# Chiminey: 3D Insect Surface Fingerprints

Loy Larvin Rao, RMIT University

**Abstract**

The purpose of this project is to bring together three software packages, to create a toolkit that can be used by researchers to approximate roughness conditions of insect surfaces. The project was developed in collaboration with DATA 61, CSIRO. This report demonstrates how the components were modified and configured to produce a toolkit that can be used to analyse surfaces of different scales without any changes required in the tool. The final result was tested using open-source elevation data obtained from the Mars Global Surveyor, LIDAR elevation data of Tauranga and Coast, Bay of Plenty, New Zealand captured for BOPLASS Ltd by Aerial Surveys in January through April 2015, and LIDAR data of Victor Harbour, South Australia, captured for The Coastal Urban Digital Elevation Modelling in High Priority Regions (UDEM) project. Using The National eResearch Collaboration Tools and Resources project (Nectar), which provides cloud infrastructure, the toolkit was tested for scalability by spawning several virtual machines. The toolkit can serve as a valuable resource for etymologists who wish to fingerprint surfaces of insects for the purpose of classification. Also, test results suggest that the utility of the toolkit can be extended to other areas, such as in topology, to approximate roughness of very large geographic surfaces.

## Introduction

The aim of this project was to build a tool chain for CSIRO and citizen researchers to approximate roughness conditions from 3D insect data. The tool would allow researchers to select surface windows of interest through a browser interface, and determine the roughness of these surfaces. Such a tool could be used by entomologists for the purpose of fingerprinting and classification of insects. The entire toolkit was developed using cloud resources provided by The National eResearch Collaboration Tools and Resources project (NeCTAR), making it scalable. The toolkit was built by integrating three existing software packages.

### MyTardis: For curation of insect files and roughness results

MyTardis is an open-source data curation system, whose development began at Monash University to solve the problem of users needing to store large datasets and share them with collaborators online. Its particular focus was integrating with scientific instruments, instrument facilities and research storage and computing infrastructure, to address the challenges of data storage, data access, collaboration and data publication. It is currently being used to capture data from areas such as protein crystallography, neutron and X-ray scattering, optical microscopy, electron microscopy, medical imaging, etc.

### Chiminey: For high-throughput parallel analytics of insect data

Chiminey was developed by RMIT eResearch and AICAUSE and is a software platform that enables scientists to perform scalable computations on cloud-based and High Performance Computing (HPC) facilities. This system gives special importance to resource access and management abstraction. Chiminey provides definition, execution and monitoring of high-performance, big data, and cloud computing applications. It provides a user interface that focuses both on the domain-specific parts of a task for scientists and a framework that allows developers to build computation tasks. Some of the key features of Chiminey are

- Automatic generation of parameter sweeps over variables that can be scheduled on HPC clusters or across cloud IaaS nodes.

- Advanced fault tolerance framework. A smart connector at most recovers a failed execution, at least prevents the failed execution from causing a failure in the entire system
- Smart Connectors for data transfer to and from remote data sources and remote execution platforms for both Unix and cloud computation resources. A smart connector is the core concept within Chiminey that enables end users to perform complex computations on distributed computing facilities with minimal effort.

### Roughness Package: For analysing surfaces statistically

3D Surface Roughness Analysis Tool is a surface roughness package written in RMIT CSSE[Abd] to analyse surfaces roughness. This has been used to characterise nanosurface digital data sets from microscopy[Iva09]. For any given surface, the Roughness package calculates the following 8 parameters

| | |
|---|---|
| Mean plane | $R_a$: Roughness average, the arithmetic average of absolute values |
| $R_q$: Quadratic mean: Roughness average, the arithmetic average of absolute values | $R_p$: Maximum profile peak height |
| $R_v$: Maximum profile valley depth | $R_t$: Maximum Height of the Profile |
| $R_{sk}$: Skewness of the profile | $R_{ku}$: Kurtosis of the profile |

Table 1: Surface roughness parameters

## Approach

To ensure project targets were achieved in time, Agile software development principles were adopted. The project spanned over a period of 3 months and was broken down into 5 Sprints of 2 weeks each. Regular meetings were organized at the beginning of each sprint where goals for the next Sprint were decided and tasks from previous sprints were evaluated. The following table shows the breakdown of tasks for each Sprint.

| Sprint 1 | Sprint 2 | Sprint 3 | Sprint 4 | Sprint 5 |
|---|---|---|---|---|
| Get access to NeCTAR cloud dashboard, create MyTardis instance | Install Chiminey within a container | Register the MyTardis instance as Data Curation System in Chiminey | Switch to 3DRAC branch of Chiminey and test the roughness algorithm connector | Write package to translate image data to input format for 3DRAC |
| Understand how docker containers work and installaing them on nectar nodes | Understand the main components in Chiminey | Create a realistic experiment in MyTardis | Test 3DRAC connector using sample data | Test 3DRAC connector on the image data |
| Install MyTardis within container, and setup an instnace on a Nectar node | Setup a Chiminey instance on a Nectar node | Fork existing MyTardis and Chiminey repositories to incorporate biofilters module | Investigate the HRMC connector and attempt changes | Write package to translate LIDAR elevation model to input format for 3DRAC |
| Install the biofilters module in MyTardis to extract metadata from datasets | Execute HRMC (Hybrid Reverse Monte Carlo) type connectors. HRMCLite | Register HRMC connector and test using sample data | Create some sample data by converting images to input matrices | Test 3DRAC on LIDAR digital elevation model data |

Table 2: Sprints

# Methodology and Results

## Configuring MyTardis

To utilize the advantages of docker, a docker build of MyTardis[Thob], developed by Ian Edward Thomas was installed on a Nectar instance. It creates an assembly of containers to provide a fully functional MyTardis instance, running on a single host.

MyTardis allows users to configure custom filters to extract metadata upon ingesting new data. Extraction of metadata from insect data was done by installing MyTardisBF, a biofilters metadata filter package developed by Keith Schulze[Sch]. The package uses the Bioformats library via python-bioformats and supports the extraction of preview images and limited metadata information for all images supported by Bioformats. Additionally, the filter supports extraction of metadata and preview images.

To install this filter, the docker build was modified to include the filter package. The following files had to be modified to install dependencies and the bioformats package.

- `options/db/postgres/Dockerfile`
- `mytardis-portal/settings.py`

The filter was tested by uploading sample images in png, and jpeg format. The table below shows metadata extracted by the filter on uploading a png image shown in Figure 12(a) into an experiment in MyTardis.

| | |
|---|---|
| Dimension Order | XYCTZ |
| ID | Image:0 |
| Name | visual.png |
| Samples per Pixel | Channel 0: 4 |
| SizeC | 4.0 |
| SizeT | 1.0 |
| SizeX | 160.0 |
| SizeY | 160.0 |
| SizeZ | 1.0 |
| Pixel Type | uint8 |

Table 3: Roughness parameters

## Configuring connectors in Chiminey

Similar to the docker version of MyTardis, a docker version of Chiminey[Thoa] developed by Ian Edward Thomas was installed. As mentioned earlier, Chiminey allows developers to build custom computation tasks through smart connectors. The architecture of a smart connector is such that, it takes a payload which must contain the computation task to be executed. Chiminey provides a web interface for these connectors to enter the data and other parameters.

Chiminey platform also provides two types of parameter sweeps:

- External parameter sweep
- Internal parameter sweep

### External Parameter Sweep

External parameter sweep allows users to simultaneously submit and run multiple jobs. The external sweep allows the user to provide a range of input values for a set of parameters of their choice, and the resulting set of jobs span all possible values from that parameter space.

### Internal Parameter Sweep

Internal parameter sweep allows developers to create a smart connector that spawns multiple independent tasks during the smart connector's job execution. When a smart connector is created, the developer includes a set of parameters that will determine the number of tasks spawned during the execution of the smart connector.

### Connector Structure

The typical structure of a connector is:

```
connector
|---payload
|   |--- bootstrap.sh
|   |--- process_payload
|   |    |---main.sh
|   |    |---<executable jar>
|---initialise.py
```

Figure 1: Typical directory structure of a connector

The payload must be wrapped inside a connector which then runs it across VMs in the cloud. Before diving into developing a connector for the roughness package, time was spent during sprint 3 to familiarize with how connectors could be modified. Other connectors, such as random number generator and HRMC (Hybrid Reverse Monte Carlo) type connectors was investigated.

### 3D Roughness Analysis Connector

To be able to run the Roughness Analysis package through Chiminey a custom smart connector was built which wrapped the package inside its payload. The next step was to configure this branch of Chiminey[Thoa] with support for the roughness package connector. The structure of the connector was as follows:

```
connector
3DRAC
|---payload_3drac/
|   |--- bootstrap.sh
|   |--- process_payload
|   |    |---main.sh
|   |    |---run-rac.py
|   |    |---roughness-analysis-cli.jar
|---initialise.py
```

Figure 2: Directory structure of the 3D Roughness Algorithm Connector

3D Surface Roughness Analysis Connector (3DRAC) allowed running the Roughness analysis package through Chiminey. In standalone mode Roughness package accepted an input datafile with roughness information and reported roughness analysis result through Java Swing GUI. However, in Chiminey, the roughness package receives the input datafile through the 3DRAC connector which accepts name of the input datafile through Chiminey's browser based interface.

Chiminey interface for 3DRAC accepts a couple of inputs. The 'Data file name' field accepts a data file name that is located in the 'Input Location'. An example input data file file may be given as following:

```
0.0,   4.96   5.45   5.20   5.20   5.45
11.6,  5.45   5.20   5.20   5.69   5.69
23.2,  5.69   5.69   5.69   5.69   5.45
34.8,  5.45   5.69   5.45   5.69   5.94
46.4,  5.45   4.96   5.20   5.69   5.20
```

Figure 3: Sample input

The input data file is taken as a Cartesian plane excluding the first field in each line, which represents the x and y coordinate. The 3DRAC also accepts another field, the 'Virtual blocks list'. 'Virtual blocks list' field for 3DRAC in Chiminey portal accepts list of virtual blocks which is a parameter sweep that dictates the internal sweep. It allows the connector to break the Cartesian plain into blocks and spawn a task for each one of these blocks. This functionality is implemented in the `run-rac.py` file in the payload directory. Following is an example of a 'Virtual Blocks list'.

```
[ [0,0,4], [2,1,3], [3,3,5] ]
```

Figure 4: Sample virtual blocks list

## Translating LIDAR Digital Elevation Model to Roughness Package input

For the roughness connector to accept insect data, a package had to be written to translate an image into the above input format. The package was written and tested in three stages:

1. Convert a colourised hillshade jpeg image
2. Convert LIDAR digital elevation model and verify the results.
3. Test the connector with a larger dataset.

### Converting colourised hillshade

Pixel information of an image across the x and y axes closely resembles the format expected by the roughness package. To convert the image a java package was written which accepts an colour image in jpeg or png format and extracts the RGB value of each pixel, and arranges it in the format accepted by the 3DRAC. This ASCII file could then be provided to the connector through the Chiminey portal for 3DRAC.

During this experiment, it was decided to test the investigate Chiminey's external sweep in conjunction with the connector. Another Java package was written which could extract one of the four channels, Alpha, Red, Blue, Green and also luminance of the pixel from the RGB value. This jar file for the package is executed in `run-rac.py` file and runs on the virtual blocks cut out from the input for that particular task. The package would then read each block and convert the RGB values to either one of the four channels, supplied in the external sweep. The channel would be entered as a parameter value of an external variable 'channel', in the external sweep. For example, if the user wanted to extract the Red and Green channels, they would enter `{"channel": ["r","g"]}` in the 'Values to sweep over' field in the interface. Similarly, roughness of green channel or luminance can be calculated by supplying, 'g' or 'l' respectively.

Due to unavailability of insect data during the time of the project, other sources of sample data had to be investigated. The data used for the transformation was the surface elevation information recorded by the Mars Orbiter Laser Altimeter, an instrument on NASA's Mars Global Surveyor, rendered in colourised hill-shade[Ast01]. Two virtual blocks were supplied along with the input file containing the RGB matrix of the image. All four channels were supplied for external sweep.

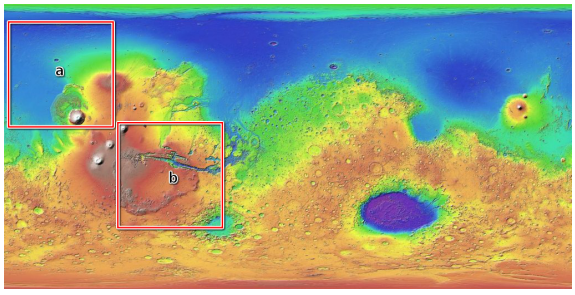
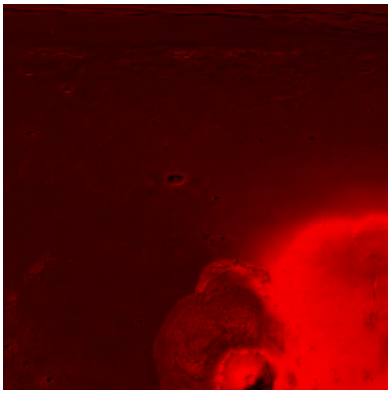
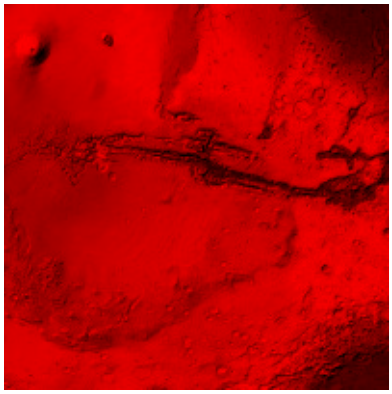



Figure 5: Colourised hillshade of Mars surface. The regions, (a) and (b) corresponding to the two virtual blocks selected by the internal sweep

For the sake of simplicity, only the average roughness($R_a$) values from the surface analysis results have been shown. Figure 6, 7 and 8 below show the average roughness of red, green and blue channel respectively on a scale between 0-255. Blue channel shows the highest value of average roughness for the 1st block and lowest for the second. Figure 9 shows the average roughness in luminance of the blocks on a scale between 1-0.

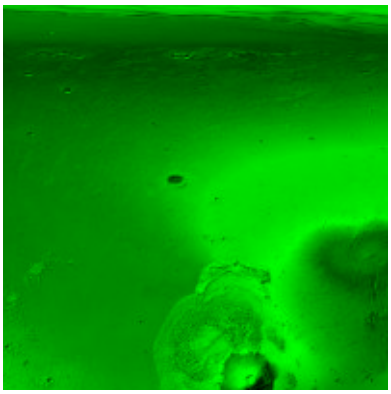


(a) 42.63

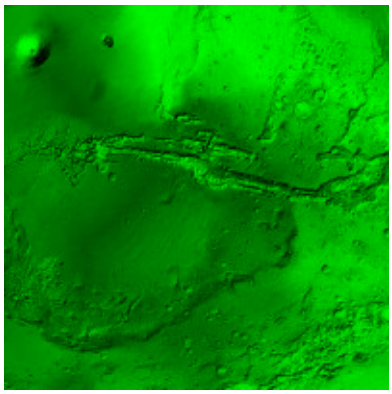


(b) 33.62

Figure 6: Red channel

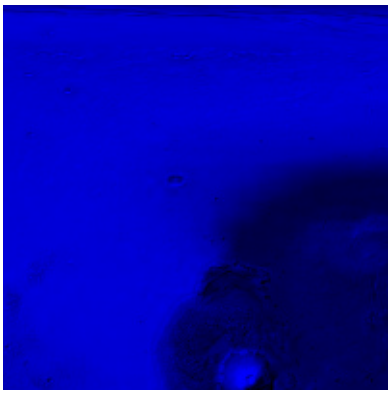


(a) 33.72

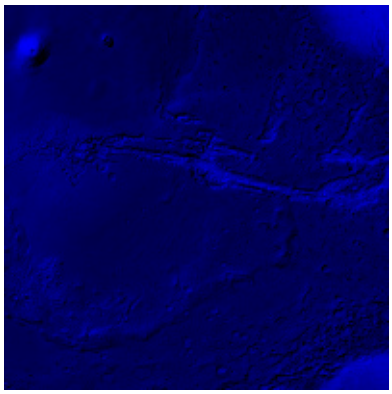


(b) 38.25

Figure 7: Green channel

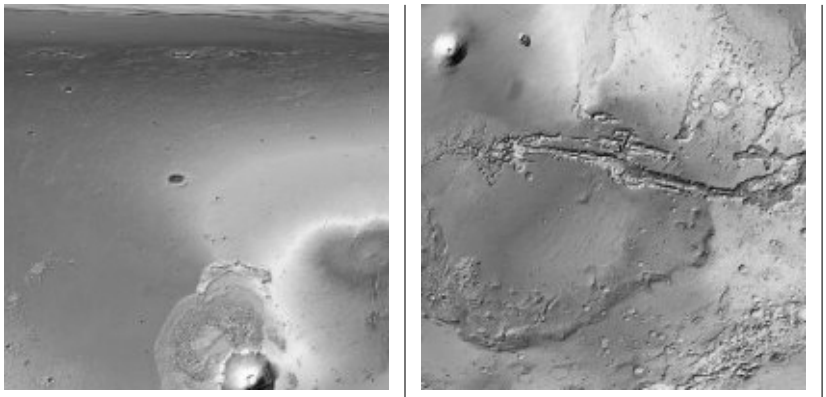


(a) 45.23



(b) 18.28

Figure 8: Blue channel



(a) 0.114 (b) 0.116

Figure 9: Luminance

**Converting a LIDAR digital elevation model**

The next step was to use an actual LIDAR dataset and convert it to a 2D image. Test data was obtained from LIDAR captured for BOPLASS Ltd by Aerial Surveys in January through April 2015 at Tauranga and Coast, Bay of Plenty, New Zealand, through OpenTopography[bAS]. A tiny section of the data near the coast was used for the test. The images below visualize the LIDAR LAS file for the section which contains information of approximately 40000 points.

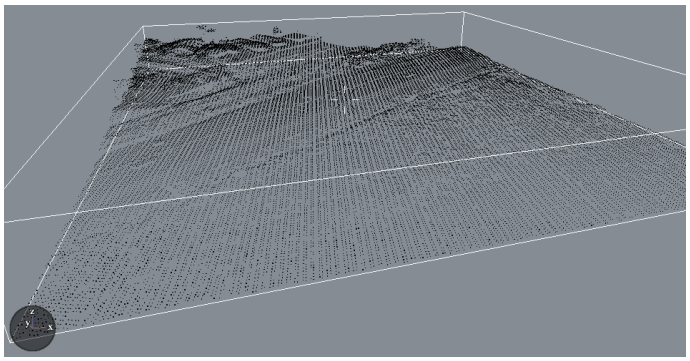
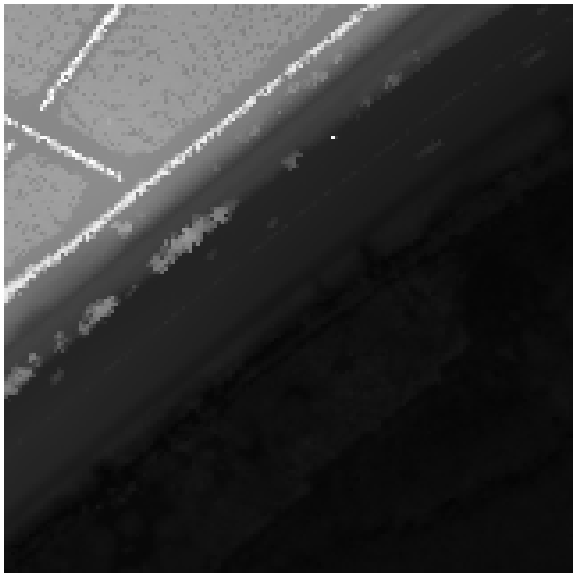


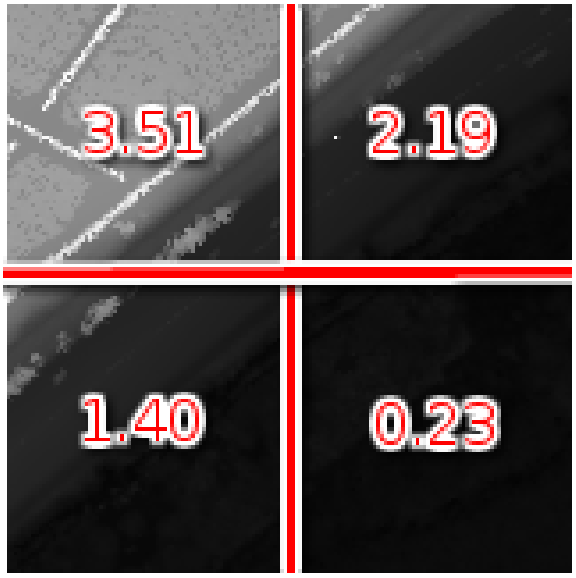
Figure 10: Tauranga LIDAR data section visualized using Displaz

The DEM(Digital Elevation Model) file was extracted from the las file using TIN (Triangulated Irregular Network) interpolation algorithm, a functionality provided by OpenTopography. A grid resolution of 1 metre and maximum triangle size of 50 units was used to generate an ASCII DEM file containing exactly 25600 points. A Java package was written to translate the DEM file into the input format for the 3DRAC connector.

A new job was then started in Chiminey using the translated input, with no external sweep parameters. The data was divided into 4 virtual blocks, each containing exactly 6400 points. This particular section of the data, corresponds to a part of the Tauranga the coast. As seen in the Figure 11(a), the bottom right part of the image which is uniformly dark relative to the top left corner, is part of the sea. A low Average roughness ($R_a$) value was expected for the bottom right block based on the assumption that ocean surfaces tend to be smoother compared to land surfaces. The results in Figure 11(b) resonates with the expectations. The bottom right block covering the surface of the sea shows an Average roughness value of 0.23 metres. These results proved that the Connector and the Roughness Package were showing results that were expected, given the nature of the data.



(a) Digital elevation model visualization




(b) Average roughness ($R_a$) values for individual blocks in metres

Figure 11: Roughness analysis of a section of Tauranga LIDAR data

**Test the connector with a larger dataset**

The next step was to test the data on a larger dataset. For this purpose another sample section of LIDAR data was used from The Victor Harbour Lidar Project[iHPRUp], which was captured during September 2011 by Photomapping Services using an airborne Optech Gemini Lidar system. The chosen section of LIDAR data had approximately 3 million points. Similar to the previous test case, the DEM(Digital Elevation Model) file was extracted from the las file using TIN (Triangulated Irregular Network) interpolation algorithm. A grid resolution of 1 metre and maximum triangle size of 50 units was used and the resulting matrix contained 1 million points.

Another Java package was developed and added to the payload to visualize individual virtual blocks in the final results. The package works by translating height data to grayscale pixel intensity. The lowest point is depicted with an intensity value of 0 while the highest point is depicted with an intensity value of 1.

After translating the data to the input format accepted by the Roughness Analysis package, two jobs were created in Chiminey.

- Job 1: Only one block (Figure 12) for the internal sweep, which encompassed the entire section of data.
- Job 2: The entire surface was uniformly divided into 16 virtual blocks of 62500 points each. Figure 14 shows how the data in Figure 13 was divided into 16 virtual blocks.

For job 1, the average roughness value ($R_a$) was 43.93 metres. For job 2, the roughness values for individual blocks can be seen in Figure 13 along with the elevation visualization.

It is important to note that, although the data was divided into uniform regions, it is not a requirement for the connector or the roughness analysis package. The 'Virtual Blocks' field allows user to enter any number of blocks, of any size and they may overlap with each other. However, for the sake of convenience, and to demonstrate the variation in roughness in the surface, the regions of equal size were picked at equal x and y offsets, so as to cover the entire surface.

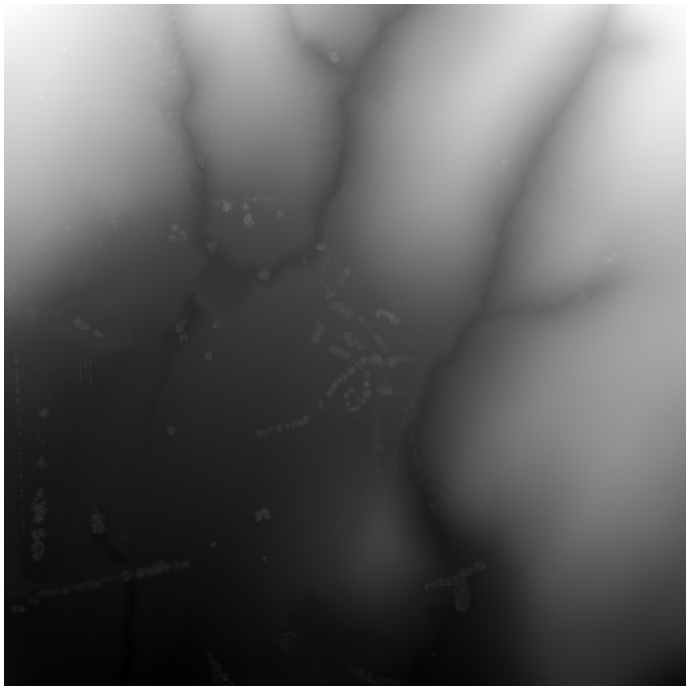


Figure 12: Victor Harbour, digital elevation model visualization, ($R_a$) = 43.93 metres

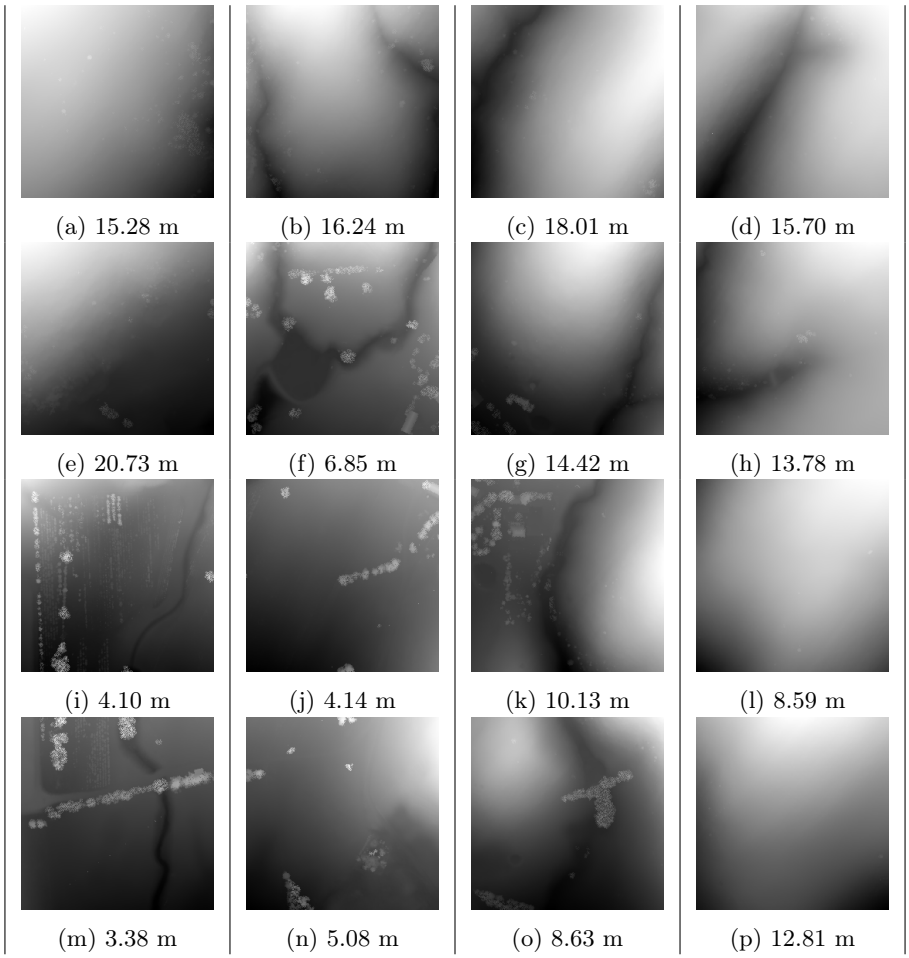



Figure 13: Average roughness ($R_a$) values of individual blocks in metres

## Conclusion

The above experiments and results demonstrate how the three components, MyTardis, Chiminey and the Roughness Analysis Package can be brought together to create a surface roughness analysis tool that is scalable using cloud computing, and independent of the scale of the data.

With respect to insect data, a majority of the data are images that are in full colour, and show the front, side and top view of the insects. To process insect data using the toolkit would require some amount of preprocessing using plugins that could:

1. Create, from these image collections, front, top and side view of these insects along with other metrics such as actual dimensions, and colour characteristic.
2. Center these views in a tight bounding box.
3. Process roughness, colour channel information and other statistical distributions.

These operations are single-pass and can be highly parallelized using the toolkit, and repeatedly, given growing image collections. The toolkit workflow currently designed and implemented demonstrates critical elements of such an overall workflow, using public domain images.

## Further Discussion

As demonstrated by the features of the toolkit, an entomologist, who knows the differentiating characteristics, needs to be able to select one of the views and define a window on it for high-throughput classification and fingerprinting. This extra step required in using the toolkit could be eliminated and is a topic of interesting future research project. A study to determine whether an automatic classifier using artificial neural networks can perform these classifications without using windows and by learning to look for the right differentiating bits to achieve accuracy of classification without human expertise at all.

## References


[Abd] Ahmed Abdullah. Roughness package. *https://bitbucket.org/ahmabd/roughnessanalysis.*

[Ast01] Astrogeology. Overview of the mars global surveyor mission. *Journal of Geophysical Research*, 106(10), 2001.

[bAS] OpenTopography: BOPLASS Ltd by Aerial Surveys. Tauranga and coast, bay of plenty, new zealand lidar data. *http://opentopo.sdsc.edu/datasetMetadata?otCollectionID=OT.122016.2193.4.*

[iHPRUp] OpenTopography: Coastal Urban Digital Elevation Modelling in High Priority Regions (UDEM) project. Victor harbour lidar data. *http://opentopo.sdsc.edu/datasetMetadata?otCollectionID=OT.062013.28354.1.*

[Iva09] Elena P. Ivanova. Impact of nanoscale roughness of titanium thin film surfaces on bacterial retention. *Langmuir*, 26(3), 2009.

[Sch] Keith Schulze. Mytardis bioformats filter package. *https://github.com/keithschulze/mytardisbf.*

[Thoa] Ian Edward Thomas. Docker version of chiminey. *https://github.com/ianedwardthomas/docker-chiminey.*

[Thob] Ian Edward Thomas. Docker version of mytardis. *https://github.com/ianedwardthomas/docker-mytardis.*


Codebase:

1. Chiminey with 3DRAC: `https://github.com/larvinloy/chiminey/tree/3drac_chiminey2`
2. Docker MyTardis with biofilters: `https://github.com/larvinloy/docker-mytardis/tree/mytardisbf`
3. Docker Chiminey: `https://github.com/larvinloy/docker-chiminey/tree/3drac_chiminey2`