Planetary Images |
| IMG | Disk Image File |
| CDR | Calibrated Data Record |
| NAC | Narrow Angle Camera |
| LIPFD | Lunar Image Processing - Feature Detection |
| TIFF | Tagged Image File Format |
|  |  |

|  |  |
| --- | --- |
|  |  |