**Introduction**

The purpose of the report is to document the findings of the investigation into the quality of a piece of software developed for texture assessment.

A texture assessment is carried out by processing data collected by a surface texture instrument in order to generate statistical or frequency spectrum of the data collected.

The program calculates parameters and spectral data from the above instrument. In particular:

*Load data from file –* The data is stored in a text file. It is assumed that it was captured using a texture assessment device

*Calculate the frequency spectrum data –* The algorithm performs calculations of the Fourier transform

*Save the spectral data –* The data is stored in text format file

*Compute parameters –* The parameters for the surface texture are computed

The report will focus on the accuracy and quality of algorithms for delivery of these parameters.

**Quality Requirements**

The quality of software depends on how well it meets the requirements. There are two types of requirements; functional and non-functional.

Functional requirements detail the operations that the software needs to perform such as, in the case of this task, store data in text format or calculate the Fourier transform of the parameters.

Non-functional parameters would include things that support delivery of functional requirements such as readability of the code and availability of support service for the end user.

There is a degree to which quality is necessary however, which varies depending on the use case. For example, a child’s toy does not require the same degree of quality as a control system for a nuclear reactor. The first product can fail as long as it does not cause injury and is fit for purpose while the second must have a very high degree of reliability as failure will result in catastrophic damage to the surrounding area and massive loss of life.

While the minimum standard of quality is required, it is sensible to exceed this standard. However there is a limit to how much quality is feasible to achieve given time and resource constraints. For example, cost, development time, development tools and libraries as well as the ways of working within the development team.

Therefore the quality requirements for this code would be:

Functional:

* The program must load data from file
* Calculate frequency spectrum of data in form of a Fourier transform
* Save the data in .txt format
* Calculate surface texture parameters

Non-functional:

* Maintainability of code
* Reusability of code
* End user support
* Compatibility with the gathered data

**Testing Requirements**

Testing is a practice of running a program with direct intention of finding a previously undiscovered fault. Depending on the complexity of the code and availability of resources, it may be difficult to carry out a full test of the code. For example, the time/cost implications may be too severe or the code may require a piece of hardware that is unavailable. Also, testing becomes more difficult with increase of complexity of the system.

Therefore testing should be carried out whenever it is feasible to do so. Furthermore it is a process that requires strategy and planning as, in order to deliver a robust piece of code, many tests would be required. It may thus be necessary, at least from cost/time investment perspective, to automate testing as much as possible.

The aim of testing is to indicate presence of errors and reduce defects. However testing has some limitations so far as the extent of testing is determined by various standards - in-house, national and international.

There are two types of testing, static and dynamic. Static testing focuses mainly on error prevention through code inspection or automatic tests by various tools or computers. The tools include MISRA C, Embedded C++, and Lint. The code is not compiled or run during static testing.

Dynamic testing is different from static testing in requiring the code to be compiled and run. Manual testing would include use by ‘beta’ testers or end users. It can also be tested automatically through heap checking which involves checking whether there is sufficient memory for the dynamic variables, that no garbage data is collected and stack checking where actual memory use is compared to the defined value.

Furthermore there is white box testing which focuses on individual functional blocks and black box testing which focuses on the operation of the code as a whole on the interface level.

The tests can be performed either by an in-house software developer team or outsources to independent tester.

**Reliability**

Reliability is defined as the probability that the software does not fail in a specified time frame and environment.

The interesting thing about reliability when applied to code is that, while physical objects may be subject to wear and tear, software is very much the opposite of that in a sense that it becomes more refined with each revision that correct the known defects.

There are three stages of software failure.

Fault is a defect that can propagate to the subsequent software components and cause errors down the line. Errors occur when the actual state of the software is different from the defined state, causing failure. Failure then occurs when the component ceases to perform its function.

**Conclusion**