Carolyn Holness

Generating Space-time Composite Images for GIS Data

Submitted in partial requirement for fulfilment of
Master of Science (Information Technology)

Department of Computer Science
University of Cape Town

February 2006

Abstract

For many decades scientists have been collecting marine data from survey boats that spend weeks at sea covering large areas and collecting data at regular points along the journey. With the introduction of remotely sensed satellite images, large amounts of environmental data (e.g. sea surface temperature) is now available to be used in studies on the effect of the ocean on phenomena observed on such journeys.

Up until recently, mapping this environmental data from the satellite images has been done by averaging daily images for the duration of the survey to obtain a single image approximating the entire time span and the whole area covered in the survey. The problem with this however, is that in creating these temporal composite images, values are averaged out so that small variations that occur over short time scales are lost. In an effort to overcome this problem, the idea of a spatio-temporal composite image was proposed. This entails using pieces of daily images, where each piece represents the area covered by the survey on the same date, and merging these pieces to create a single image. This allows the original values from the satellite images to be retained which provides a much more realistic mapping of the environment.

Satellite images and survey data can be viewed and analysed in a Geographical Information System (GIS), however creating these spatio-temporal composite images manually using the software is a time consuming and difficult task. This project investigates whether it is possible to develop an application to automatically create these images. On completion of the project it was found that not only was this possible, but that it also provided the means to create these images more accurately and far quicker than could be done manually and it was also easy to use even without prior GIS experience.

Table of Contents

List of Figures

1 Introduction

1.1 Objectives and Motivation

Scientists have long been studying the effect that the environment has on the population dynamics of species both on land and in the ocean. Such studies in the ocean are complicated by the fact that the ocean is very dynamic where environmental conditions are changing constantly due to the continuous movement of the water caused by wind, currents, tides etc.

Collection of environmental data in the ocean has traditionally been done by *in situ* techniques such as floating buoys or survey and research boats. The problem with these techniques is that it is all but impossible to collect data for large areas due to the sheer size of the ocean and limited area that survey or research boats can cover over any set time.

Remote sensing technology has however changed this, and now allows the collection of large quantities of environmental data through satellite images. This has opened opportunities for new avenues of research and is especially useful for studies of the ocean because the satellite images cover large areas, are taken at regular intervals and can measure a range of ocean environmental parameters including sea surface temperature, chlorophyll concentrations, wind etc. The images are therefore useful in capturing the changes in environmental conditions in the ocean that occur over small time and space scales, which was almost impossible to do previously. Added to this, the images can easily be incorporated into a Geographical Information System (GIS) where they can be used in conjunction with biological or environmental data collected by traditional means.

Marine and Coastal Management (MCM) in Cape Town is concerned with the protection and management of marine and coastal resources off the southern African coast, including the rich fisheries resource sustained by the ocean waters in this area. Much data has been collected in the past during surveys where research boats spend many weeks at sea collecting data at regular points in the ocean.

Relating data collected in these surveys with environmental data from satellite images has in the past been done by creating a temporal composite image for the duration of the survey. So for example, daily chlorophyll images for each day of the survey are averaged out to create a single weekly or monthly composite image that can be applied to the data points for the whole survey. The problem with this however, is that

averaging these images smoothes out the values, so that variations that exist over small time and space scales are lost.

Parameters such as temperature and chlorophyll concentrations may change gradually over a number of days and while small variations may be lost, averaging these may still produce fairly meaningful results. The same cannot however be said for parameters such as wind speed or wave size which can change significantly over very short time periods. A gale force wind blowing for a few days followed by a few days of absolute calm would average out to a moderate wind for the duration of the study period. This type of result could potentially have a great impact on the result of an investigation into the effect of wind speed on a particular entity. This problem of averaging values is even more problematic when looking at parameters such as wind or wave direction. A southerly wind blowing for one day and a northerly wind blowing the next cannot be averaged out to any meaningful value. Creating temporal composite images for parameters such as this would be completely meaningless.

Scientists at MCM have been investigating ways in which the environmental data from the satellite images can be realistically mapped so that the original values in the images can be retained. This would not only allow parameters such as wind and wave direction to be mapped in a manner that was meaningful, but for parameters such as sea surface temperature and chlorophyll it would allow the variations that occur over small time and space scales to be retained and represented on a map.

They proposed the idea of a spatio-temporal composite image which creates a single composite image that can be applied to the whole area covered by the survey, but instead of averaging the daily images to create the composite image, they suggested using portions of the daily images, where the portion would represent the area covered by the survey on that date. The small portions of daily images would then be combined to create a single composite image. The advantage of this is, that as before, a single composite image can be applied to the whole area of the survey but now the original values are retained rather than being averaged values. Such composite images could be called spatio-temporal composite images because their composition is determined by the spatial and temporal extents of the surveys rather than the spatial aspect alone.

Such space-time composite images can be created in a GIS using a sequence of different functions provided on the software's user interface. Creating the images is however a very time consuming process. It is also a complicated task that even

experienced GIS users would find very difficult. It was therefore proposed by MCM to develop a tool that would automate the process of creating these composite images. As far as we could ascertain after a thorough search and to the knowledge of those at MCM who proposed the idea, no such tool existed, providing the opportunity to undertake a novel project.

1.2 Project Description

The aim of the project was therefore to investigate whether a tool could be developed to automatically create space-time composite images. The investigation would include determining whether the tool:

1) Would be easily useable even by inexperienced GIS users.
2) Would reduce the time taken to make the images.
3) Could be developed generically so that it could potentially be used for a variety of data not just the survey data from MCM. Data that could potentially be used would be any 'point' data that is collected over large spatial areas and long time spans, for example tracking or tagging of animals or fish.

This thesis starts with a more detailed background to the project. The specific requirements for the project are outlined including the steps that are performed to create one of these composite images. The design of the tool is then discussed which is followed by a detailed look at how different aspects of the design were implemented including the user interface of the tool. On completion of any system it is important to test and evaluate the final product. Details of testing the tool, both in terms of functionality and accuracy of the output, as well as testing usability, is then discussed. The thesis ends with a conclusion that outlines the main findings of the project.

2 Background

This chapter starts by putting the project into context by giving a background to marine and fisheries research and some of the issues involved. Following this is a discussion of remote sensing techniques as a source of data for this research. This is followed by a section on features of Geographical Information Systems and how they can be used as a tool to aid marine research, including ways of incorporating remotely sensed data. Finally, a description of GIS software is given including a more detailed look at developing applications with this software.

2.1 The Marine Environment

2.1.1 Marine and Coastal Management

Marine and Coastal Management (MCM) is branch of the Department of Environmental Affairs and Tourism. It is concerned with the development and conservation of marine and coastal resources to ensure the sustainable use of these resources, as well as to maintain marine ecosystem integrity and quality.

South African scientists from MCM and the University of Cape Town have been working in collaboration with French scientists over the last few years on a research project called IDYLE (Interactions and Spatial Dynamics of renewable resources in upwelling Ecosystems). This is a broad study being carried out to quantify the potential spatio-temporal interactions for the different trophic levels in the Benguela ecosystem.

2.1.2 The Benguela Ecosystem

The Benguela ecosystem, located off the southern and western coasts of southern Africa, is identified as being one of the large marine ecosystems of the world. It is characterised by strong upwelling caused by the predominant south-easterly winds which blow in this area and cause an off-shore flow of surface water. This water is then replaced by cold, nutrient rich water from the deep ocean. This process is illustrated in Figure 2.1. Other features of the Benguela system include the warm Agulhas current from the Indian Ocean in the south, and a seasonal coastal jet current. These features make the system highly complex and dynamic, with high seasonal and inter-annual variability [SO98].

Strong upwelling leads to high levels of phytoplankton which use the nutrients in the surface waters. This plankton is then able to support a large biomass of higher trophic level species. The Benguela system is noted for being one of the most productive

ocean systems in the world, supporting a large numbers of fish, crustaceans, sea birds and marine mammals [NOA03]. Such large numbers of fish in the Benguela system make it an extremely rich fishery resource. In order to ensure that this resource is exploited in a sustainable manner, management policies are needed which must be based on a good understanding of the ecosystem dynamics.

Figure 2.1 Upwelling in the Benguela system

2.1.3 Links Between Biological and Environmental Processes

It is well known that biological processes including species abundance, distribution and breeding patterns are affected by environmental conditions and this holds true both for terrestrial and marine environments. In terrestrial environments these conditions may include water availability, air or ground temperature, humidity, surrounding vegetation etc. In marine environments, the environment of the species is the ocean itself, and the condition of the ocean. The ocean is in constant motion and the forces causing this movement include winds, waves and tides. This movement results in currents and other oceanic processes such as upwelling, fronts etc., which in turn affect the water temperature, oxygen content, PH, nutrients and stratification. It is these parameters and their spatial and temporal variation that ultimately have a strong influence on the population dynamics (distribution and abundance) of marine life [Dra03], including the important fish stocks for which management policies are required.

Many studies have been undertaken worldwide investigating the relationships between different oceanic processes or environmental variables, and the spatial and temporal distribution of species populations. For example, [WQ00] investigated the relationship of the recruitment of Pacific herring to environmental variables; [RGH98]

looked at the effects of environmental factors including temperature and salinity on phytoplankton communities in the Gulf of Finland, and [BH98] analysed environmental effects on survey catches of squid in the northwest Atlantic. Some studies have also been done on the Benguela system, for example, [PHH98] investigated the environmental factors affecting anchovy recruitment in the Benguela system. However, [NOA03] notes that there is still a limited understanding of this highly variable and complex system's physical and biological interactions and processes.

For any research related to the environment it is obviously necessary to take measurements of the environmental parameters in question. This can be done by taking *in situ* measurements; that is where the instrument comes directly in contact with the body being measured. So for example, a measurement of sea surface temperature may involve lowering a thermometer into the sea and taking a reading at particular sample points [Dud03]. Such methods have their obvious drawbacks: gathering data is difficult, expensive and time consuming, and it is near impossible to get accurate readings for large areas of ocean.

2.2 Remote Sensing

2.2.1 Development of Remote Sensing Techniques

Over the last 20 years, the development of remote sensing (RS) techniques has paved the way for much easier collection of large amounts of environmental data. The term remote sensing encompasses any method of data collection that does not involve any direct, physical contact with the object being measured. The technical term 'remote sensing' was first used in the United States in the 1960's [JAR96] and evolved as a by-product of the space program [GMR04].

Initially remote sensing was limited to what could be seen in the visual part of the electromagnetic spectrum i.e. the detection of visible light to produce photographs [Kin01, WGI02]. More recently, however, the parts of the spectrum which can not be seen with the human eye (e.g. sound and heat) were able to be utilised through special filters, photographic films, digital cameras and other types of radar and thermal sensors [Kin01]. These detect invisible electromagnetic radiation which is reflected or emitted from an object [JAR96]. The characteristics of an object can be determined because each object has a unique and different characteristic of reflection or emission if the type of the environmental condition is different [JAR96].

These sensors detecting either visible or invisible radiation can be located on aircrafts or can be space borne where the sensors are mounted on space shuttles or satellites orbiting the earth [Lie01]. Since Landsat-1, the first earth observation satellite launched in 1972, remote sensing has become widely used [JAR96] and at present there are several remote sensing satellites located at various positions above the earth's surface providing imaging for research and operational applications [JAR96, Lie01]. Each of these satellite-sensor platforms is characterised by the wavelength bands employed in image acquisition, spatial resolution of the sensor, the coverage area and the temporal coverage, i.e. how frequently a given location on the earth surface can be imaged by the imaging system [Lie01].

2.2.2 Advantages of Satellite RS

Satellite remote sensing provides the following advantages over other methods of data acquisition such as ground based surveys:

1) It can transmit images that are generated by heat, electromagnetic radiation, and sound, and not just the conventional visible light [GMR04].
2) It has a large area coverage.
3) It can cover remote areas or hostile environments which are difficult or dangerous to reach such as the polar regions, deserts and remote oceans [Sho04].
4) It allows frequent and repetitive coverage of an area of interest and so allows changes in events to be documented at regular intervals and facilitates research on time scales, both in duration and frequency that were previously impossible [Lie01, Sho04].
5) Data can be processed efficiently and automatically by computers making it possible to gather massive quantities of data that observers on the ground could not [GMR04].
6) Data can be readily integrated with other data or incorporated into Geographic Information Systems (GIS).

The agenda of this thesis is to take point 6 further: using RS satellite data in a GIS and integrating it with *in situ* point data.

2.2.3 Applications of RS

While remote sensing technology has diverse applications and is used in many disciplines including agriculture, forestry, geology, hydrology, cartography and meteorology, it is especially useful for oceanographic and marine research. The large extent of the earth's oceans makes it impossible to equip sufficient research vessels to

study more than a small area of an ocean at one time. Added to this, because the oceans exist as one continuous body of water, what happens in any one area of an ocean is dependent on processes at work in other parts of the world's oceans. This means that researchers need to study the oceans as a total system [OA98]. Remote sensing provides a solution to these problems in that it provides total ocean coverage with large amounts of data which could not be obtained by any other means. RS is also very useful in that it collects data at frequent, regular intervals. The ocean is a very dynamic environment where significant changes can occur over very small spatial and temporal scales. RS provides a means to capture these small changes.

Oceanographic data that can be derived from RS are ocean pigmentation (chlorophyll), currents (location, direction, speed), sea surface temperature, wind (direction, speed) and waves (location, direction, speed). Figure 2.2 below shows a satellite image of sea surface temperature (SST).

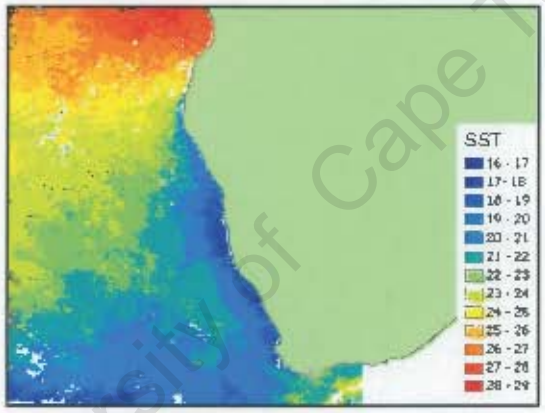


Figure 2.2 Satellite images showing SST

Being able to obtain such varied types of environmental information on large scales and at regular interval makes RS a very useful source of data for both studies of the ocean environment and the relationship between oceanic processes and the temporal and spatial distribution of fish species.

[CL99] point out that most previous work done on interactions between fish distribution and environmental parameters tend to assume micro-sale observations as representative of meso or macro-scale values. For example, *in situ* measurements of sea parameters made at single locations in the ocean are assumed as representative of an entire area and a few months' sparse measures are averaged to get proxy annual mean values. Satellite data helps in overcoming these limitations.

Although new technologies now allow satellites to infer the depth of the ocean, most RS satellite images that are available cannot penetrate below the ocean's surface [OA98]. This means that although this data is useful in its own right for many applications, it becomes most valuable when using in conjunction with *in situ* data collected of phenomena occurring below the ocean's surface for example vertical chlorophyll profiles, fish and eggs densities etc.

A number of studies have been done which incorporate the data available from satellite images. [Col99] used a time-series of mean weekly SST images to investigate the relationship between oceanographic conditions and the recruitment of anchovies and pilchards in the Benguela system and [CV00] use similar images to investigate the variability of SST and discuss the implications of these findings for fisheries research. [CL99] investigated the interaction between satellite derived environmental factors and Bluefin tuna behaviour in the Mediterranean, while [DF00] showed links between environmental indices derived from satellite SST images and recruitment patterns of two fish species off the coast West Africa.

2.3 Geographical Information Systems

A Geographic Information System (GIS) is 'a system of computer software, hardware, data and personnel that allows the storage, retrieval, manipulation, analysis and visualisation of geographically referenced spatial data'. In short, it is a 'technology that manages, analyses and disseminates information about the earth's surface and the features found on it' [GIS05].

2.3.1 Layers or Themes

The most basic function of a GIS is to take data which has a spatial location and produce a visual representation of that data. The most common form of this representation is on maps where related data is displayed as a layer. For example, different vegetation zones would be displayed in one layer and a road network in another layer. Layers are also called themes or coverages. Layers can be displayed individually or overlaid with other layers. When layers are overlaid, every location on one layer is precisely matched to its corresponding locations on all the other layers. Figure 2.3 below illustrates this concept, where five layers have been overlaid.

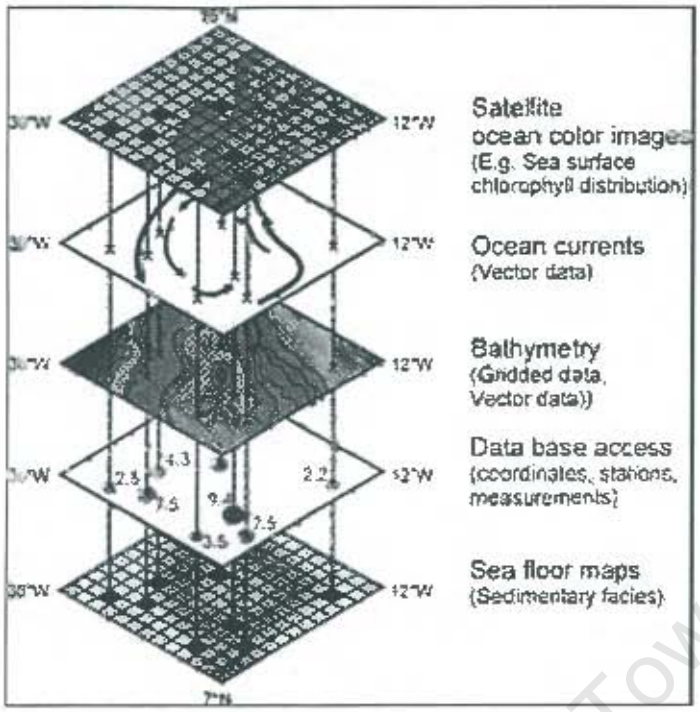


Figure 2.3 Different layers of data overlaid with each other (Diagram from MCM)

2.3.2 Types of GIS Data

There are two main models for storing and representing spatial data in GIS: the vector and raster data models.

Vector Data

Vector data is data where features are represented as points, lines or polygons. For example a house may be represented as a point, a street or river as a line and a country or vegetation zone as a polygon. Each feature is stored with a spatial location and a set of attributes. The spatial location is recorded as a single x,y co-ordinate for points or a set of x,y co-ordinates for lines and polygons. Figure 2.4 below shows how some real world features could be presented as vector data.

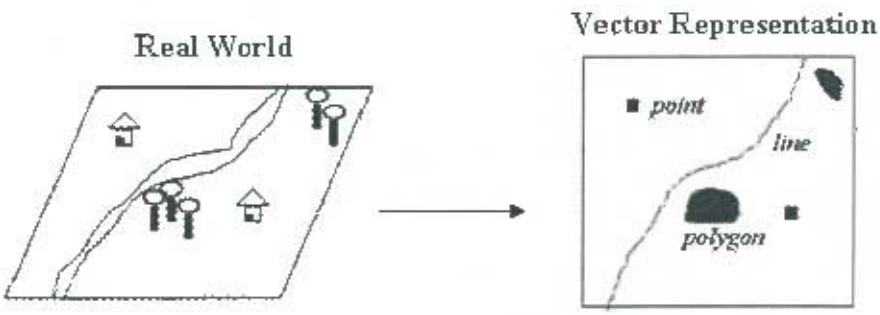


Figure 2.4 Real world features represented as vector data

Raster Data

Raster data is spatial data expressed as a matrix of cells or pixels. Each cell within this matrix contains location co-ordinates as well as an attribute value. The spatial location of each cell is implicitly contained within the ordering of the matrix, unlike a vector structure which stores topology explicitly. Figure 2.5 below shows how the same real world features shown in Figure 2.4 could be represented as raster data.

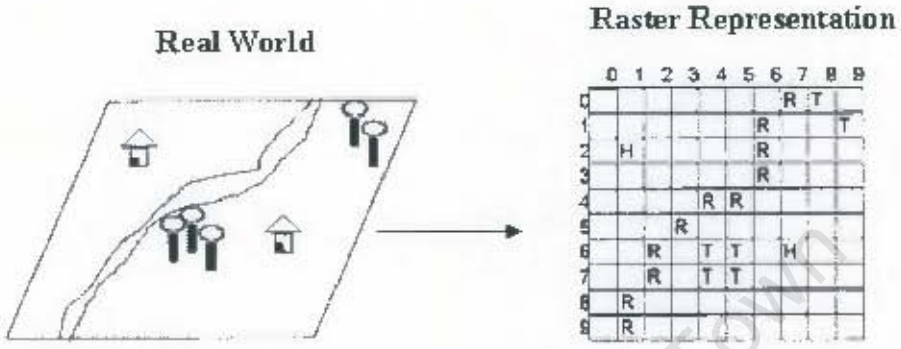


Figure 2.5 Real world features represented as raster data

An example of how layers of raster and vector data are displayed in a GIS is illustrated in Figure 2.6 which shows three layers displayed in a map.

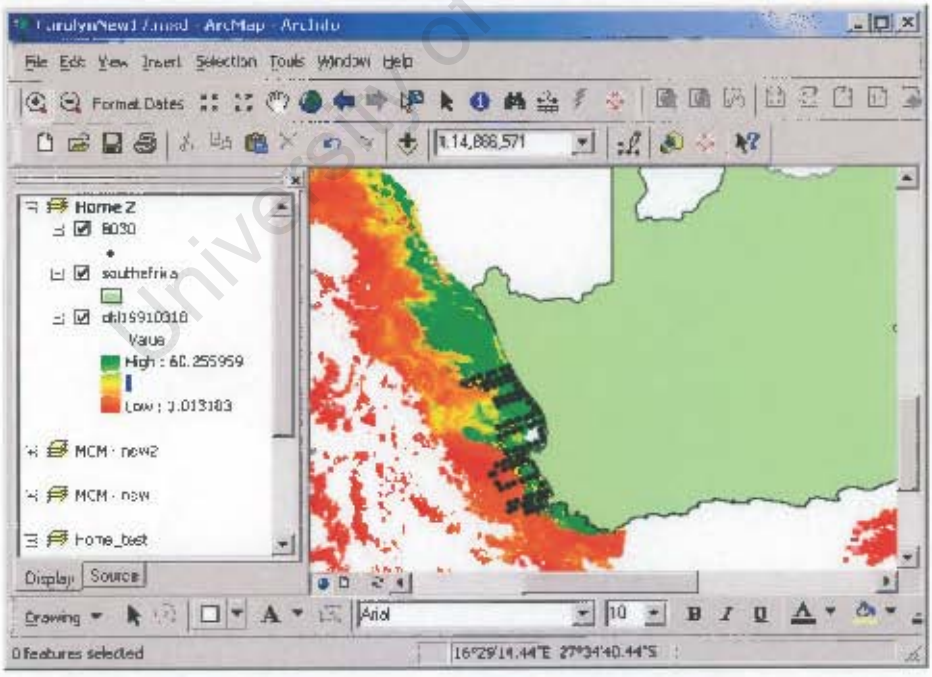


Figure 2.6 A map displaying three layers: 'B030', 'South Africa' and 'chl19910318'.

The 'South Africa' and 'B030' layers both represent vector data. The 'South Africa' layer contains a polygon feature (shown in light green) representing the land mass of South Africa, while the 'B030' layer contains point features (shown as black dots)

each representing a location in the ocean where sampling took place. Each feature is stored with a set of attributes which contain related information about the feature and which can be viewed in a layer's attribute table. Figure 2.7 below shows the attribute table for the 'South Africa' layer. There is only one feature in this layer: a polygon representing the area of land for South Africa. Area and Perimeter are two of the attributes shown. Figure 2.8 shows the attribute table for the 'B030' layer. This layer consists of a number of points. Each point is shown in the table with its associated attributes of position, date, period, temperature etc.

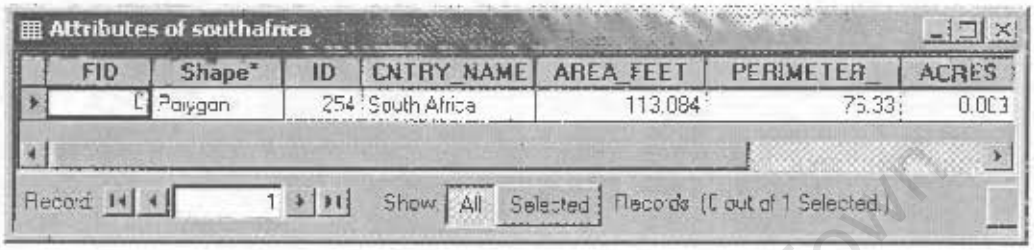


Figure 2.7 Attribute table for 'South Africa' layer

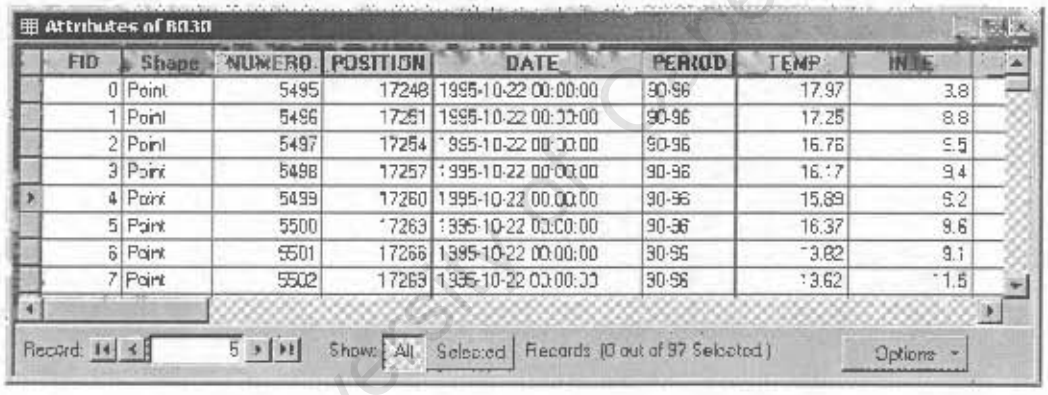


Figure 2.8 Attribute table for 'B030' layer

The 'chl19910308' layer (from Figure 2.6) is raster data of a remotely sensed satellite image indicating chlorophyll concentrations. The pixels are not apparent at this resolution, however Figure 2.9 below, which is zoomed in, illustrates the matrix of pixels or cells that make up the image.

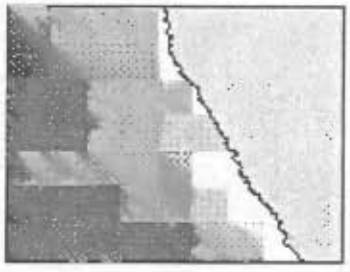


Figure 2.9 Close-up of a raster image showing individual pixels

The attribute table for the 'chl19910308' layer is shown in Figure 2.10 below. It consists only of a list of each pixel and its associated value, in this case, the chlorophyll concentration.

Figure 2.10 Attribute table for the 'chl19910308' layer

2.3.3 Features of a GIS

GIS technology integrates a number of different very useful features. It acts as a database management system by storing, organizing and managing data. It allows the sharing of data and can link different datasets together by common spatial locations. It provides tools for the input and manipulation of geographic information as well as for querying and analysis including statistical analysis. Finally it provides a means to visualise data not only through maps but also through tables, graphs etc.

Together, these features allow the exploration of possible spatial relations, patterns and trends that may exist within and between data, giving us a better understanding of the world around us. A GIS therefore provides a means to turn data into useable information which can be used to answer questions, solve problems and support decision making processes.

2.3.4 GIS and the Marine Environment

From their inception about 35 years ago, GIS's have traditionally been used for terrestrial environments [Mea99]. In the early 1990's, land-based applications accounted for almost all of the commercial market for GIS software. Since then, the use of GIS applications in the marine and fisheries sectors has slowly expanded with the recognition that GIS could hold the key to better management of the numerous spatially related problems that face these sectors [Bar99].

One of the reasons GIS's have been slow to be used in marine compared to land-based systems is a comparative lack of data for marine environments. In general, data is much more difficult to obtain for marine systems, which tend to be sampled sparsely and infrequently [Luc99]. With the development of RS techniques, this situation has however changed in recent years, with much more data becoming available from satellite images. RS of the marine environment comprises the greatest source of information about several parameters of the ocean surface and this data is invaluable for studying physical and biological oceanic processes [Val02].

Another reason for this relatively recent use of GIS applications in marine environments is due to the multi-temporal nature of marine data [Mea99]. Since the ocean is continuously moving, there can be significant changes in environmental conditions over very short time and space scales. Traditional measurements taken at sea, particularly on ships, do not permit a clear separation between time dependent changes of the environment and spatial changes. Environmental phenomena can affect entire marine regions over large spatial scales (from hundreds to thousands of km) and short time scales (from hours to days) and such processes are difficult to follow using *in situ* techniques [Bar99]. RS of the sea can however be used to overcome these problems in that it provides data repetitively over short time scales and for extended spatial areas.

RS images can be incorporated into a GIS where the images are represented as raster data. The availability of time series of RS satellite data provides new avenues for fisheries research into the relationship between species and their environment. To use this remotely sensed data to its full potential, there is however a need to integrate it with *in situ* data. [Val02] suggests that marine GIS development should incorporate a variety of raster and vector datasets and goes on further to note that development of large-scale, information-based fisheries management is limited due to the lacking of monitoring and decision aid tools that integrate the capabilities of the new technologies such as GIS and RS. There is therefore clearly a need to extend the use of GIS as a tool for marine and fisheries research by incorporating RS data.

2.3.5 GIS and MCM

MCM has acknowledged the need to use GIS technology to assist in understanding and making decisions regarding the management of fisheries resources in the Benguela system. Large amounts of *in situ* data collected by its marine scientists have been georeferenced and incorporated into a GIS. For example, acoustic surveys on board research boats have been conducted biannually from 1988 to the present along

the coast of southern Africa. Each survey lasts a few weeks, during which time fish and fish egg densities as well as phytoplankton and zooplankton abundances are collected. This data is collectively stored in a file that can be used in a GIS.

The relatively recent acquisition of time series of RS data for the coastal areas off southern Africa has opened opportunities for integrating this satellite data with the *in situ* data that has been collected over the years. A large number of images have been processed so that they are now available to be viewed, used and analysed within a GIS. These images have been processed from different sources (i.e. different satellites) and provide information on sea surface temperature (SST) and chlorophyll.

One aspect of the IDYLE project has been to study and compute spatial correlation indices between the *in situ* survey data and the remote sensed satellite images. One particular ongoing study is the spatial correlation between chlorophyll concentrations shown in satellite images and the *in situ* observations giving vertical profiles of plankton.

A number of tools and methods have been developed to use and incorporate these different data sources into studies on the Benguela and other ecosystems worldwide. This project involves developing a tool which integrates RS data with traditional *in situ* data, the results of which will be used to aid studies of the spatio-temporal interactions between biological resources and the environment.

2.4 Composite Images

2.4.1 What is a Composite Image?

Satellite images are available as daily images or as user defined ones such as weekly or monthly. These weekly or monthly images are called composite images as they have been produced by combining a number of single images. The values (for each pixel) in a composite can be calculated by a number of methods. For example, the highest value from the individual images (for each pixel) can be used in the composite image. Alternatively the values for each pixel from the individual images can be averaged and this value used in the composite image. 'Averaging' the values is the most common method for creating composite images.

Such composites have the obvious benefit that they may display a relatively complete set of data compared to daily images which may have large areas of data loss due to cloud cover. For many applications, the use of such composites with the averaging of values may suffice. For example, in a study of the seasonal variability in sea surface

temperature, weekly or monthly composites would give a good indication of the changes in temperature over the year and there would be no need to include the smaller fluctuations that exist over shorter time spans such as days.

2.4.2 Limitations

If environmental conditions remained constant over short periods these composites would be adequate. However, as has been noted, the ocean is very dynamic with constant changes in environmental conditions. Therefore, for other applications or studies, for example investigating whether the presence of a fish species at a particular location is influenced by the sea temperature, it may be necessary to capture these smaller changes in environmental conditions. The need to do this is also noted by [DF00] who, in a study on the effects of upwelling in the recruitment of two fish species, note that the seasonal aspects of recruitment and migration need to be considered at small space and time scales before a mechanistic level of understanding can be achieved. Such weekly or monthly composites would 'smooth out' any variability that exists over small time scales and therefore potentially mask biological patterns that may exist.

As an example, for integrating satellite images with *in situ* data collected on survey boats, using weekly or monthly composites covering the duration of the boat's journey would almost be pointless if one was attempting to correlate an environmental parameter from an image with data collected at a particular point. The value would be an average over a number of days, during which time the parameter may have fluctuated considerably. Ideally one would like to obtain the value for the given environmental parameter for the same date on which the data was collected for a given point.

This above example illustrates the potential problems and limitations that using composite images with their averaged values may have. It highlights the need, in certain circumstances, to preserve the original values of environmental parameters as much as possible.

All this confirms a need to take a new approach to integrating satellite data with *in situ* data that is collected over wide spatial areas and over long time spans. The example of survey boats collecting data around the coast of South Africa over a number of weeks illustrates the issues involved. The problem however, is extended to any data collected over a long time spans and large areas, for example, tagging and tracking of fish or other animals.

In this project, this problem is addressed by providing a novel solution to applying satellite images to data with large spatial and temporal extents. This solution comes in the form of a spatio-temporal composite image. Whereas before pixels were averaged over a number of days, usually weekly or monthly to create a time composite image, and this applied to the whole area covered by all the data points, with a spatio-temporal composite image, pieces of daily images are obtained which correspond to the area or spatial location that was covered on each day. These pieces of images are then spliced together to create a spatio-temporal composite image.

It is therefore necessary to find a way to determine the total area that was covered by the survey on each day. Since the survey is only represented as points, the total sea region must be divided into regions where each region is associated with a single point. There are a number of different ways to accomplish this, however, the method that will be used for this project is called a Dirichlet tessellation. Dirichlet tessellations are also known as Voronoi diagrams and in GIS terms Thiessen polygons. Thiessen polygons can be defined as 'polygons in which all areas in each polygon are closer to that polygon's central point than any other point' [Gui00].

Tessellation is therefore a key step in producing spatio-temporal composite images because it is used to divide the sea into regions associating each part of the ocean with the nearest survey point. Once this is done, the satellite image corresponding with the date of that point will be used for that point's region.

2.5 Dirichlet Tessellation

Although there are tools provided with the GIS software to easily create Thiessen polygons around a set of points, for reasons that are discussed in Chapter 5, it became necessary to write functions from scratch to perform this tessellation. The following section gives some background information on the algorithm used to write this code.

There is considerable information and many studies that have been conducted on Dirichlet tessellation/Voronoi diagrams. In particular, a close relationship exists between Voronoi diagrams and Delaunay triangulation. A triangulation involves creating from a set of points, a set of non-overlapping triangles, where the vertices of the triangles are the input points [Bou89]. A Delaunay triangulation is one where the circumcircles (the circle that passes through the three points of a triangle) of all the triangles do not contain any other points of the triangulation [Bou89]. Figure 2.11 shows the Delaunay triangulation for a set of points, named P1 to P5.

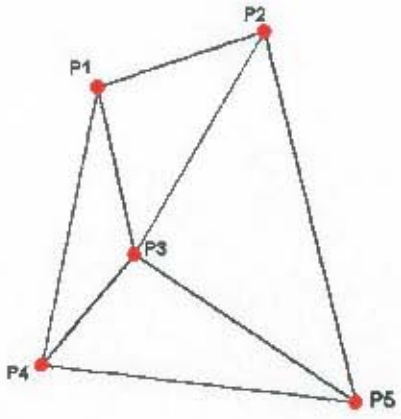


Figure 2.11 Delaunay triangulation for points P1 to P5

In Figure 2.12 below, Circles 1 to 4 are the circumcircles for the four triangles in the above Delaunay triangulation. It can be seen that none of these circles contain any other points. Should a circle be drawn through P1, P3 and P5, (illustrated in red), it can be seen that this circle contains another point, P2, and so a triangle formed from these points would not form part of the Delaunay triangulation. Removing this red circle would leave us with four circles passing through P3, each of which contains no other points. This feature of a Delaunay triangulation: that the circles drawn through the three points of each triangle will contain no other points, is referred to as the 'empty circumcircle property'.

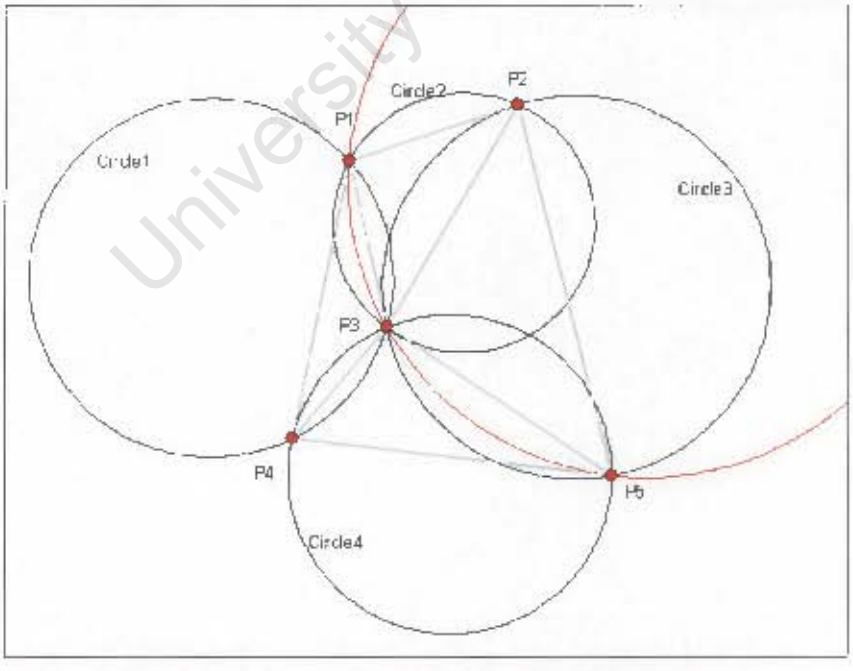


Figure 2.12 Circumcircles for the Delaunay triangulation

As mentioned previously, a Delaunay triangulation is closely related to a Dirichlet tessellation. This is because a Dirichlet tessellation can easily be created from a

Delaunay triangulation and vice versa. To create a Dirichlet tessellation from these circles, the centre points of all the circles passing through a point become the vertices of the Thiessen polygon for that particular point. For the points and circles in Figure 2.12, the centres of circles 1 to 4 will therefore become the vertices for the Thiessen polygon for P3 etc. This is illustrated in the Figure 2.13. This diagram shows how the area has now been divided into five regions: the first is the region nearer to P1 than to any of P2 - P5; the second is the region nearer to P2 than to any of P1, P3, P4 and P5 etc.

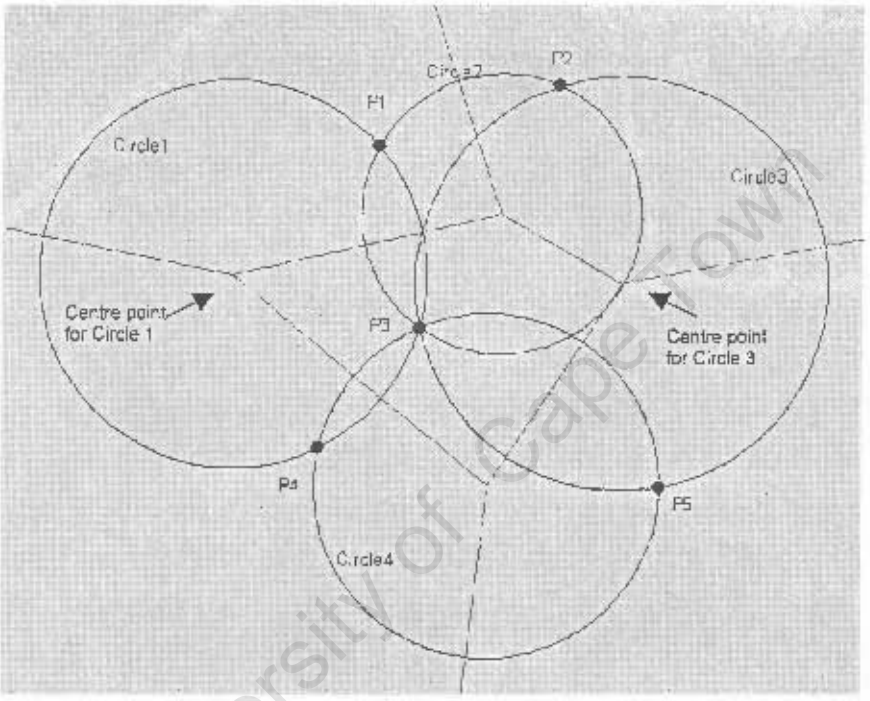


Figure 2.13 Thiessen polygon for P3

There are a number of algorithms which have been written to construct Voronoi diagrams. This 'empty circumcircle' property is used in several Delaunay triangulation and Dirichlet tessellation algorithms. One of these is called the Bowyer-Watson Algorithm which performs a triangulation. This algorithm is described and illustrated in Figure 2.14 as follows. The process is initiated by generating a supertriangle which is an artificial triangle encompassing all the points. At the end of the triangulation process any triangles which share edges with the supertriangle are deleted. Points are added in one at a time (Figure 2.14a). All the triangles whose circumcircles enclose the point to be added are identified and the outside edges of those triangles form an enclosing polygon (Figure 2.14b). The triangles in the enclosing polygon are deleted and new triangles are formed between the point to be added and each outside edge of the enclosing polygon (Figure 2.14c). This process is repeated until all points have been added [Bou89].

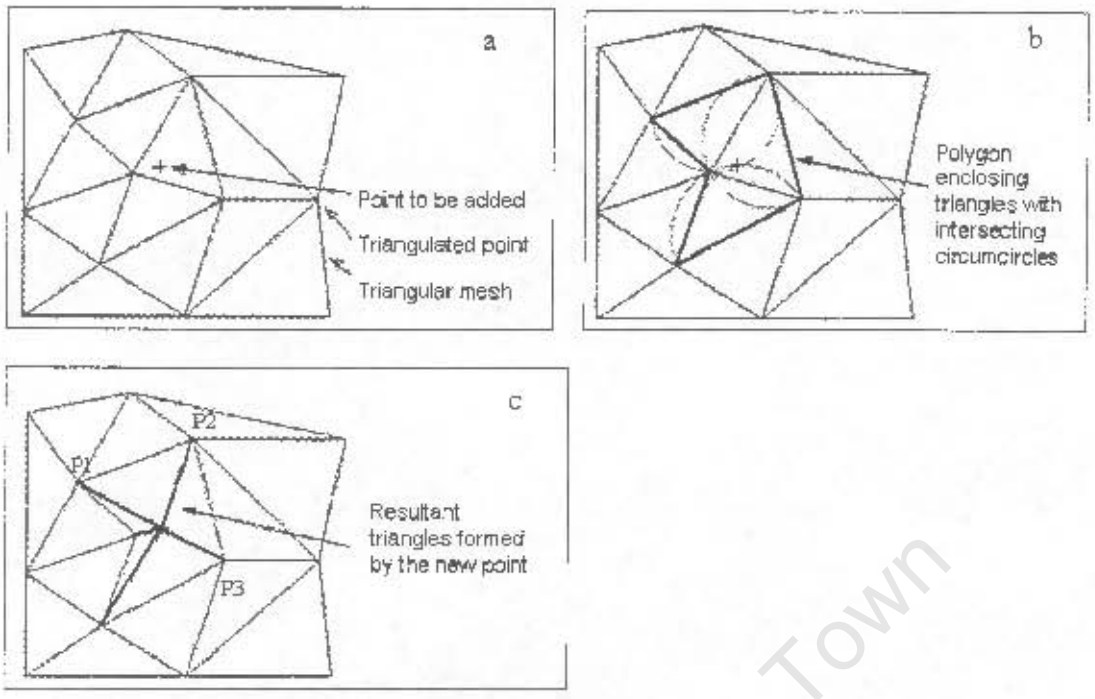


Figure 2.14 Process of creating a triangulation (Diagrams from [Bou89])

This algorithm forms the basis for the algorithm used to perform the tessellation for this project, but with a few slight changes. This is discussed in more detail in Chapter 5.

2.6 GIS Software - ArcGIS

A number of companies have developed both free and commercial GIS software, however the Environmental Systems Research Institute (ESRI) is the largest of these and its products the most extensively used worldwide [Hal04]. It has developed an integrated collection of GIS software products called ArcGIS which allows the deployment of GIS functionality in desktops, servers (including the Web) or mobile information systems. These different functionalities have been developed under the names Desktop GIS, Server GIS and Mobile GIS [Zei02].

2.6.1 ArcGIS Desktop Products

ArcGIS Desktop is a collection of software products that run on standard desktop computers. They are used to create, import, edit, query, map, analyse and publish geographic information. There are four products in the ArcGIS Desktop collection, each adding a higher level of functionality. The most basic is ArcReader which is a 'free viewer for maps developed using other ArcGIS Desktop products' [ADH04]. ArcView follows and 'provides extensive mapping, data use and analysis tools'. ArcEditor includes some advanced editing features and finally ArcInfo which is the

'full function, flagship GIS desktop product' [ADH04]. These products are relatively easy to use for non technical users with some training, have sophisticated tools for advanced users and can be easily customized by developers using industry-standard programming languages.

Each of these products contains a suite of integrated applications including ArcMap, ArcCatalog and ArcToolbox. ArcMap is the central application among these for all map-based tasks including cartography, map analysis, and editing. ArcCatalog is used to organize and manage all GIS information such as maps, globes, data sets, models, metadata, and services, and ArcToolbox contains tools for data management, data conversion, statistical analysis etc [ADH04].

New capabilities can also be added to these desktop products through a series of ArcGIS extensions from ESRI and other organizations. These extensions allow tasks to be performed such as raster geoprocessing (Spatial Analyst extension), three-dimensional visualization (3-D Analyst extension) and geostatistical analysis (Geostatistical Analyst extension), to name a few.

2.6.2 ArcGIS Releases

The first of ESRI's desktop GIS products was ArcView 3 first released in 1992. After several upgrades, this was followed by ArcGIS 8 where a few versions were released, the last being 8.3. ArcGIS 9 followed this, released in late 2004.

2.6.3 The ArcMap GUI

The ArcGIS products have comprehensive graphical user interfaces (GUI) with extensive menu systems as well as many command buttons which allow users to manually perform tasks. These exist for ArcMap, ArcCatalog and ArcToolbox, however the focus of further discussion will be on ArcMap as this is the application that is used in this project. An example of the GUI for ArcMap is shown in Figure 2.15 below.

Figure 2.15 Menus and tool bars on the ArcMap interface

The tools provided in these ArcMap menus and tool bars operate on the layers that are displayed in ArcMap. The results of any tasks performed on the layers will be seen directly after the operation, either in the visual representation of the layer itself or in its attribute table. Tasks cannot be performed on files that are only stored on a physical medium and not displayed in ArcMap.

2.6.4 Customizing ArcMap

It is also possible for users to customize ArcMap by adding custom commands, tools and menus, as well as by writing program code to create their own modules (small sections of code including declarations and procedures) or complete applications. Writing applications allows tasks to be performed on the data contained in files or raster images without having to actually display it in ArcMap, although they can also be performed on layers that are displayed.

Writing applications allows non-GIS experts to perform tasks without needing to know how to use the complex ArcMap interface. A command button can be added to the ArcMap GUI and this is used to run an application so that the user never needs to use any of the standard ArcMap GUI elements to perform the task.

Many operations on GIS data can be fairly complicated for inexperienced GIS users. Regardless of a user's experience, some tasks can also be very time consuming especially if it involves a number of different steps to obtain results or if the same operations must be performed repetitively on a large number of different datasets or files. Writing applications allows such tasks to be performed much quicker with minimal input required from the user.

In summary therefore, although using the ArcMap GUI to manually perform tasks may be the easier option for simpler tasks and trained users, writing applications provides an ideal way to automate tasks that are time consuming, complicated and involve many steps. They also make GIS functionality usable by the inexperienced.

2.7 ArcObjects

ArcObjects is the development platform for all the ArcGIS Desktop products. The ArcObjects software components expose the full range of functionality available in ArcInfo and ArcView to software developers and provide an infrastructure for application customization. ArcObjects is built using Microsoft's Component Object Model (COM) technology. The ArcObjects library is a comprehensive set of COM components consisting of over 1,000 classes and 2,000 interfaces. Using these classes

it is possible for developers to create a wide variety of customizations and custom applications to extend existing functionality provided in the ArcGIS applications.

2.7.1 Writing Code Using ArcObjects

In general, there are three ways to write ArcObjects code:

- As a Visual Basic for Applications (VBA) macro in an ArcGIS application.
- As an ActiveX COM component such as a DLL or OCX.
- As a standalone EXE.

Each of these, including some advantages and disadvantages of using them is discussed briefly below. This information has been taken from a publication entitled 'Exploring ArcObjects', written to assist developers in understanding and using ArcObjects [Zei02].

VBA Macros in an ArcGIS Application

VBA is embedded within ArcCatalog and ArcMap and can be accessed from their GUIs. Figure 2.16 below shows the ArcMap GUI with the menu selection to open the VBA editor.

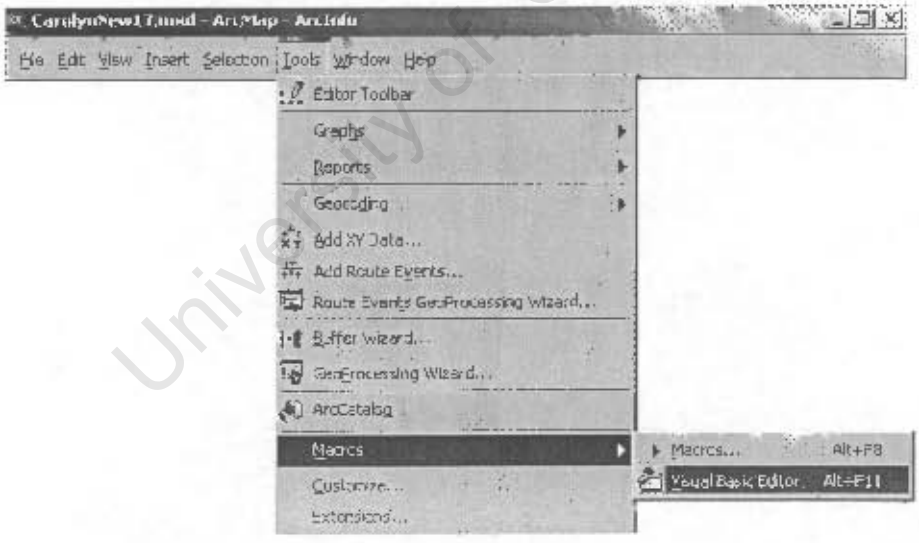


Figure 2.16 Menus to open the Visual basic Editor from ArcMap

VBA can be used to extend the ArcGIS products by adding modules and forms to create applications just as would be done when creating standard VB applications. VBA inside ArcGIS Desktop allows the majority of required customization with relatively little development effort and is the most common way that developers customize ArcGIS Desktop applications [Zei02]. Even if the final product will be

delivered to a customer in another format, writing code as a VBA macro in either ArcMap or ArcCatalog is ideal and provides a number of advantages:

- It is fast and easy to create, test, and debug.
- The standard ESRI type libraries are already referenced.
- It is simple to assemble GUI forms using VBA.
- It is straightforward to integrate VBA code with new ArcObjects UIControls.
- Many code samples available in the help system are macros that can be cut, pasted, and run within the VBA environment [Zei02].

Active X COM Components

Using a programming language other than VBA or wanting to package ArcObjects functionality into a COM DLL, EXE, or OCX, would necessitate working outside of the VBA development environment. This approach generally requires creating a project, referencing the ArcObjects type library, adding code and then compiling the source into a binary file.

Although these approaches allow the ArcObjects code to be hidden in a binary file and the functionality easily delivered to end users via custom setup programs, one of the main disadvantages of working outside of the VBA environment is that it necessitates the acquisition and use of another COM-compliant development tool. It is also often more difficult to debug the code.

GIS application maintenance is a problem in most small to medium institutions, like MCM. Their focus is on using the GIS software applied to their field of interest and not on software development. Should any code in the application need to be debugged or changed it would be far easier to do if the application was written as a VBA macro rather than an Active X Component.

Standalone Applications

Writing standalone applications with ArcObjects generally requires creating a project, referencing the ArcObjects type library, then assembling the required code to support the functionality of the application.

Although these applications hold some advantages in that it is possible firstly, to design highly customized user interfaces specific to the application and secondly, to create small, lightweight applications very quickly, they also hold a number of disadvantages. These include not being able to take advantage of the extensive functionality that ESRI has built into the existing ArcGIS applications such as

ArcMap or ArcCatalog; the need to provide a map display for visual applications; and not being able to use any of the extensions or take advantage of the components that give the ability to extend the existing ArcMap and ArcCatalog framework.

Although it is possible, the creation of standalone applications is not recommended if the desired functionality can be realized by extending existing ArcGIS applications. All ArcGIS applications share the same application framework, designed to be extended by third-party developers. Creating a standalone application will involve a significantly higher development effort and is only really appropriate for highly specialized implementations.

2.7.2 Developing with ArcObjects in the VBA Environment

The VBA Environment

The following discusses aspects of VBA development that are specific to the ESRI applications.

In the VBA development environment, modules, class modules, and user forms can be added to a default project contained in every ArcGIS application document. Modules and class modules contain a set of declarations followed by procedures, and user forms are containers for user interface controls. A project can consist of as many modules, class modules and user forms as is required.

ArcMap's default project is listed in the Project Explorer as 'Project' followed by its filename (e.g. CarolynNew17.mdx). In addition, another project is listed in the Project Explorer called 'Normal' (Normal.mxt). An example of a Project Explorer is shown in Figure 2.17. 'Normal' is a template for all documents and is always loaded into the document. Any modifications made to it will be reflected every time a document is created or opened. Modifications made to the Project however, will only be shown for the specific project.

Once the Visual Basic Editor has been invoked, modules, class modules, or user forms can be inserted as necessary. Procedures can then be entered in the item's Code window, where the code can be written, displayed, and edited. Figure 2.17 shows a number of forms (e.g. frmCompComplete, frmCompIntro) and Modules (e.g. AddColour, CheckDateField) that have been added to the CarolynNew17.mxd project.

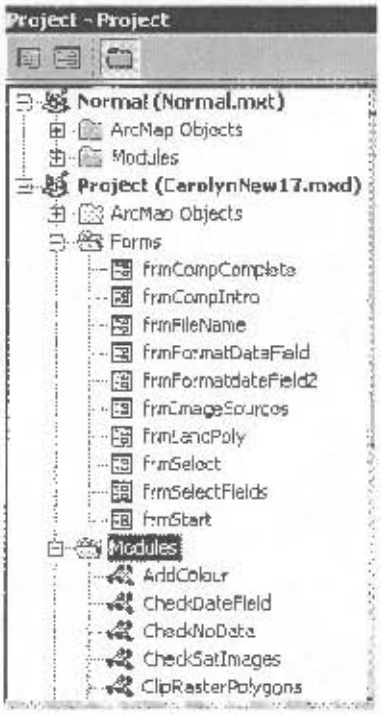


Figure 2.17 The ArcMap Project Explorer

VB Coding Standards and Guidelines

It is recommended that when writing applications, developers adhere to a number of VB/VBA standards. The relevant of these are discussed briefly below and have been used in the development of this project.

Naming standards

Modules, controls and data types should be named according to the function they provide together with the appropriate prefix [Zei02]. For example, forms are named with the prefix 'frm' (e.g. frmLoadData), labels with 'lbl' (e.g. lblInstruction) and Strings with an 's' (e.g. sImageName).

Variable declarations

'Option Explicit' should be used. This forces all variables to be declared before use, thereby preventing careless mistakes. *'Public'* and *'Private'* are used to declare variables at module scope and *'Dim'* used for declaration in local scope. Explicit types should be provided for variables, arguments and functions, otherwise they default to *Variant*, which is less efficient [Zei02].

Error handling

To ensure fault-tolerant code, regular error checking is strongly recommended. This is implemented by using the *On Error* statement, which allows control to be passed to a

single error handler in each function, which deals with all exceptional conditions that are likely to be encountered [Zei02].

Memory management

To ensure efficient use of memory resources, it suggested that all unused objects are set to *Nothing* to free up their memory.

2.7.3 ArcObject Interfaces

COM Classes and Interfaces

Developing with COM means developing using interfaces, the so-called interface-based programming model. All communication between objects is made via their interfaces. COM interfaces are abstract, meaning there is no implementation associated with an interface; the code associated with an interface comes from a class implementation. The interface sets out which requests can be made of an object that chooses to implement the interface [Zei02].

Within ArcObjects there are three types of classes of which the developer must be aware: abstract classes, coclasses, and classes. An abstract class cannot be instantiated; it is solely a specification for instances of subclasses (through type inheritance). ArcObjects' 'Dataset' or 'Geometry' classes are examples of abstract classes. An object of type Geometry cannot be created, but an object of type Polygon can. This Polygon object in turn implements the interfaces defined within the Geometry base class, hence any interfaces defined within object-based classes are accessible from the coclass.

A coclass is a class that can be publicly instantiated. In other words, it is possible for COM to create an instance of that class and give the resultant object to the client in order for the client to use the services defined by the interfaces of that class. A class cannot be publicly created, but objects of this class can be created by other objects within ArcObjects and given to clients to use [Zei02].

Important ArcObjects Interfaces

The purpose of this section is to illustrate the use of the ArcObjects classes and interfaces when developing applications. A few of the most important classes and interfaces that have been used in the development of this project are discussed briefly, including a list of some of their methods. The first section covers the use of shapefiles and the second the use of raster images. A shapefile is the name given to the file in which vector data is stored in the ArcGIS products. This information has been taken

from the ArcObjects Developers Help. An extract of working code for each section is shown in appendices A and B and illustrates the use of some of these interfaces.

Shapefiles

The *IWorkspaceFactory Interface* provides access to members that create and open workspaces and supply workspace factory information. A *WorkspaceFactory* maintains a pool of currently connected, active workspaces that are being referenced by the application. A *Workspace* is a container of spatial and nonspatial datasets such as feature classes, raster datasets and tables. It provides methods to instantiate existing datasets and to create new datasets

The *IFeatureWorkspace Interface* provides access to members that create and open various types of datasets and other workspace level objects. When working with shapefiles, if operations need to be performed on the features (points, lines, polygons etc.) themselves, then it is the shapefile's featureclass that is used, whereas for operations on the data contained in the shapefile, it is the shapefile's table that will be used.

Example methods:

OpenFeatureClass – Opens an existing feature class.

OpenTable – Opens an existing table.

The *IFeatureClass Interface* is the main interface for getting and setting properties of a feature class.

Example methods:

CreateFeature - Creates a new feature.

FeatureCount - The number of features selected by the specified query.

FeatureType - The type of features in this feature class.

The *ITable Interface* provides access to members that return information about and manage tables.

Example methods:

RowCount – The number of rows selected by the specified query.

GetRow – The row from the database with the specified object ID.

The *IFeature Interface* provides access to members that return and set properties of a feature.

Example methods:

Shape - A reference to the default shape for the feature.

Value - The value of the field with the specified index.

IGeometry Interface provides access to members that describe properties and behaviour of all geometric objects. The geometry of a feature can be obtained from a feature's shape. Coclasses of *IGeometry* include polygons and points.

The *IPolygon Interface* provides access to members that identify a polygon while the *IPoint Interface* provides access to members that define two dimensional points. The *IPointcollection Interface* provides access to members that manipulate the points of a Multipoint, Ring, Polyline or Polygon.

For performing geoprocessing operations on geometries (points or polygons) the *ITopologicalOperator* or *IBasicGeoprocessor Interfaces* can be used.
Example methods from *ITopologicalOperator:*
Difference - Constructs the geometry containing points from this geometry but not the other geometry.
Buffer - Constructs a polygon that is the locus of points at a distance less than or equal to a specified distance from this geometry.
Example methods from *IBasicGeoprocessor:*
Dissolve - Dissolves features.
Clip - Clips features.

Appendix A shows a section of code from my project to illustrate the use of some of these interfaces. The procedure creates a buffer area around the points in an input shapefile. A buffer is defined as a zone of a specified distance around a set of features [ESR05]. The geometry of the buffer area is returned from where it can be used to create a new shapefile or be operated on as is. The procedure includes creating appropriate workspaces, opening the featureclass, getting the geometry of the features, adding these to a collection and finally creating a buffer geometry using this collection.

Rasters
As is done when working with shapefiles, it is also necessary to create workspaces when working with raster images.
The *IRasterWorkspace* provides access to members that control a raster workspace.
Example method:
OpenRasterDataset - Opens a RasterDataset in the workspace given its name.

The *IRaster Interface* provides access to members that control an in-memory raster. It controls the reading of pixels from a Raster object.
Example methods:

CreatePixelBlock - Allocates a PixelBlock of requested size.
Read - Reads a block of pixels starting from the top left corner

The *IPixelBlock Interface* provides access to members that control a PixelBlock.
Example methods:
GetVal - The value for a specified pixel.
SafeArray - A variant SafeArray of pixels for a specified plane.

IRasterProps Interface provides access to members that control the most common raster properties.
Example methods:
NoDataValue - Data value used to indicate invalid or excluded data.
SpatialReference - SpatialReference of the Raster.

The *IExtractionOp Interface* provides access to members that control the extraction operations.
Example method:
Polygon - Extracts the cells of a raster based on a polygon.

The *IRasterGeometryProc Interface* Provides access to members that allow raster geometry processing.
Example method:
Mosaic - Mosaics the input rasters into a single dataset.

Appendix B shows the code for a procedure that takes two input raster images and creates a new image whose pixel values are calculated by averaging the pixel values of the two input raster images. Although there is a built-in function which performs this operation, it was found that the values it calculated for each pixel were incorrect and hence there was a need to write the procedure from scratch. The code illustrates the use of a number of interfaces used for raster processing.

2.7.4 Development Resources

A number of resources are available to assist developers. Some of these are mentioned later in this write-up and so a brief description of each is given below.

ArcGIS Object Model
This consists of a number of class diagrams which show the overall structure of the object model implemented by ArcObjects and are available as a series of PDF files. A small section of this Object Model is shown in Figure 2.18 below.

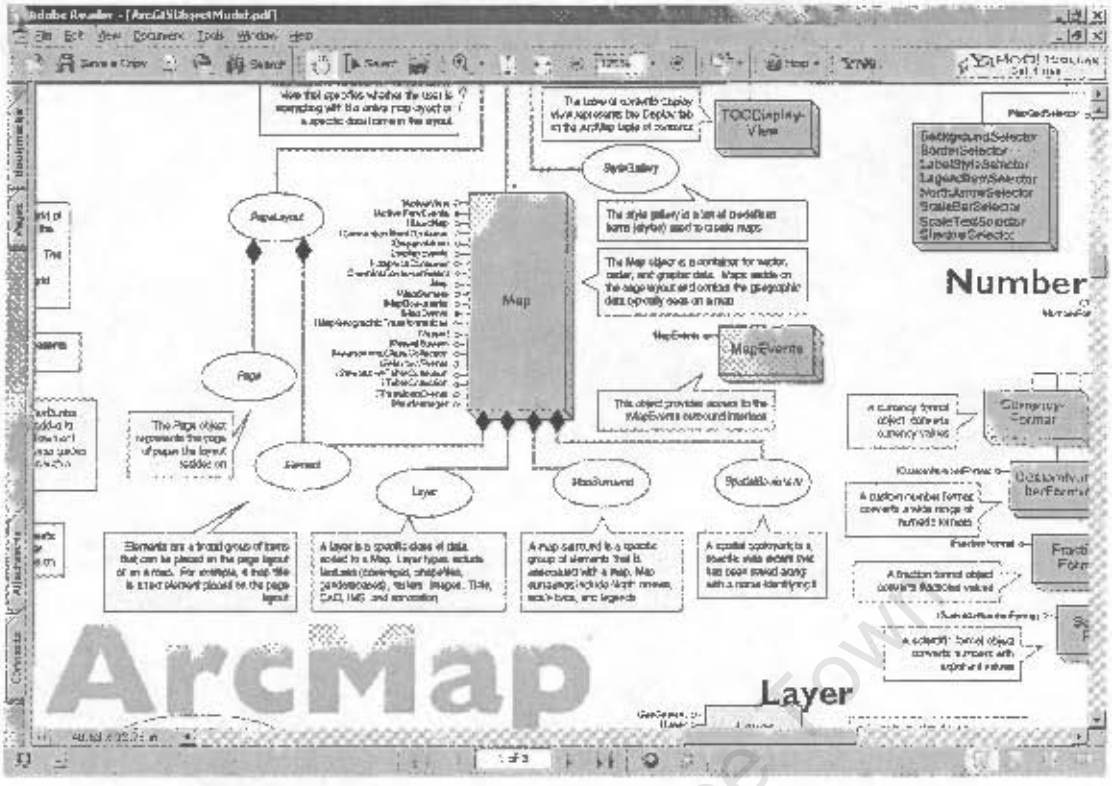


Figure 2.18 Extract from the ArcObject class diagrams

Object Browsers

In addition to the class diagram PDF files, the type library information can be viewed using a number of object browsers. Although Visual Basic has a built-in Object Browser (OLEView), the best object viewer to use is the ESRI object viewer. This has been customised for use with ArcObjects and was created to address certain limitations of standard object browsers [Zei02]. It can be used to view type information for any type library but defaults to the ESRI Object Library.

ESRI Forums

These are online forums hosted by ESRI where GIS users (both desktop users and developers) can browse and post focused questions and problems (http://support.esri.com/index.cfm?fa=forums.gateway).

ArcObjects Developers Help

The Developers Help contains detailed reference documentation about every class, coclass, interface, and enumeration within ArcObjects as well as sample code and technical documents. It ships with the ArcGIS software or is available online (http://arcobjectsonline.esri.com/).

2.8 Summary

This chapter started off by putting the project into context by giving a background to marine and fisheries research and some of the issues involved. It then discussed the importance of remotely sensed satellite images including how these can be used in a GIS. A short description of what a GIS is, was then given including how these images can be represented as raster data, while other data such as points sampled in the ocean from survey ships is represented as vector data. Details of a spatio-temporal composite image were given including the need to perform a tessellation around the data points, and the algorithm used for this. The focus of discussion then shifted to the actual GIS software that was used for the project. This included how it can be customized using ArcObjects, which is the development platform for the software. Finally a short description was given of some of the development resources that are available.

3 Project Requirements

In order to address some of the limitations that are encountered when applying RS satellite imagery to datasets that cover wide spatial areas and long time spans, the generation of 'spatio-temporal' composite images specific to individual datasets was investigated. This project aims to automate the processes of creating these composite images.

This chapter begins by describing exactly what a spatio-temporal composite image is and what steps are required to create these images. The problem caused by cloud cover in satellite images and how a solution to this problem will be incorporated into the product is then discussed. Finally, an overview is given of what the product is required to do and the features it will contain.

3.1 What is a Spatio-temporal Composite Image?

A spatio-temporal composite image is a composite image whose creation is based on both the temporal and spatial extent of a dataset. More specifically this involves applying the satellite image of the specific date on which data points were sampled or surveyed, to the total area sampled or surveyed on that date. For example, if a boat did a two-day survey of an area of coastline, this would involve obtaining the satellite image for the first day of the survey and applying that to the total area covered on the first day, and likewise obtaining the satellite image for the second date of the survey and applying it to the area covered on the second day.

Let us imagine, in terms of a GIS, two polygons, each surrounding the total area covered on one of the two days. For example, Figure 3.1a below shows the location of the data collection points on our example two-day survey. The points from the first day are shown in red and second day in blue. Figure 3.1b then shows two polygons surrounding all the data points for each day.

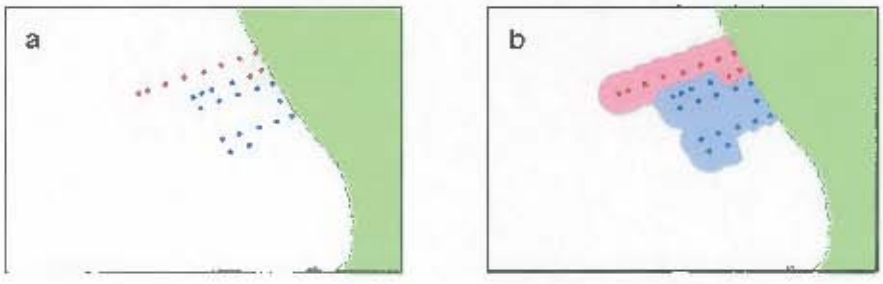


Figure 3.1 a) Data collected at points over two days shown in red for one day and blue for the next. b) Polygons representing the total area covered on each day

These polygons would be used to determine the extent that each of the satellite images would cover. This would be done by clipping these images with the polygons (cutting a piece of the image out using the polygon to determine the shape and location of the cut piece). These pieces of clipped images would then be merged into one image so that the resulting image, although appearing as a single image would actually consist of two pieces of different images. This in essence would be a spatio-temporal composite image: a single satellite image that would be generated to be used with a whole dataset regardless of the time span and spatial extent of the dataset while retaining the values from the original images.

The advantage of these images is that they prevent the smoothing of phenomena that are highly dynamic. The diagrams below illustrate this point. Figure 3.2 shows the data points for a survey conducted around the southern African coast. The survey lasted about a month and in the diagram each coloured polygon represents an area covered on a different day. If we wanted to map this survey with satellite images of sea surface temperature, ordinarily we might use monthly composite images (Figure 3.3a). Figure 3.3b however, shows the same area being mapped but using a daily spatio-temporal composite image. The difference in the two images is quite apparent: the monthly averaged image shows smoothing of the values where potentially important variability may have existed. For example, in the daily spatio-temporal composite there is a patch of very cold water (in blue), off the Cape peninsula coast as well as a strip of much warmer water (in orange) off the southern coast which is not apparent when the monthly composite is used.

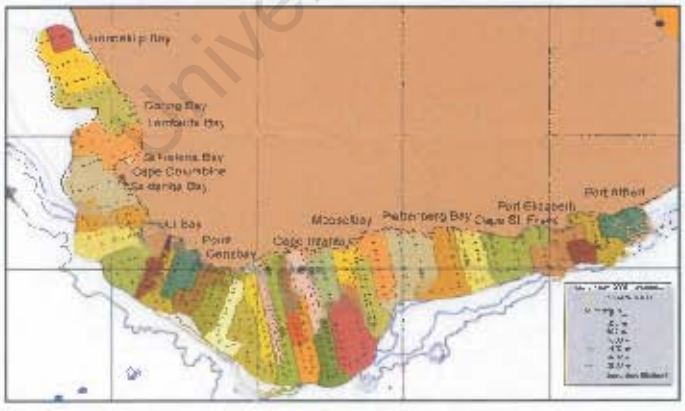


Figure 3.2 A coastal survey where the area covered for each day is represented by a coloured polygon

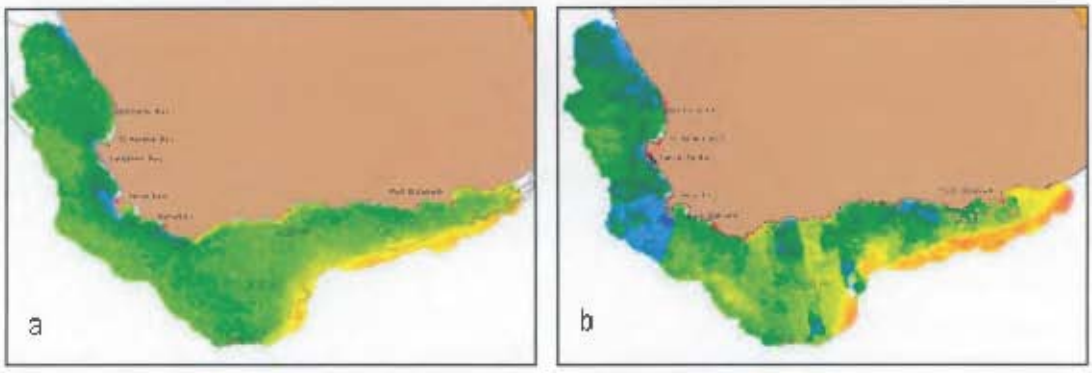


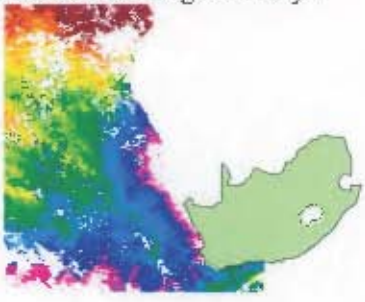
Figure 3.3 a) Monthly average SST b) Daily composite SST

3.2 Steps for Creating a Spatio-temporal Composite Image

This process of creating a spatio-temporal composite image can be broken down into a few well defined steps illustrated in the diagrams in Figure 3.4 that follow.

Step 1

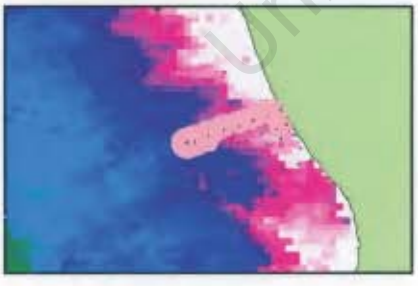
S1: Satellite image for Day 1



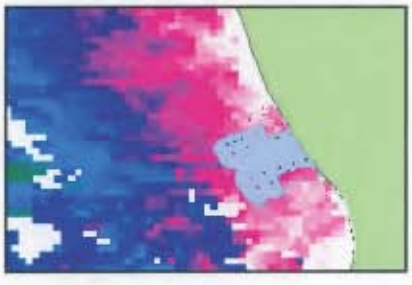
S2: Satellite image for Day 2

Step 2

Polygon for Day 1



Polygon for Day 2



Step 3

Required part of S1 for Day 1

Required part of S2 for Day 2

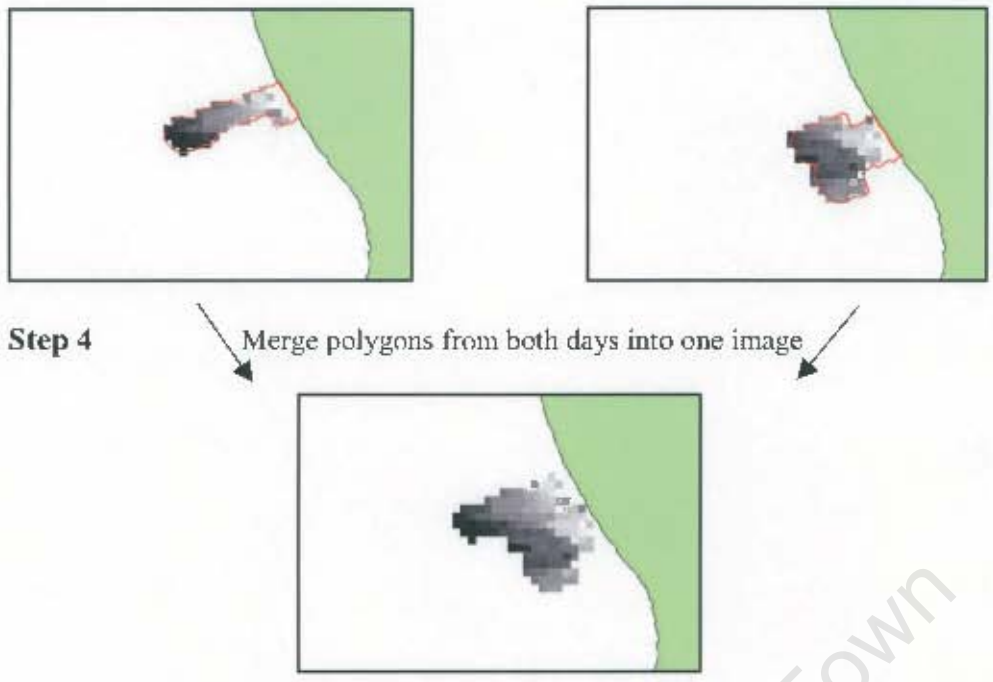


Figure 3.4 Steps for creating a composite image

The steps illustrated in Figure 3.4 can be summarised as follows:

Step1: Obtain satellite images for each day
Step 2: Get the polygon for each day
Step 3: Clip the relevant satellite image with relevant polygon
Step 4: Merge the image clips to create a single image

3.3 Dirichlet Tessellation

While the above may seem a rather straightforward process, there are a few complicating factors. The first of these is that most *in situ* data is represented by points while satellite data is represented as a continuous surface in a raster image. This means that it is necessary to find a way to associate these points and their surrounding areas with equivalent pixels or spatial areas on the RS images.

As mentioned earlier, ultimately we need polygons that represent the areas covered by each date of the dataset. The question therefore is how to determine the shape and extent of the polygon ensuring that each polygon represents the area covered by a particular day only and that polygons for different days do not overlap?

This can be done by creating a Dirichlet tessellation around all the points (defined in section 2.4). When creating these tessellation polygons it is necessary to set a limit for the outer lying ones. This limit is set using a predefined 'buffer area' which is created

around the points. The Thiessen polygons for a sample dataset are illustrated in Figure 3.5 below. Each polygon here represents a hypothetical area covered by each survey data point.

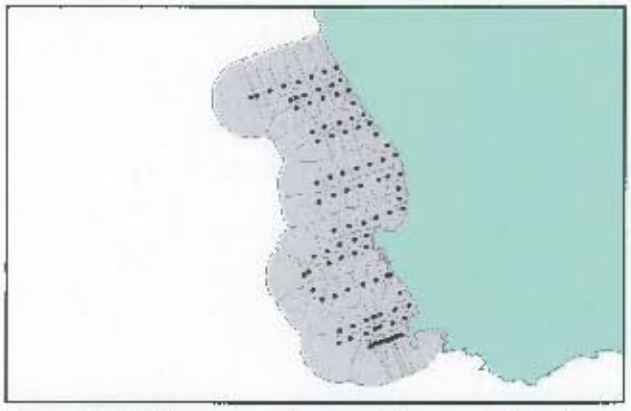


Figure 3.5 Thiessen polygons for a set of point

What is really needed however, is the area covered on each day of the survey, not just the area for each data point. If we look at Figure 3.6 below, the data points are displayed in different colours according to their date.

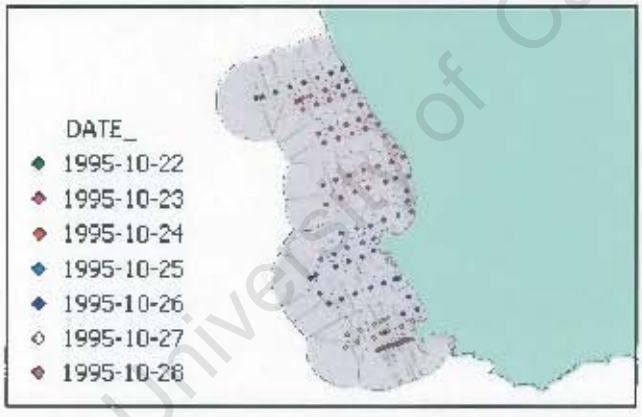


Figure 3.6 Data points for different dates displayed in unique colours

These dates can then be applied to the Thiessen polygons for each data point, so each polygon will have a date. Figure 3.7 below shows the polygons now displayed in different colours according to the date of the point around which they were created.

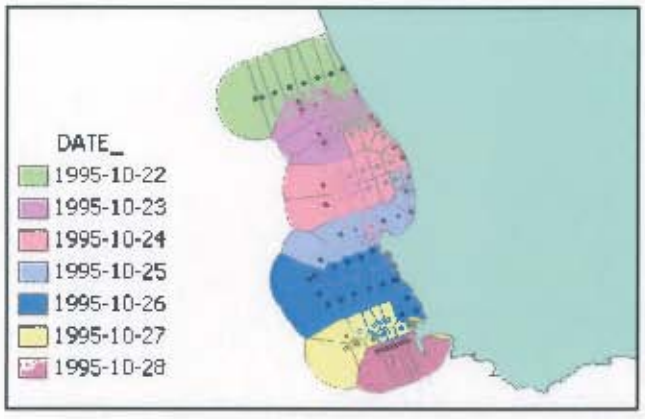


Figure 3.7 Thiessen polygons displayed in different colours according to their date

Finally, all polygons of the same date can then be dissolved to form a single polygon covering the total area for each date. Dissolving is the process of removing boundaries between adjacent polygons that have the same values for a specified attribute [ESR05]. This is illustrated the Figure 3.8 below.

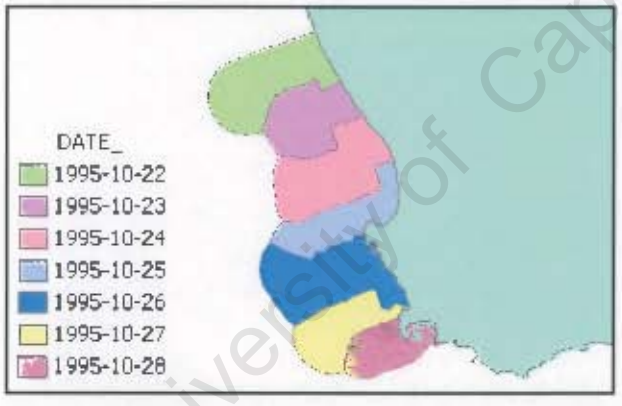


Figure 3.8 Thiessen polygons dissolved by date

These polygons, each with a date value, can then be used to clip the satellite of the corresponding date.

3.4 Problems with Cloud Cover

One problem with most satellite images is that data, for whatever parameter is being measured, cannot be obtained when there is cloud cover. Figure 3.9 below illustrates this problem, where all the white areas (pixels) have no data due to cloud cover.

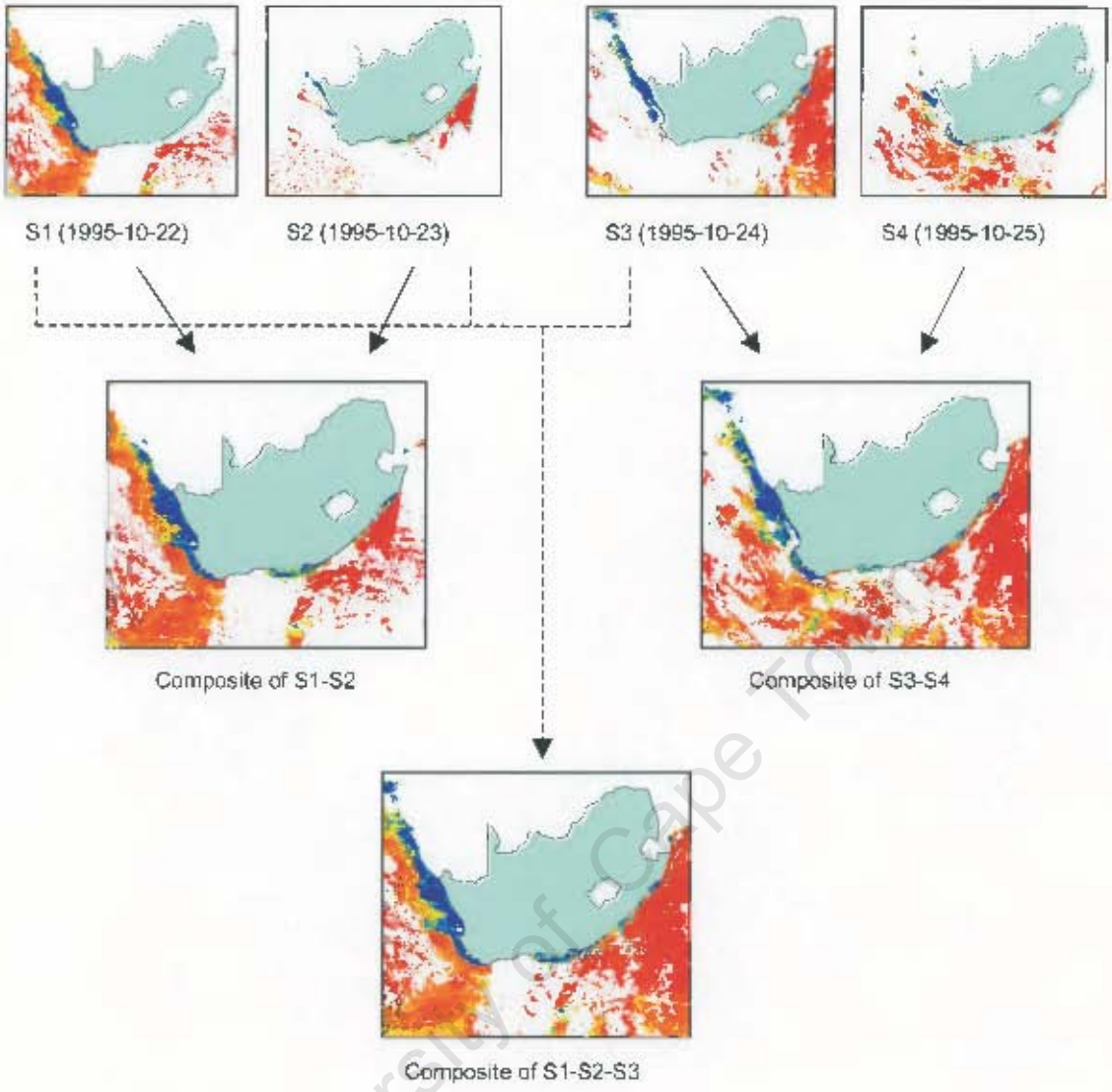


Figure 3.10 Creating temporal composite images

Given that we now have a way to decrease the cloud cover in the images, the question arises as to how this can be applied to the spatio-temporal composite images we are trying to create. The polygons that have been constructed are for each date while the images above are for every two dates. It is then simply a case of applying these images above to polygons for both dates. Referring back to the polygons we constructed earlier in Figure 3.8, the S1-S2 composite would be clipped using the polygons for 1995-10-22 and 1995-10-23, and likewise, the S3-S4 composite would be clipped using the polygons for 1995-10-24 and 1995-10-25.

To make this clearer, the polygons for each consecutive two days can first be dissolved together and this single polygon used to clip the above composites. This is illustrated in Figure 3.11 below. The 1995-10-22 and 1995-10-23 polygons are

dissolved to form a single 1995/10/22-23 polygon, which will then be used to clip the S1-S2 composite and likewise the 1995/10/24-25 polygon will be used to clip the S3-S4 composite.

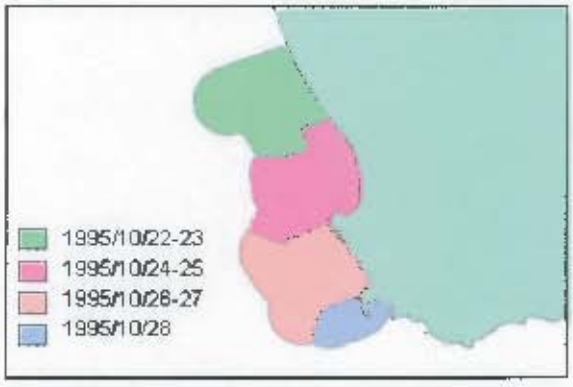


Figure 3.11 Polygons for two consecutive dates dissolved

Should the daily images contain no cloud cover then there is no need to perform this extra operation of creating the temporal composites. The benefit of this is that the original pixel values are retained and will be present in the final spatio-temporal composite image. It is likely however that there will be some cloud cover in the final composite image. If this is the case, then the option exists to try and decrease the cloud cover (increase the percentage of data pixels) by performing the temporal composition. The increase in the percentage of data pixels obtained by doing this will incur the cost of decreasing the accuracy of the pixels values (because they have now been averaged). As the temporal aggregation is increased and more days are averaged together, so the accuracy of the data will decrease. Ultimately it should be left to the user to decide what temporal aggregation they wish to use. The product is therefore required to give the user the option of generating the composite images at different temporal resolutions.

3.5 Overview of Product Requirements

It can be seen that the whole process of creating a spatio-temporal composite image which includes creating the polygons (including performing the tessellation), clipping these with the appropriate image and if necessary performing temporal aggregation of the images, involves a large number of steps. The example dataset illustrated in this section covered only seven dates. Other datasets may cover much longer time spans, possibly weeks and so would need much more processing to complete.

Creating these composite images manually using the ArcMap interface is quite possible, however it is clear that for data covering more than a trivial number of days this would be a very time consuming activity. It would also be required that the user

have a good knowledge of how to use ArcMap especially considering creating the images involves both operations on vector data and raster images.

Given that it is possible to easily develop applications using ArcObjects that can include all the functionality offered from the ArcMap interface, it became obvious that there existed a need to develop an application that could automate the process of creating these spatio-temporal composite images.

The aim of this project was therefore to do just that, develop a tool that could perform all the steps needed to create the composite image. Obviously some input would be needed from a user, however ideally this would be kept to a minimum.

The main requirement of the tool is therefore to create a composite image:

- for a given dataset of point vector data
- for a particular type of satellite image (SST/Chlorophyll etc)
- at a particular temporal resolution (daily, 3 days, 5 days etc).

A complete list of steps required to do this is as follows:

1) Obtain the input dataset.
2) Create a buffer area around the data points in the dataset.
3) Perform a tessellation on the datapoints using the buffer area.
4) Dissolve the Thiessen polygons created in the tessellation by dates.
5) Get the dates of the dataset.
6) Locate the correct satellite images (chlorophyll/SST) for these dates.
7) If necessary perform a temporal aggregation on these images.
8) Clip the images with the relevant dissolved polygons.
9) Merge the clips to create one composite image.
10) Calculate the percentage cloud cover in the final image.
11) Display the final image.

Although the tool will be developed to be used for the MCM data, the aim is to develop it as generically as possible, so that it can be used to create composite images for other sets of data.

If successful, the tool should:

- produce accurate, correct composite images
- create these composite images quickly
- be void of any bugs or errors
- be easy to use without training or experience with GIS's.

3.6 Summary

This chapter began by describing the concept of a spatio-temporal composite image. It was shown how using daily images to map environmental variables to *in situ* point data can prevent the smoothing of phenomena that occurs when using monthly composite images. This is because the values in the monthly composite image have been averaged, while using daily images retains the original values. The steps required to create a spatio-temporal composite image were then discussed including the need to perform a Dirichlet tessellation. This is required because the input data consists of points while the satellite images are continuous and there is a need to divide the total area that will be present in the composite image among the points so that all areas will be associated to exactly one date. In most satellite images cloud cover prevents data from being captured. Details were given as to how to overcome this problem with a spatio-temporal composite. Finally, an overview was given of the particular requirements for the tool, including all the steps that must be performed to create an image as well as the particular features required in order to consider the tool a success.

4 System Design

The main goal during the design phase was to design a product which would contain all the functionality needed to successfully create a composite image for a set of input data and parameters given by a user. One of the main issues during the design was to consider the format of the data at MCM so that the tool could run successfully, but at the same time attempt to design it so that it could potentially use data from sources outside MCM. This chapter discusses the various factors that were taken into consideration when designing the tool. This is followed by a detailed look at different design sections and ends with a brief look at considerations for designing the user interface.

4.1 Design Considerations

When designing the system, consideration had to be given to the format of any external data that would be used. This data would include the input point shapefile and the satellite images.

4.1.1 Format of Input Shapefile

A shapefile is merely a means to store vector data. The specific format that the data is stored in and what data is stored together in a single shapefile depends entirely on the user. An example is given below to illustrate this point.

Example:

Different surveys are conducted where a number of points are sampled over a few days. Each survey covers a specific area (e.g. South or West Coast) and is conducted from a specific research ship (e.g. P.R. Taylor, SailAway). Data collected at each point includes the SST, a count of a specific marine species, and the geographical position of each point (latitude and longitude). Some data for this example is shown in Table 4.1 below.

Survey	Date	Ship_Name	Area	SST	Count	Lat	Long
101	16/2/04	P.R. Taylor	South_Coast	24	32	31.521	18.254
101	16/2/04	P.R. Taylor	South_Coast	23	23	31.532	18.256
101	16/2/04	P.R. Taylor	South_Coast	21	28	31.534	18.260
101	17/2/04	P.R. Taylor	South_Coast	15	9	31.542	18.260
101	17/2/04	P.R. Taylor	South_Coast	14	5	31.555	18.262
101	18/2/04	P.R. Taylor	South_Coast	20	22	31.560	18.256
102	22/7/04	P.R. Taylor	South_Coast	19	12	30.870	18.321
102	22/7/04	P.R. Taylor	West_Coast	13	7	30.872	18.342
102	23/7/04	P.R. Taylor	West_Coast	14	2	30.900	18.342
102	24/7/04	P.R. Taylor	West_Coast	14	5	30.952	18.325
102	24/7/04	P.R. Taylor	West_Coast	15	6	30.962	18.341
103	9/10/04	SailAway	South_Coast	22	41	31.421	18.421
103	9/10/04	SailAway	South_Coast	21	24	31.432	18.432
103	10/10/04	SailAway	South_Coast	18	25	31.421	18.428
103	11/10/04	SailAway	South_Coast	18	30	31.324	18.433

Table 4.1 Example data for point shapefile

Different options exist for storing this data:

- Grouping this data by 'Survey' and storing each survey in a separate shapefile; the shapefile could be named using the survey number (Figure 4.1a).
- Grouping and storing the surveys together in a shapefile according to the 'Area', so there would be two shapefiles named 'South_Coast' and 'West_Coast' (Figure 4.1b).
- Storing all the surveys together in one shapefile, giving this shapefile a new name, for example 'Coastal_Surveys' (Figure 4.1c).

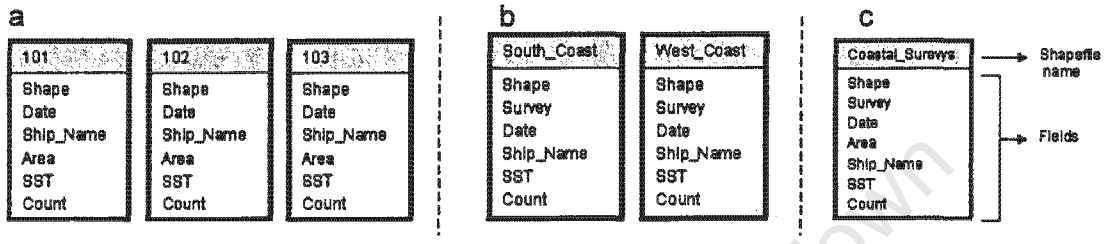


Figure 4.1 Data grouped and saved in shapefiles according to according to a) 'Survey' b) 'Area' c) all the data stored in a single shapefile

The reasons for choosing one method over another may vary. For example, if there are a large number of surveys it may be more convenient to store them in a single shapefile rather than having dozens of separate shapefiles each storing an individual survey. Storing them in a single shapefile does not mean the different surveys cannot be distinguished. Including the 'Survey' as one of the fields, allows the data for any selected survey to be extracted at any time. Likewise, including the 'Area' as one of the fields allows all the surveys for a specific area to be extracted at any time. Storing all the data in a single shapefile in effect means that the data can be grouped by whatever field is needed at the time. A user only has to select the field by which they want to group the data and then the value in this field in which they are interested.

The above example illustrates how the format of any input data can vary. The input shapefile may contain data that could not logically be grouped by anything. In effect it could contain a single dataset or data that could not be grouped by one or more fields. While in reality a shapefile where data within it can be grouped is still really a single dataset, for the purposes of this project it will from here on be referred to as containing 'multiple datasets'. This will facilitate easy referencing to such a shapefile.

If the shapefile contains multiple datasets, it is up to the user to decide which groupings of data within this shapefile they want to use. This will obviously depend on the specific objectives of the user's study or research. The user may only want to use the data that was collected for a specific coast, in which case they would have to

group the data by 'Area' and then select which area they want to use; they may want only data collected on a specific ship, in which case they would have to group the data by 'Ship_Name' and then select they ship they want, or they may want to use only data for a specific survey in which case they would group the data by 'Surveys' and then select the appropriate survey.

This example makes clear that when designing the tool, consideration had to be given to the format of the input shapefiles: they may contain single datasets in which all the data in the file would be used, or they may contain multiple datasets, in which case it would be necessary to get the user to first indicate by which field they would like to group the data and then which of the groups for this field they want to use as the input data for creating the composite image.

4.1.2 Deciding How to Locate and Check Image Availability

At some point in the process of creating the composite image it is necessary to check to see if there are images available for the required dates. If a user was creating a composite image manually they would simply provide the correct images. In automating the process it is somehow required to locate these images. One option was for the user, at some point in the process to actually provide and enter the names and locations of the images. The images provided would also have to be matched correctly to the required dates. This would probably entail the tool indicating a list of the required dates and the user matching up the images with each date or the user renaming each image by their date so that the tool could recognise each image as such. Since this could be a time consuming and tedious process, especially with large datasets, and since the purpose of developing the tool was to automate the process with minimal input from the user, this was not considered an ideal solution for this project.

An alternative solution was to set up a metadatabase which would contain information on all the available images including details such as the image names, dates and where they were stored on the computer network. As long as this metadatabase was kept up-to-date and accurate, it could then be used to check for and locate the required images. All that would be required would be to make a connection to the database every time the tool was run. Advantages of this method would be that no input would be required from the user and all the checking and retrieving of the correct images could be done automatically. The major disadvantages of using this method would be that the successful use of the tool would be bound to the presence of such a metadatabase and that the metadatabase would have to be kept up-to-date.

It was decided that the better of the two solutions would be to use a metadatabase to check for available images because this could be done automatically without requiring any input from the user. As it was, a comprehensive metadatabase had already been set up by the client in Microsoft Access, containing details of all the satellite images currently available at MCM. This metadatabase could therefore easily be used by the tool which was subsequently designed to perform queries according to the structure of the database.

4.1.3 Calculating the Percentage Cloud Cover

When viewing a satellite image, whether it is an original or composite image, it will be obvious to a user as to how much cloud cover there is in the image. 'No data' pixels are shown in a distinct colour, usually white and so are easily identifiable. Instead of a user simply being able to look at an image and subjectively gauging how much cloud cover there is, an extra feature to include in the tool would be to calculate the exact percentage of cloud cover. This may be useful for a user when deciding at what temporal resolution to create a composite image or, once the composite image has been created, whether the cloud cover was low enough or whether they want to try and reduce this by increasing the temporal resolution. When creating the composite images, it was therefore decided to include an operation to calculate the percentage cloud cover in each image.

4.1.4 What Temporal Resolution to Use

A decision had to be made as to how to go about determining how the user wanted to create the image: the user stipulating a temporal resolution they wanted to use or the user stipulating a maximum percentage cloud cover they wanted in the composite image for a particular dataset. Each of these would follow certain steps:

1) Stipulating temporal resolution:
User selects the starting temporal resolution > create the composite image > calculate percentage cloud cover > display percentage to user > if okay quit, if percentage too high let user choose another resolution

2) Stipulating a maximum percentage cloud cover:
User specifies maximum percentage cloud > create image for daily resolution > check percentage > if greater than specified, increase resolution > create image and check percentage > repeat until cloud cover is less than specified > display image with percentage to user

Before beginning the process of creating the image there is no way to determine how much cloud cover will be in each of the images and then the final composite image. If the user were to stipulate a maximum percentage cloud cover as the criteria for creating the image, at one end of the scale, if there is little cloud cover in the original images a daily resolution may meet the given criteria and the process would be quite quick. At the other end of the scale however, if there is lots of cloud cover in the original images, the resolution may have to be increased several times to reduce the percentage cloud cover to the required amount. This would mean that a number of composite images would have to be created before being successful. These operations take quite some time and so the user would have a fair wait for the process to be completed. Even at the highest temporal resolution there would still be no guarantee that the final image would have a percentage cloud cover below the stipulated maximum. Since the whole point of creating these composite images is to try and use the original values and not to use averages, the user may not want to use high temporal resolutions but rather have an image with more cloud cover.

Designing the system using the first option made more sense in that the user would be in full control of what temporal resolution would be used. Once the composite image is complete they can use the calculated percentage cloud cover to make a decision about whether to try again with a new resolution or keep the current image. The system was therefore designed so that the user would be required to stipulate the temporal resolution at the start. In most cases, the user would probably choose to start with a daily resolution and then increase this if necessary, so although the user would be given the option of starting at any resolution, the daily option would be set as default.

4.1.5 Making the Tool Generic

While many of the design decisions were based on the format of and what was required from the MCM data, at the same time the system was also designed in an attempt to make the product as generic as possible so that it could potentially be used with a variety of datasets.

The main feature that would make the tool more generic would be to allow any shapefile (as long as it has points with dates) to be used as input. The following features were identified that would ensure that the tool was generic:

- To let the user indicate the input shapefile, so any file can be used.

- To allow shapefiles that contain both single and multiple datasets to be used (as discussed in section 4.11).
- To let the user indicate the field that represents the 'date' so that this field does not need a standard name (can be called anything).
- To allow the tool to handle all date fields regardless of what data type the date field is defined as i.e. to be able to handle dates that are typed as Strings as well as Dates.
- To allow the tool to handle shapefiles where there are duplicate points.

4.2 External Data Sources

There are four external sources of data that will be used as input for the system:

1) A shapefile containing point data.
2) The remotely sensed satellite images.
3) The Access database containing metadata on all available satellite images.
4) User defined parameters.

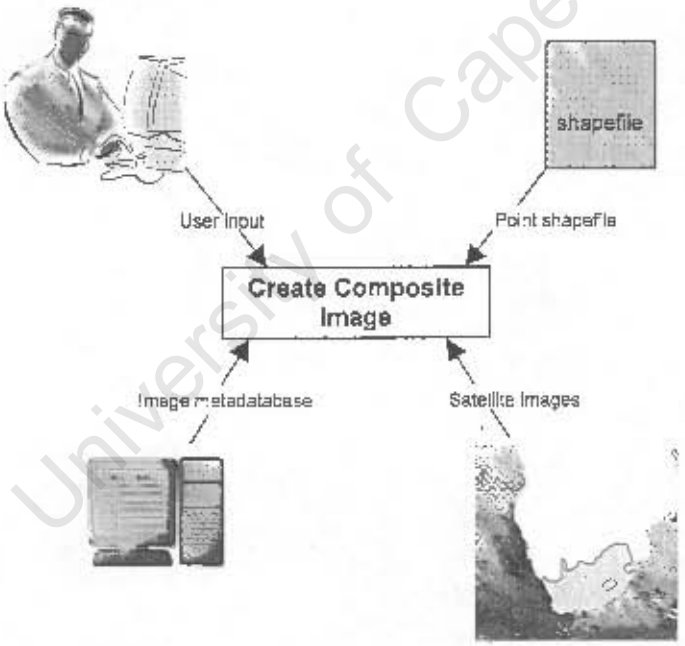


Figure 4.2 External data sources.

An overview of the first three sources follows, before going on to discuss the design issues in more detail. The input required by the user is discussed further in the next section.

4.2.1 A Shapefile Containing Point Data

As discussed in section 4.1.1, the system should be designed to accept shapefiles containing a single or multiple datasets. The only requirement of the shapefile is that

is contains point data and that each point has an associated date value. Should the shapefile contain multiple datasets, it would be necessary for there to be a field or fields that could distinguish the different groupings (datasets).

Regardless of what other fields are contained in the shapefile, this is the only information needed to create the composite image: the spatial location (which is contained in the 'Shape' field) and date of each data point. This means that the tool can be used for any other data, as long as it contains points which have a date attribute. The remainder of the fields which may contain information on biological or environmental parameters that were measured at each point are important for analysis or correlations with data from the final composite image, but are not needed to create the composite image itself.

4.2.2 Satellite Images

Remote sensing satellite images are obtained from various sources worldwide. The available images differ in the manner the data is obtained (i.e. the satellite and type of sensor used), the resolution of the image (i.e. size of the pixels), the size of the image (i.e. number of pixels), what parameter the image measures (e.g. SST, chlorophyll concentrations, wind directions etc.) and the algorithms used to convert the raw satellite data to values of the measured parameter.

The images themselves need to be correctly processed and formatted to be displayed in a GIS. At MCM there are currently four image sources available. These are:

- *Meteosat*. This is a geostationary weather satellite launched by the European Space Agency. It provides images of sea surface temperature (SST).
- *SeaWiFS*. (Sea-viewing Wide Field-of-view Sensor). Launched by NASA, providing images of chlorophyll.
- *NOAA*. (National Oceanic and Atmospheric Administration). Provides images of SST.
- *Ocean Space*. Located at the University of Cape Town. Provides images of chlorophyll and SST.

4.2.3 Image Metadatabase

As discussed previously, checking image availability would be done by means of the metadatabase. The layout of this database is illustrated in the Figure 4.3 below which shows the different tables, their relationship to each other and the fields in each table.

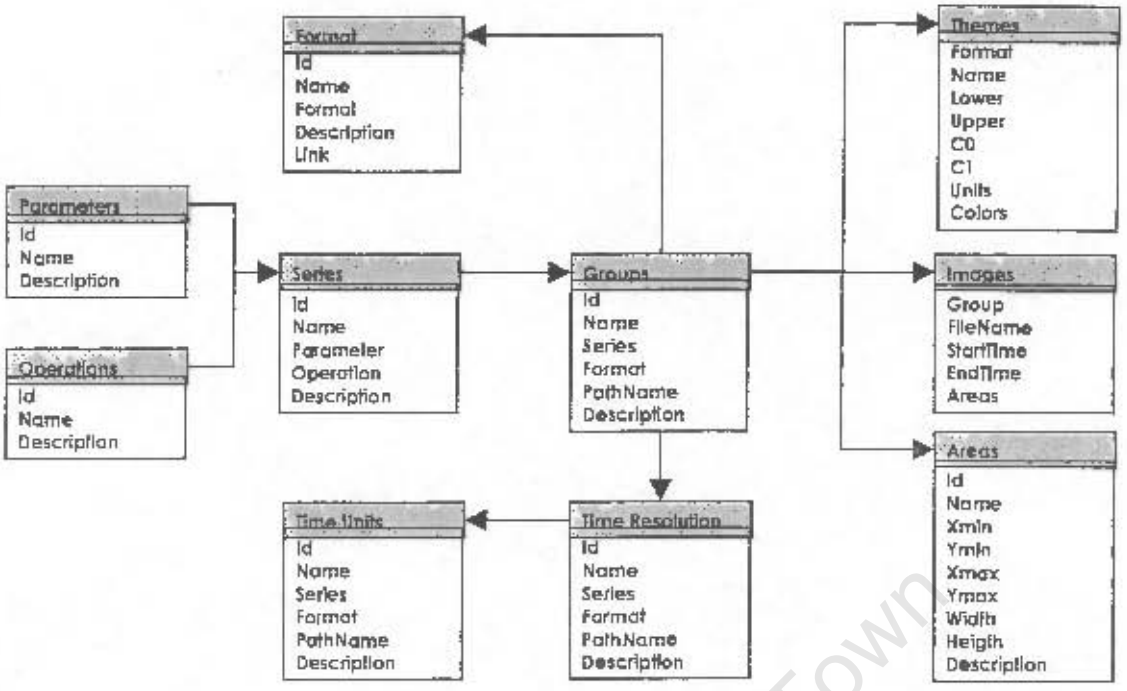
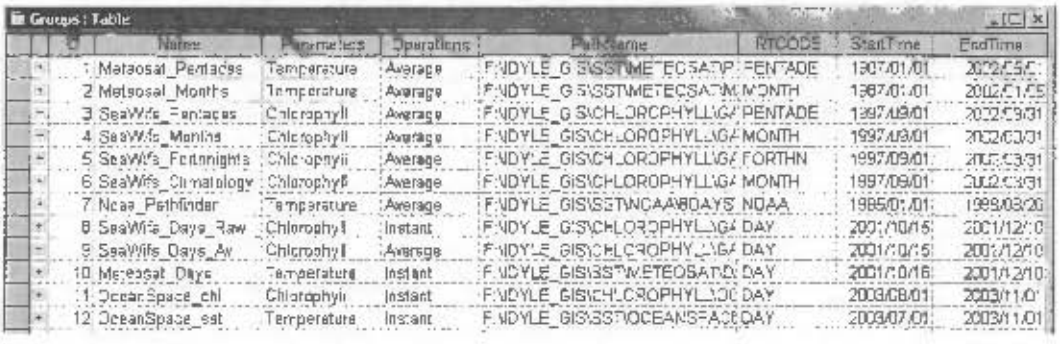


Figure 4.3 Layout of the MCM metadatabase showing the table names and fields

Information from only three of these tables is actually needed for the system. The first of these tables is called 'Groups' and it stores information on the different groups of images. The different groups are distinguished by the source of the satellite images, the parameter measured and whether the images in the group are daily or composite images. Figure 4.4 shows the records for this table. The relevant fields in this table are:

- *Name*: Gives the name of the group of the images (e.g. Meteosat_Days, OceanSpace_chl, Seawiffs_Days_raw).
- *Parameters*: Which parameter the image describes (Temperature or Chlorophyll).
- *PathName*: Gives the file path (e.g. F:\IDYLE_GIS\METEOSAT\DAYS) to where the group of images is saved.
- *RTCODE*: whether the images are daily, monthly etc (day, month, pentade).

It should be noted here that it is only those groups with an RTCODE of 'Day' that will be used by the tool as these images are all daily images. Groups with RTCODEs of 'pentad', 'forthn' and 'month' contain 5-day, 2-week, and monthly composite images respectively.



Groups : Table

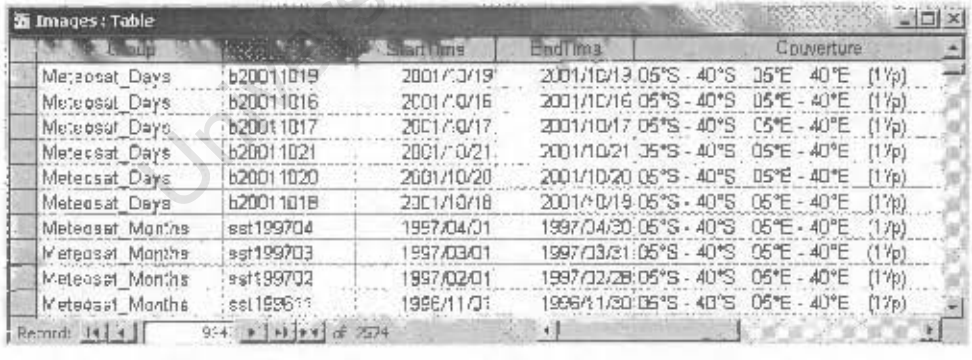
	ID	Name	Parameters	Operations	PathName	RTCODE	StartTime	EndTime
+	1	Meteosat_Pentades	Temperature	Average	F:\DYLE_G S\SST\METEOSAT\P	PENTADE	1997/01/01	2002/05/0
+	2	Meteosat_Months	Temperature	Average	F:\DYLE_G S\SST\METEOSAT\M	MONTH	1997/01/01	2002/01/05
+	3	SeaWifs_Pentades	Chlorophyll	Average	F:\DYLE_G S\CHLOROPHYLL\GA	PENTADE	1997/09/01	2002/09/01
+	4	SeaWifs_Months	Chlorophyll	Average	F:\DYLE_GIS\CHLOROPHYLL\GA	MONTH	1997/09/01	2002/09/01
+	5	SeaWifs_Fortnights	Chlorophyll	Average	F:\DYLE_GIS\CHLOROPHYLL\GA	FORTHN	1997/09/01	2002/09/01
+	6	SeaWifs_Climatology	Chlorophyll	Average	F:\DYLE_GIS\CHLOROPHYLL\GA	MONTH	1997/09/01	2002/09/01
+	7	Noaa_Pathfinder	Temperature	Average	F:\DYLE_GIS\SST\NOAA\8DAYS	NOAA	1985/01/01	1999/03/20
+	8	SeaWifs_Days_Raw	Chlorophyll	Instant	F:\DYLE_GIS\CHLOROPHYLL\GA	DAY	2001/10/15	2001/12/0
+	9	SeaWifs_Days_Av	Chlorophyll	Average	F:\DYLE_GIS\CHLOROPHYLL\GA	DAY	2001/10/15	2001/12/0
+	10	Meteosat_Days	Temperature	Instant	F:\DYLE_GIS\SST\METEOSAT\D	DAY	2001/10/16	2001/12/10
+	11	OceanSpace_chl	Chlorophyll	Instant	F:\DYLE_GIS\CHLOROPHYLL\OC	DAY	2003/08/01	2003/11/0
+	12	OceanSpace_sst	Temperature	Instant	F:\DYLE_GIS\SST\OCEANSPACE	DAY	2003/07/01	2003/11/01

Figure 4.4 Groups Table

This 'Groups' table has a one-to-many relationship with the 'Images' table which stores information about every image that is available in each group. Figure 4.5 below shows a selection of records from the Images table, including a list of images available from the 'Meteosat_Days' group and some from the 'Meteosat_Months' group. It can also be seen from the table that the 'Meteosat_Days' images, which are daily images have the same 'StartTime' and 'EndTime', whereas images from the 'Meteosat_Months' group, which are monthly composites, have 'StartTime' and 'EndTime' which indicate the first and last date of the composite image. Since the tool will only use daily images, it is only the 'StartTime' that is needed.

The relevant fields in this table are therefore:

- *FileName*: Name of the actual image (e.g. b20011019, sst19704).
- *StartTime*: The date the image was taken (e.g. 2001/10/01).



Images : Table

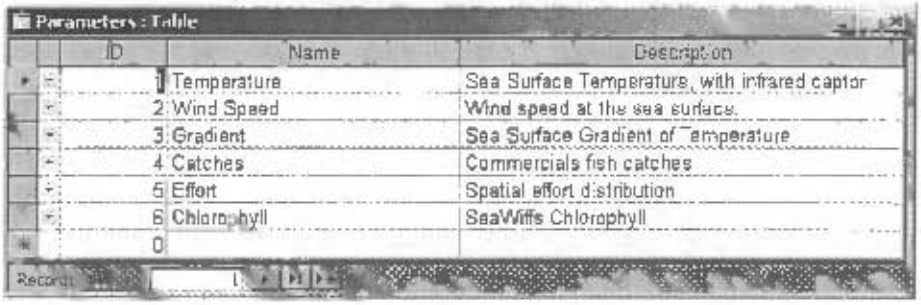
Group	FileName	StartTime	EndTime	Couverture
Meteosat_Days	b20011019	2001/10/19	2001/10/19	05°S - 40°S 05°E - 40°E (1/p)
Meteosat_Days	b20011016	2001/10/16	2001/10/16	05°S - 40°S 05°E - 40°E (1/p)
Meteosat_Days	b20011017	2001/10/17	2001/10/17	05°S - 40°S 05°E - 40°E (1/p)
Meteosat_Days	b20011021	2001/10/21	2001/10/21	05°S - 40°S 05°E - 40°E (1/p)
Meteosat_Days	b20011020	2001/10/20	2001/10/20	05°S - 40°S 05°E - 40°E (1/p)
Meteosat_Days	b20011018	2001/10/18	2001/10/19	05°S - 40°S 05°E - 40°E (1/p)
Meteosat_Months	sst199704	1997/04/01	1997/04/30	05°S - 40°S 05°E - 40°E (1/p)
Meteosat_Months	sst199703	1997/03/01	1997/03/31	05°S - 40°S 05°E - 40°E (1/p)
Meteosat_Months	sst199702	1997/02/01	1997/02/28	05°S - 40°S 05°E - 40°E (1/p)
Meteosat_Months	sst199611	1996/11/01	1996/11/30	05°S - 40°S 05°E - 40°E (1/p)

Record: 1 of 2574

Figure 4.5 A selection of records from the 'Images' table

The last table of importance is the 'Parameters' table shown in Figure 4.6 below. This has a one-to-many relationship with the 'Groups' table and gives details of the different Parameters. As can be seen from the table, there are six different parameters that have been included. The important fields in this table are:

- *ID*: This is used as the key to link to the 'Groups' table.
- *Name:* The name of the parameter (e.g. Temperature, Wind Speed).



Parameters : Table

ID	Name	Description
1	Temperature	Sea Surface Temperature, with infrared captor
2	Wind Speed	Wind speed at the sea surface.
3	Gradient	Sea Surface Gradient of Temperature
4	Catches	Commercials fish catches
5	Effort	Spatial effort distribution
6	Chlorophyll	SeaWiffs Chlorophyll
0		

Record: 1

Figure 4.6 Records in the 'Parameters' table

Using these three tables, SQL queries can be done on the data to determine if images are available for the required dates, and if so, what these images are called and where they are stored.

4.3 Design Sections

When designing the system, the operations involved in creating the composite image could be logically broken down into four steps:

1) Acquiring all the necessary information from the user. Preparing the input shapefile.
2) Checking image availability and obtaining the actual images.
3) Creating polygons which will be used to clip the images.
4) Raster (image) processing.

Steps 2 and 3 could be done in any order as the outputs of each are used together in step 4. This is illustrated in Figure 4.7 below.

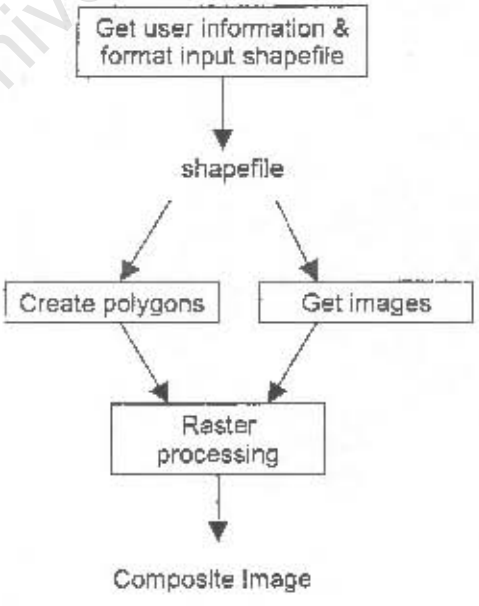


Figure 4.7 Main design sections for the tool

If however, it was found that there were no images available for the required dates, then the process could not be completed and creating the polygons would have been done for nothing. For this reason, the system has been designed to check the image availability first and only then to create the polygons once it is known that there are images available. Some of the main operations in each of these steps are discussed below.

4.3.1 Getting User Input and Preparing Input Shapefile

User information

The following information is needed from the user:

1) The name and location of the input shapefile.
2) Whether the input shapefile contains a single or multiple datasets.
3) A folder where the final image would be saved to.
4) The name and location of a shapefile with a land polygon.
5) The field representing the date.
6) The field indicating the different datasets (if multiple).
7) The particular dataset within this shapefile that is to be used (if multiple).
8) The type of satellite image to be used.
9) The temporal resolution.

Preparing the input shapefile

Using the information obtained from the user, a few operations need to be performed on the shapefile so that it can be used for further processing. The inputs and outputs of the step are illustrated in Figure 4.8 below.

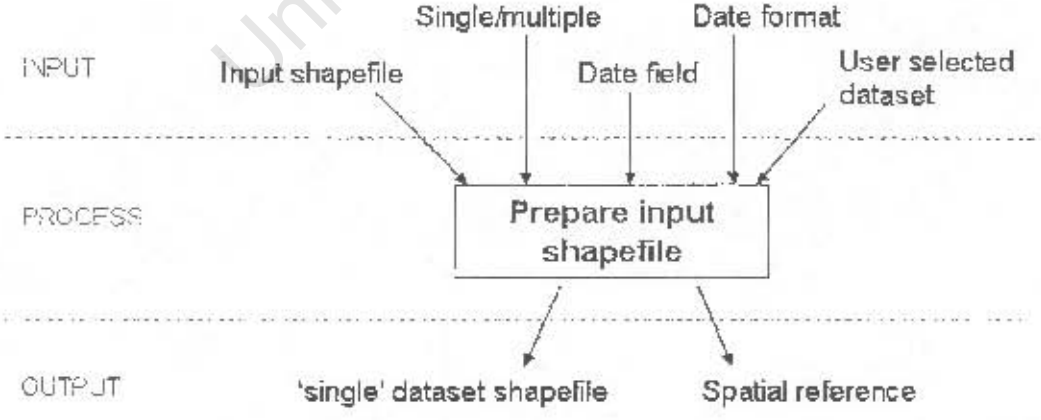


Figure 4.8 Input and output for the process preparing the input shapefile

There are three activities included in this section:

1) Obtaining a spatial reference from the input shapefile. When creating any new file in a GIS it is necessary to provide a geographical co-ordinate system so that it can be projected properly. Each raster and shapefile has a property called 'Spatial Reference' which contains details of the geographical co-ordinate system used to display data in the files. This spatial reference needs to be obtained from the original input shapefile, stored and used when necessary when creating new raster images or shapefiles.
2) If it is a multiple dataset, create a single shapefile in output folder. If it is a single dataset, copy shapefile to output folder.
3) Check the field type of the date field. If not a 'Date', then obtain the format of the date from the user and convert values to proper dates.
4) If file name is invalid or too long, get a new name from the user.

What should be noted at this point, is that once the creation of a composite image is complete, should the user opt to create a new composite image for the same data but at a different resolution, this whole step need not be repeated. The same shapefile with the formatted date fields can be used.

4.3.2 Checking Image Availability

The purpose of this section is to determine whether there are satellite images (of the user selected type i.e. SST or chlorophyll) available for the required dates; that is the dates of all the points in the shapefile. The inputs and outputs of this section are illustrated in the Figure 4.11 below.

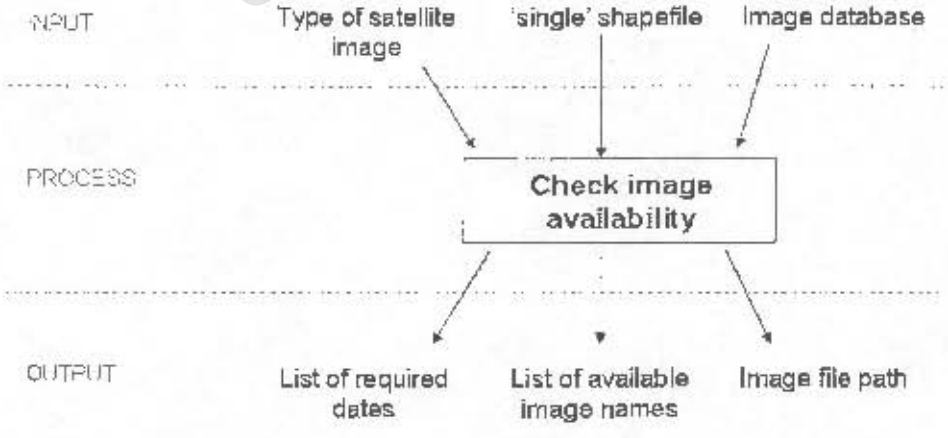


Figure 4.9 Inputs and outputs for the process of checking image availability.

This section is divided into three subsections:

1) Obtaining a list of required dates.
2) Making a connection to and querying the Access database containing metadata on all available images.
3) Using the results of this query to check if any available images match the required dates. If so, creating a list of these image names and obtaining the file path (if images from one group) or paths (if images from more than one group) to where they are stored.

In checking for available images there are therefore four possible outcomes which the system design needed to take into account. These are:

1) No images found for the required dates.
2) Images available for all the required dates and all these images belonging to the same group (see section 4.2.3. for discussion on different groups).
3) Images available for some of the required dates and all these images belonging to the same group.
4) Images available for the required dates, but images available from more than one group.

If no images are available for the given dataset then the process cannot continue and the user is returned to the beginning. These possible outcomes together with proposed actions for each are illustrated in the Figure 4.10 below.

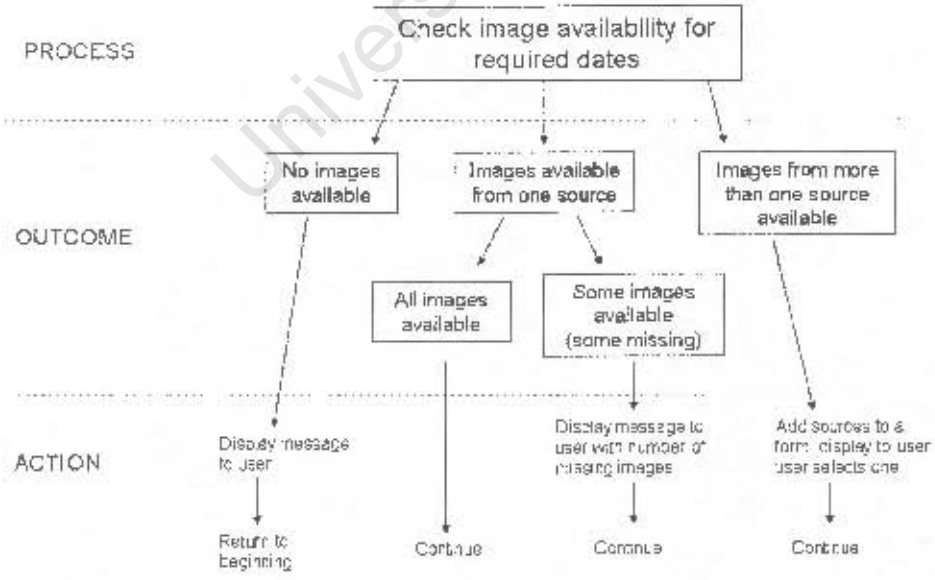


Figure 4.10 Potential outcomes and proposed actions when checking for image availability.

If any images are missing, certain operations would be affected by this and the system needed to be designed to handle these missing images. The operations which would be affected would be:

1) Averaging images, if the resolution is greater than daily.
2) Clipping the images with the relevant polygons.
3) Merging these image clips into a single image.

4.3.3 Creating Polygons for Clipping Images

As discussed earlier, what is required from the point shapefile is individual polygons each covering the area covered by the datapoints for a specific date. The inputs and outputs for this section are illustrated in the Figure 4.11 below.

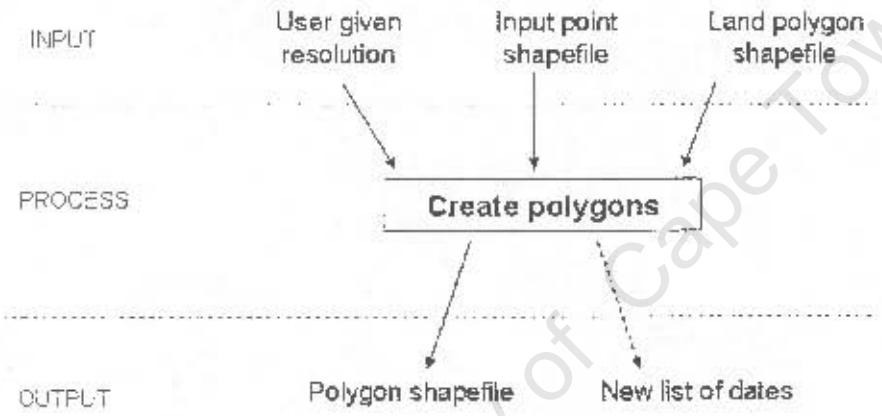


Figure 4.11 Inputs and outputs for the process of creating polygons.

The process of creating these polygons goes through a number of steps which are illustrated in the Figure 4.12. Each of these steps will produce a shapefile that will be used as input for the next step.

In summary, these steps are:

1) Create a buffer around the points.
2) Clip the buffer area with the land polygon.
3) Create Thiessen polygons from data points.
4) Perform a spatial join with Thiessen polygons to the data points.
5) Clip the Thiessen polygons using the buffer area.
6) Dissolve the Thiessen polygons by days.

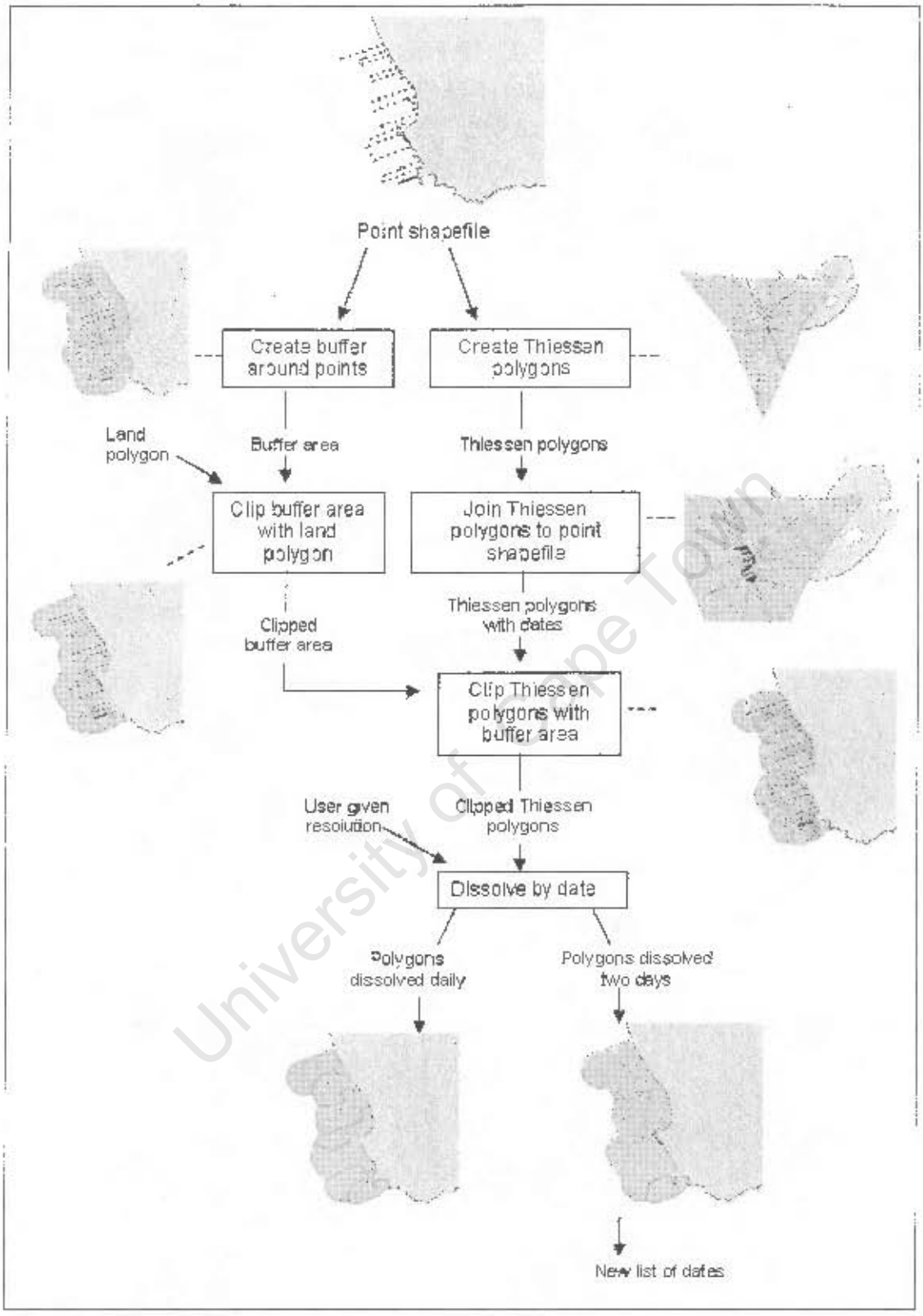


Figure 4.12 Steps for creating polygons from the input shapefile

4.3.4 Raster Processing

The last section in the system involves all the raster processing that is all the manipulating (clipping, averaging, merging) of the satellite images to create and display the final spatio-temporal composite image.

An overview of the step's inputs and outputs are shown in the Figure 4.13 below.

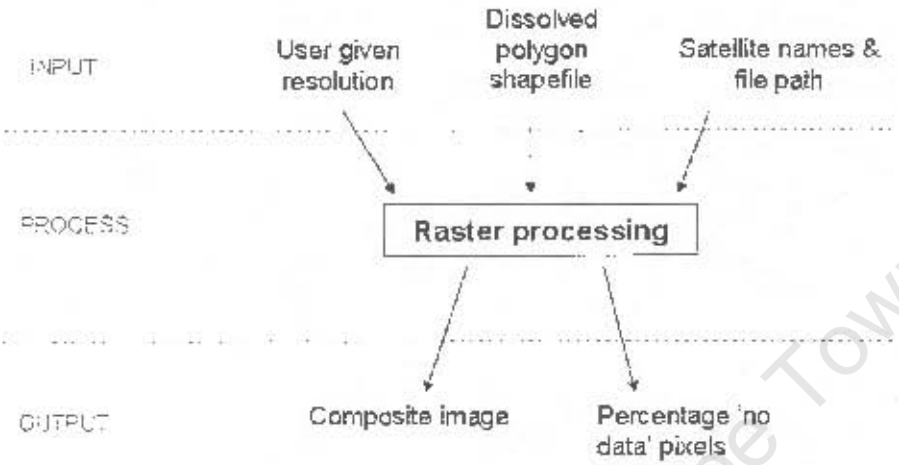


Figure 4.13 Inputs and outputs for the raster processing

The section can be divided into six subsections:

1) Averaging images if resolution is greater than daily.
2) Clipping images with the dissolved polygons.
3) Merging the image clips into one image.
4) Creating colour renderers for displaying the composite image.
5) Calculating the percentage cloud cover in the composite image.
6) Displaying a results screen.

Some of the operations would differ slightly depending on whether the temporal resolution was daily, or anything greater than daily. These are illustrated in the Figure 4.14 below.

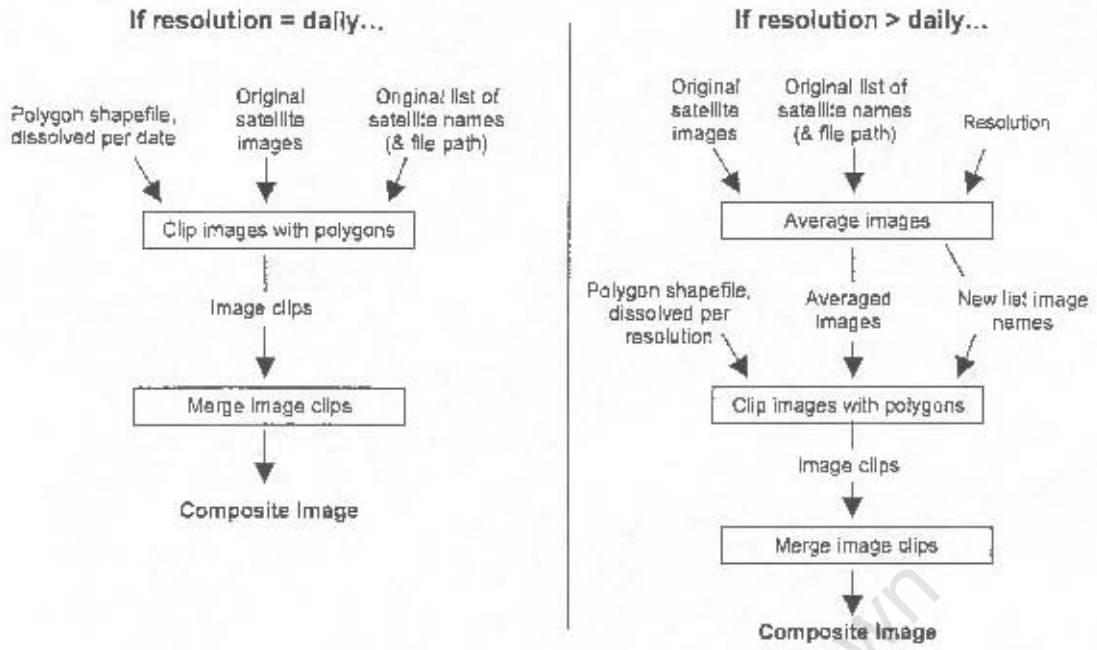


Figure 4.14 Final steps for creating the composite image when resolution is daily (left hand diagram) and anything more than daily (right hand diagram).

4.4 Interface Design

The design of the interface was basically determined by the information that was needed from the user. As was discussed previously this information is:

1) The name and location of the input shapefile.
2) Whether input shapefile contains a single or multiple datasets.
3) A folder where the final image would be saved to.
4) The name and location of a shapefile with a land polygon.
5) The field representing the date.
6) The field indicating the different datasets (if multiple).
7) The particular dataset within this shapefile that is to be used (if multiple).
8) The type of satellite image to be used.
9) The temporal resolution.

Two possibilities existed for the interface: putting all the required inputs on a single, large screen or dividing them up and putting them on a series of smaller screens. Some of the options that would be displayed for the user to choose would depend on previous information given by the user and which also then require some processing to determine the options. For example, the user must first indicate the input shapefile, from which the system obtains a list its fields which are then displayed to the user.

This means that the order that some of the information is obtained is fairly important and would best be done through a series of smaller screens. These inputs, together with the inputs where the order does not matter (e.g. location of the land polygon shapefile or temporal resolution), were divided into groups of inputs with similar functions, and each group displayed on a different screen.

When designing the layout of the screens the following features were considered as important:

- Keeping the screens simple, with not too much information on each.
- Keeping the design of all the screens consistent.
- Providing feed back to the user so that he may evaluate the results of his actions. For example, when the user indicates an output folder, displaying the name of this folder so that the user can check and see if it is correct.
- Allowing the user to return to previous steps (screens) if needing to change any input.
- Allowing the user to quit the program at any point.
- Displaying meaningful message when data is entered incorrectly e.g. incorrect character entered for the date format.
- Preventing the user from progressing to the next screen if they fail to enter any information. This includes displaying an appropriate message.

The layout and design of the screens that were used in the final product are shown and discussed in more detail in the next chapter.

4.5 Summary

One of the main goals when designing the tool was to make it generic so that it could be used to generate composite images for point data stored in any format in a shapefile. A number of features were included in the design to ensure this. These included allowing all the data or records or just a selection of records in the input shapefile to be used; to be able to use the relevant fields regardless of what they were named, and allowing the date field to be formatted in any way. One of the other main design considerations was how to go about checking to see if images were available for the required dates. The system was designed so that this information would be obtained from a metadatabase. This would allow this whole process to be done automatically without needing any input from the user. The main disadvantage to this method however, would be that it would require that the metadatabase be kept up-to-date. The whole process of creating a composite image involves a large number of different operations including creating a number of intermediate shapefiles and raster

images. Once a composite image has been created, and should the user wish to create another for the same input shapefile but at a different temporal resolution, many of these intermediate operations need not be repeated; the same intermediate files could be used. The system was therefore designed so that in such cases there is no unnecessary repetition of any operations, in effect meaning a faster processing time. Because some pieces of information are required to determine further options that will be given to the user, the interface was designed to consist of a series of small screens, each requiring a small amount of input from the user. This had the advantage that the screens could be kept simple, uncluttered and thus not overwhelming for users unfamiliar to the GIS software.

5 Implementation

This chapter describes how the design issues discussed in chapter four have been implemented. A brief overview of the development environment is given which includes the software and programming language used to develop the tool. The implementation of the user interface is then discussed. This is followed by details on the other three steps outlined in the design: checking image availability, creating polygons and performing all the raster processing.

5.1 Development Environment

5.1.1 Software Used

At the time this project began ArcView 8.3 was the software that was being used at MCM and so it was the logical choice of software to develop the application. Added to this, a new development platform called ArcObjects (discussed in section 2.6) was included with ArcGIS 8 which allowed for much easier customization than was present in ArcView 3, its immediate predecessor, which required coding in a language called Avenue.

It should be noted that since ArcEditor and ArcInfo contain all the functionality that ArcView has, the tool will run on these products as well. The only requirement is that they are version 8.3. In the latest release of the ArcGIS software (ArcGIS 9), there have been significant changes to many of the classes and interfaces and so the tool will not run successfully using this software. Likewise, there were changes from ArcView 8.2 to 8.3, and so (although this has not been verified) it is possible that the product will not run on releases prior to 8.3.

As there is raster processing involved, it is necessary to have the Spatial Analyst extension. If the software being used to run the tool does not have a license for Spatial Analyst the tool will not run.

5.1.2 Writing ArcObject Code

As discussed in section 2.6.1, there are three options for developing applications using ArcObjects: writing VBA macros, writing Active X COM components or using other programming languages and working outside the VBA environment, or writing standalone applications.

It was decided to develop the tool using VBA embedded in ArcMap. This was chosen primarily because it was recommended by ESRI as the best method to develop

smaller applications. Added to this, it would be easy for MCM to modify or extend the code should this become necessary.

5.2 User Input

In the previous chapter it was noted that before any processing can be done, a number of pieces of information are required from the user. It was decided that this information would best be obtained through a series of small, simple screens rather than a single, more complex screen. The graphical user interface (GUI) for the tool was therefore implemented as such. During earlier stages of implementation, a series of simple, prototype screens were constructed for the GUI using VB and used during the tool's development. Creating interfaces in VB is very simple. Windows are created from forms which act as containers for controls. Forms can be added when necessary and controls such as labels, textboxes, option buttons, command buttons etc. can be added to the forms by simply selecting the required control's icon from a toolbox and dragging it to the required position on the form. The interface for the tool was created in this manner.

Once development of the tool was near completion, a demonstration was given to the client of the application using this prototype GUI. The client was consulted about his desired layout of the final GUI including any changes he thought should be made on the prototype. His response was that he was quite happy with the GUI as it was and he did not propose any significant changes. The basic format of the GUI was therefore left as is.

A final step was to get other users to assist in the design of the GUI. A usability test was conducted which included gaining input from the users as to how they found the interface and any improvements they could suggest. Using these comments, slight modifications were made, after which the modified GUI was used in the final product. User evaluation is discussed in more detail in Chapter 7. The rationale for the user interface screens is presented in this section, along with implementation issues that input processing raised. The screens are shown in the order in which they appear when running the application.

5.2.1 Input File and Output Folder

The first screen, shown in Figure 5.1 below, requires the user to:

- Select an input shapefile.
- Indicate whether this file contains a single dataset (i.e. all data in the file will be used) or multiple datasets (i.e. only selected data in the file will be used).
- Select an output folder.

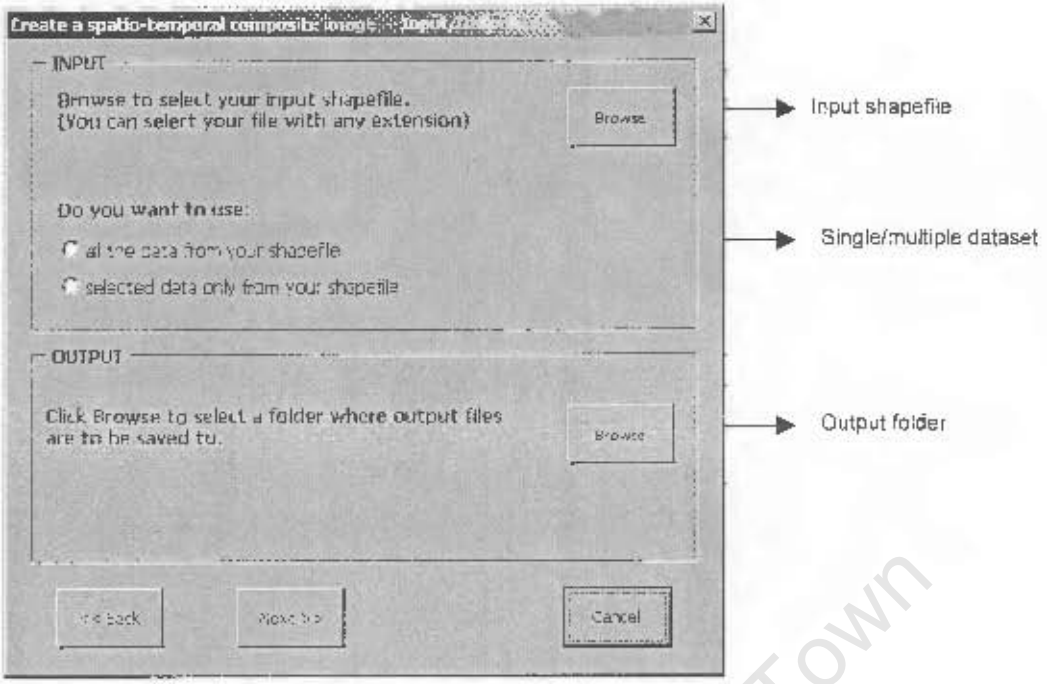


Figure 5.1 Screen 1: Input shapefile and output folder

Input shapefile

This has been implemented by allowing the user to browse for the relevant file using the Common Dialog of the Windows API [Kul00], shown in Figure 5.2. As can be seen, each shapefile consists of a number of files with different extensions (e.g. .shp, .shx). The user can select any one of these files for the shapefile by highlighting it and clicking the 'open' button.

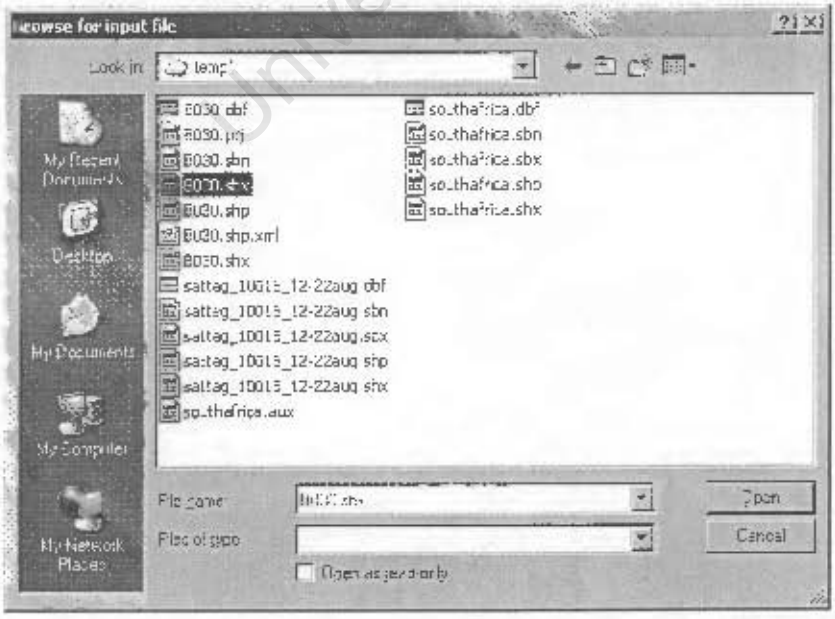


Figure 5.2 Common Dialog for browsing and selecting the input shapefile

The input shapefile provided by the user, must contain points which have a date attribute i.e. there must be a field which indicates the date on which the points were sampled. The only other requirement of this shapefile is that the points have been georeferenced correctly.

Once the shapefile has been selected, the user must then indicate whether this file contains a 'single' dataset so that all the data or records in the file will be used, or a 'multiple' dataset, in which case only a selection of the data or records will be used. If 'multiple' is selected, the user must indicate which particular dataset they want to use through screens that appear further on.

Output folder

As has been discussed, a folder is required in which to save the final composite image. Two options existed for this folder: using a fixed, pre-defined folder or allowing the user to select one. It was decided to allow the user to select the folder as it is more user-friendly. This was implemented using the shell32.dll, a library which contains Windows Shell API functions which are used when opening web pages and files [US05]. When the user clicks the 'Browse' button the window shown in Figure 5.3 appears.

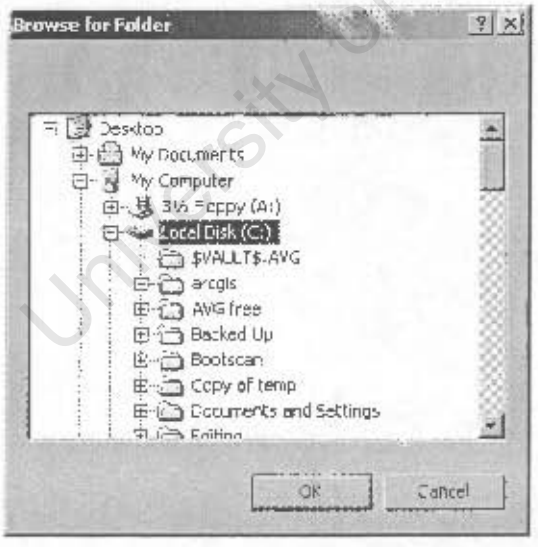


Figure 5.3 Window for browsing for the output folder

Should the user attempt to proceed without making a selection, a message will appear as in Figure 5.4. Similar messages appear in all cases where the user fails to enter required information.

Figure 5.4 Error message displayed when user fails to select an output folder

5.2.2 Land Polygon

In order to determine the extent of the tessellation and the composite image, a buffer area is created around the datapoints in the input shapefile. It is then necessary to use a polygon representing any land that lies near the datapoints to clip the buffer area ensuring that it only covers the sea. Polygon features could potentially be used in general to indicate areas to be excluded from the final image.

Initially the system was implemented to use a default land polygon of South Africa. After discussions with the client however, the system was changed to be more generic by allowing any polygon to be used. The screen where the user must indicate this polygon shapefile is shown in Figure 5.5 below.

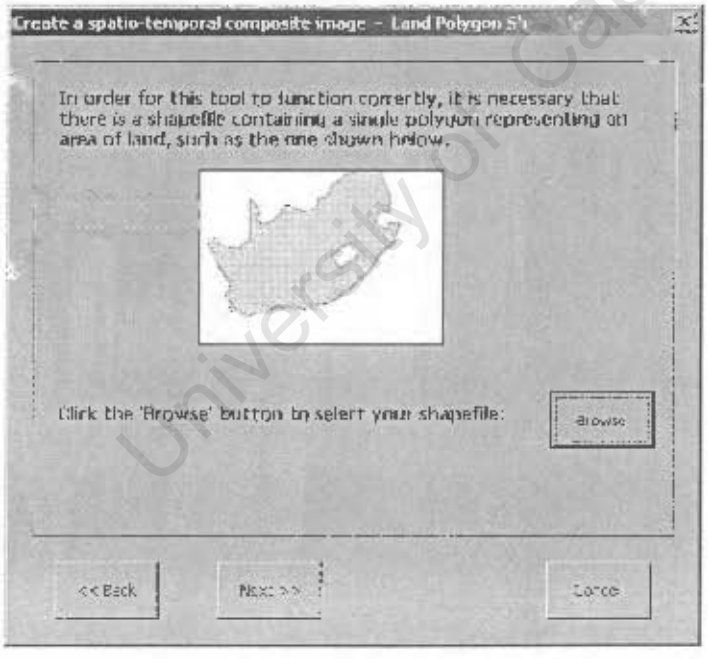


Figure 5.5 Screen 2: Land polygon shapefile

In the prototype screens, the user was given the option of typing the filename in a text box or browsing for the file. After a usability test was conducted (discussed in Chapter 7), it was decided to remove this option for the following reasons:

1) This option was rarely used.

2) Users struggled to enter the filename correctly (incorrectly entered spaces or capital letters which weren't there).
3) An easier alternative existed.
4) To keep the screens simple and uncluttered.

After the user has selected the file, a check is done to make sure the shapefile contains a polygon (rather than lines or points etc). If it is not, a message is displayed to the user.

It should be noted that the application has been implemented so that only a single polygon is used for the clipping operation. In most cases this will suffice as most land masses lying along coastlines will consist of a single body. Should data be collected around a group of islands, the individual islands can be represented as a single polygon in the shapefile and hence used successfully with the tool, while still being viewed as separate islands on a map.

It may also occur that a composite image needs to be created for data located away from the coast in deeper ocean waters or over land. Any polygon shapefile regardless of its spatial location can be used for such cases; the only requirement is that it does not lie in the same spatial location as the data.

In order to make it clearer as to what is required from the user here (i.e. polygon shapefile of nearby land), the client suggested adding a small diagram showing what such a shapefile should look like. A diagram showing a polygon of South Africa was therefore added to the screen for this purpose.

5.2.3 Field Names

Although most input shapefiles will probably have date fields that will be called 'Date', there is no guarantee of this. The date field may also not be distinguishable by its data type (as people often use String), or by the contents of the field (as one file may have several date fields e.g. 'date surveyed' and 'date checked'). It is therefore necessary to get the user to indicate the field in the shapefile which represents the date. Likewise, should the shapefile contain multiple datasets, it is necessary for the user to indicate which field distinguishes the different datasets.

From the input shapefile it is therefore necessary to obtain a list of all the field names so that the user can indicate these two fields. The *ILayerFields Interface* 'Provides access to members that work with a layer's fields' [ADH04], and this is used to

retrieve a list of the field names. These are displayed to the user in listboxes on a third screen as shown in Figure 5.6. If the shapefile contains a single dataset, the right hand listbox will not appear.

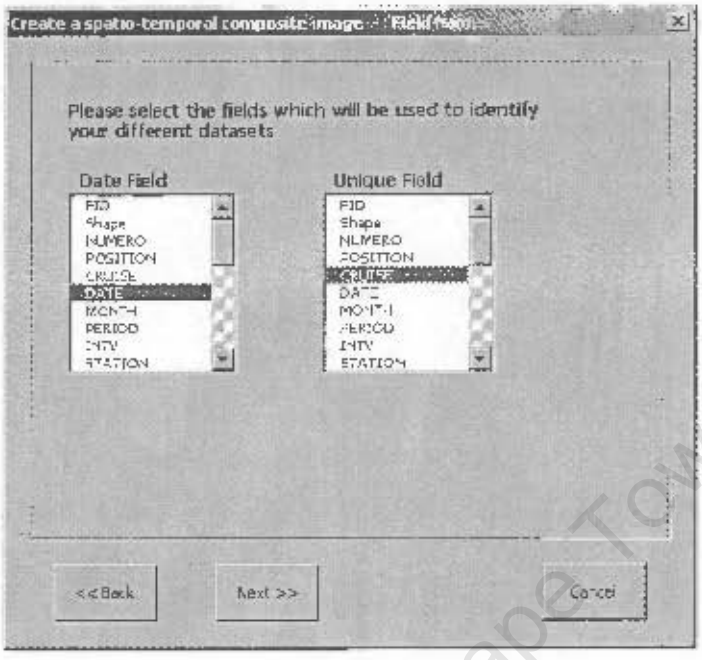


Figure 5.6 Screen 3: Field names

From the field indicated by the user to distinguish the different datasets, a list of all the unique values (datasets) in this field needs to be obtained so that the user can select the one value to use e.g. a particular cruise ID.

Initially this was done by using a cursor to move through each row in the shapefile's table extracting each new value. This process was extremely slow however, and a more efficient method was sought. Another possibility was to do a SQL query using the 'SELECT' and 'DISTINCT' functions. The problem encountered here was that this is done through the *IQueryDef Interface* which can only be used with geodatabase and personal geodatabase data sources [ADH04]. Since my application was working with standalone shapefiles and not a geodatabase, using these SQL queries was not possible.

After some research it was found that the quickest and most efficient method to do this would be to use the Scripting Dictionary which is an object of the Microsoft Scripting Runtime Object Library. This library contains objects that are useful for either VBA or script, and provide easy access to the file system making reading and writing to a text file very simple [MC05a]. The Dictionary object is a top-level object in the library and is used to store data key item pairs and returns the values (and

frequency if necessary) of any text or numeric field. By default, no reference to this library is set and so a reference to it must be set manually in VBA before it will run.

The list of datasets obtained is displayed in the lower the section of the fourth screen (Figure 5.7). The dataset the user selects here will then be used to create a new shapefile which is used for the reminder of the process.

5.2.4 Parameters

In the fourth screen the user must select:

- Type of satellite image (top section of screen)
- Temporal resolution (middle section)
- Dataset (lower section). This last section will only appear if the input shapefile contains multiple datasets.

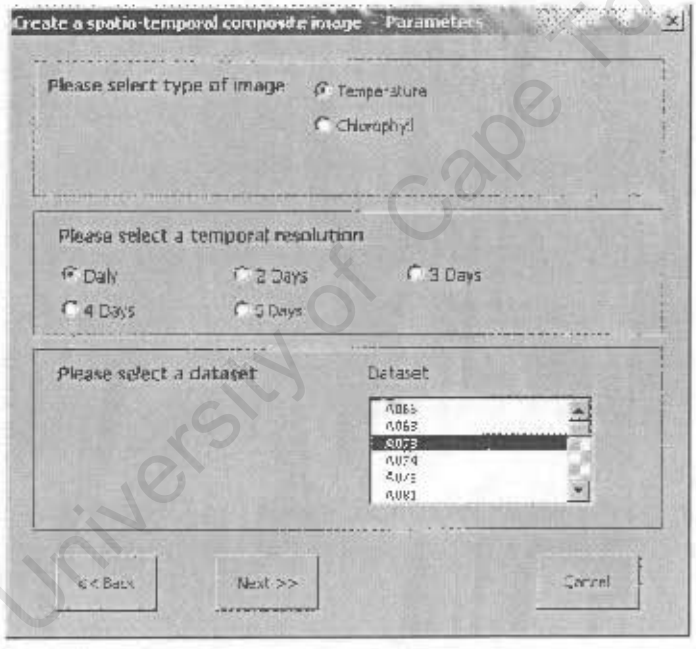


Figure 5.7 Screen 4: Parameters

Type of satellite image

The options given to the user here are limited to those for which images are available e.g. there is no point in giving wind direction as an option without images for this in the database. The tool first queries the metadatabase to determine what image types (parameters) are currently available and then give these as options to the user. This means that images for any parameters, such as wind speed or gradient, can be added to the collection in the future and used with the tool. This would simply entail adding relevant details to the 'Groups' and 'Images' table in the metadatabase.

Temporal resolution

The resolutions that were implemented were daily, 2-day, 3-day, 4-day and 5-day. Daily is set as default, as in most cases this would be the resolution of choice. The details of how these resolutions were implemented and why they were chosen is discussed in section 5.5.1.

5.2.5 New File Name

As indicated earlier, during the process of creating the composite image a number of intermediate shapefiles and raster images are created. In order to keep track of these, they are given the same name as the input shapefile, so for example if the input shapefile is called 'B030', the shapefile with a buffer area around the points in this file will be called 'B030_Buffer' and a clip of one of the images will be called 'B030_imageClip1'.

When saving raster images there is however, a restriction on the number of characters in the name; this can be no more than 13 characters. If a file name is too long use another screen is displayed (Figure 5.8), giving the name of the shapefile and requiring the user to enter a new name with a maximum of six characters (to allow for additional characters to be added to make up the 13).

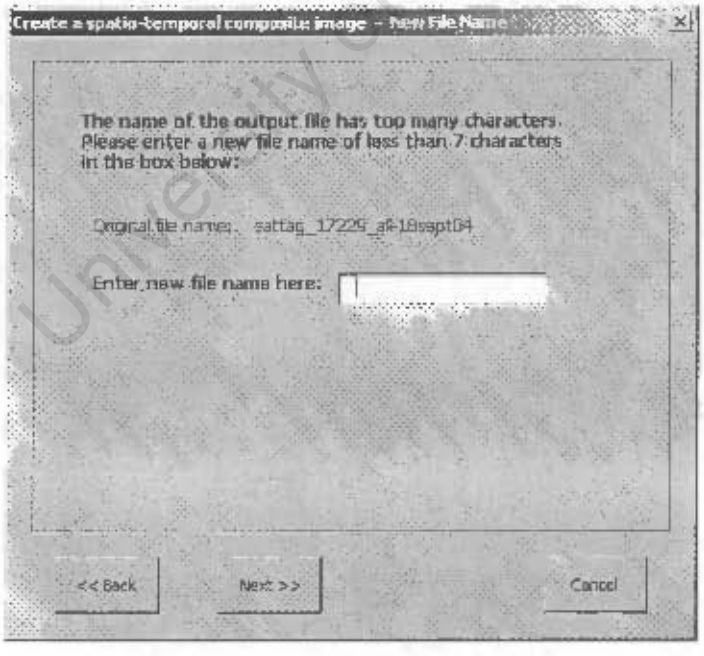


Figure 5.8 Screen 5: New file name

Intermediate files are saved in the same folder as the final image so that there is only one working directory. Should the user wish to view or use any of the intermediate files, they will then be easy to locate.

5.2.6 Formatting the Date Field

A problem regarding the dates cropped up when testing the tool on a wide range of datasets. In some shapefiles, a date value was defined as a 'String'. This meant that it was necessary to convert it to a proper Date format. Although there are ways to automatically distinguish the day, month and year values in such Strings, detecting this is complex and would require confirmation from the user anyway. It is therefore best to get this information directly from the user. An example date (i.e. a date value from the input shapefile) is therefore obtained and displayed to the user (Figure 5.9). The user must then indicate which characters in the example date are years, months and days. This is done by typing in a text box the positions of the years using 'Y', months using an 'M', days using a 'D' and any other characters with a '*'. So for example, if the example date is 02/05/04, and the date it represents is 2nd May 2004, the user would type in DD*MM*YY, and an example date of 24/07/2004 would be indicated by DD*MM*YYYY. This is subsequently referred to as the 'date format'.

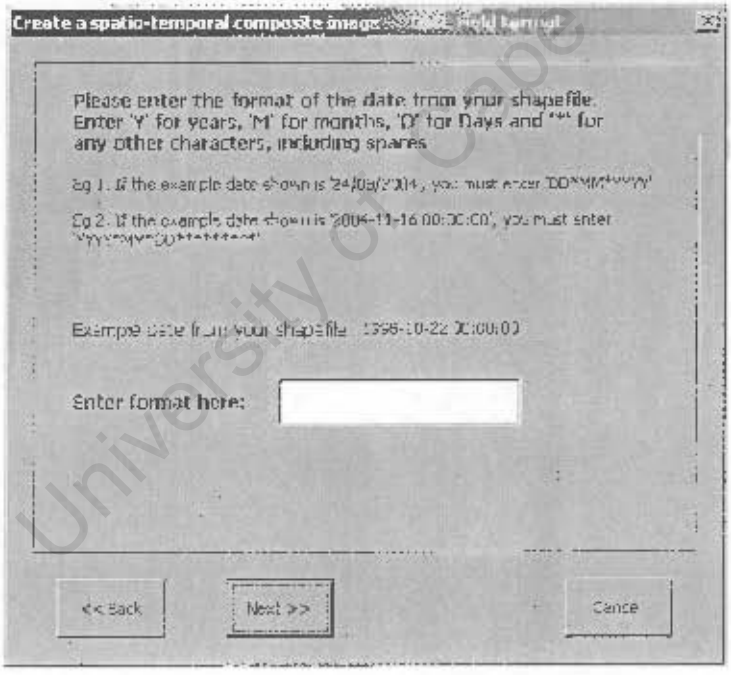


Figure 5.9 Screen 6: Ensuring that the 'date' field is correctly formatted

The reason why the user is asked to enter an '*' rather than a '/' is because some dates may be entered using dashes and may also include time values (e.g. 2001-10-23 00:00:00). To keep coding and instructions to the user simpler, it was decided to use a '*' to indicate all characters that were not years, months or days, rather than having the user enter different characters depending on the characters in the date.

Originally the user was asked to enter an 'X' to indicate non-date characters. After the user evaluation was conducted however, it was found that users had problems understanding and using this screen. In an attempt to improve this, the 'X' was replaced by a '*' and this used in the final GUI.

A new field typed as a 'Date' is created in the shapefile and the formatted date values placed here. At this point, all the information required from the user has been obtained and a shapefile with a correctly formatted date field has been saved in the output folder and is ready to be used in subsequent steps.

5.3 Checking Image Availability

5.3.1 Getting the Required Dates

As was done earlier to obtain a list of the different datasets, it is now also necessary to get a list of all the unique dates in the shapefile. Because the scripting dictionary that was used previously only returns text and numeric fields, this method cannot be used here. Instead, a cursor iterates over the rows and unique date values are stored in an array (called *SatDatesArray*). It is this set of dates that needs corresponding satellite images.

5.3.2 Connecting to and Querying the Metadatabase

Connecting to the database

As discussed in the previous chapter, details of all available satellite images are stored in an MS Access metadatabase to which a connection must be established from the application. Initially this was done by making a temporary connection each time the program was run using ADODB. This requires that the Microsoft ActiveX Data Objects 2.5 Library reference is set from within ArcMap and that 'Microsoft.Jet.OLEDB.4.0' is installed [MC05b]. In establishing the connection, it is necessary to stipulate the data source i.e. the path and database name e.g. 'Data Source=C:\temp\Images3.mdb'. This information was coded in and so could not be altered or chosen by the user.

For use of this tool for the purposes it was created at MCM, this is ideal. Should, however, the tool be used elsewhere, or the location of the database change, this would mean recoding the application. An alternative solution would be to create a permanent ODBC connection to the database on the computer but a user would have to manually set it up.

Querying the metadatabase

What is ultimately needed from this database is the name and file path of each image that will be used to create the spatio-temporal composite image. Once a connection has been established, a SQL query is used to retrieve the required information. Instead of making a query for each date that is required, one query is made which will return all the images which meet two requirements:

- *Parameter* e.g. SST or chlorophyll. Chosen by the user on Screen 4.
- *RTCode*. Because it is only daily images that are used, this will always be 'Day'.

The query returns a recordset which contains a list of all the available images which meet the above criteria. The required information that needs to be returned is:

- *Name* (image source)
- *PathName* (in order to locate the image)
- *FileName* (name of the actual image)
- *StartTime* (date of the image)

5.3.3 Checking Image Availability

The recordset returned by the SQL query must then be checked to see if there are any images that have dates (*StartTime*) matching those required by the particular dataset (and stored in *SatDatesArray*). Filenames of the available images are stored in a separate array (*SatNamesArray*) at the same position (same index value) as its matching date is stored in the *satDatesArray*. If no images are available for any of the required dates, a message is displayed to the user who is then returned to the start of the program.

It is possible that images may be available from more than one source for the required parameter. For example, if we look again at the Groups Table in Figure 5.10b, it can be seen that there are two groups of images 'OceanSpace_chl' and 'SeaWifs_Days_Raw', that both have daily images of 'chlorophyll'. Because the images from different sources may have different pixel sizes and spatial extents, they cannot be used together to create a single composite image. When checking for image availability for the required dates, it is therefore necessary to distinguish the source of the images.

Groups : Table

	1	Meteosat_Pentades	Temperature	PENTADE
	2	Meteosat_Months	Temperature	MONTH
	3	SeaWifs_Pentades	Chlorophyll	PENTADE
	4	SeaWifs_Months	Chlorophyll	MONTH
	5	SeaWifs_Fortnights	Chlorophyll	FORTHN
	6	SeaWifs_Climatology	Chlorophyll	MONTH
	7	Noaa_Pathfinder	Temperature	NOAA
	8	SeaWifs_Days_Raw	Chlorophyll	DAY
	9	SeaWifs_Days_Av	Chlorophyll	DAY
	10	Meteosat_Days	Temperature	DAY
	11	OceanSpace_chl	Chlorophyll	DAY
	12	OceanSpace_sst	Temperature	DAY

Figure 5.10 Groups table from Image database

Should available images for the required dates all originate from the same source, then these images are simply used. If some images are missing, a message is displayed to the user indicating the number of images missing and the remainder of the program will function as normal except that the final composite image will have gaps for the dates that were missing.

Should there be images available from more than one source, the image names for each source are stored in separate arrays. This information is then added to a screen (Figure 5.11) indicating the different sources as well as the number of images missing for each. The user must then select which source they want to use.

The outcome of this step is a list of image names stored in an array, which match the required dates. As all the images come from the same source, they will have the same file path, and it is merely a case of retrieving this file path from a single record.

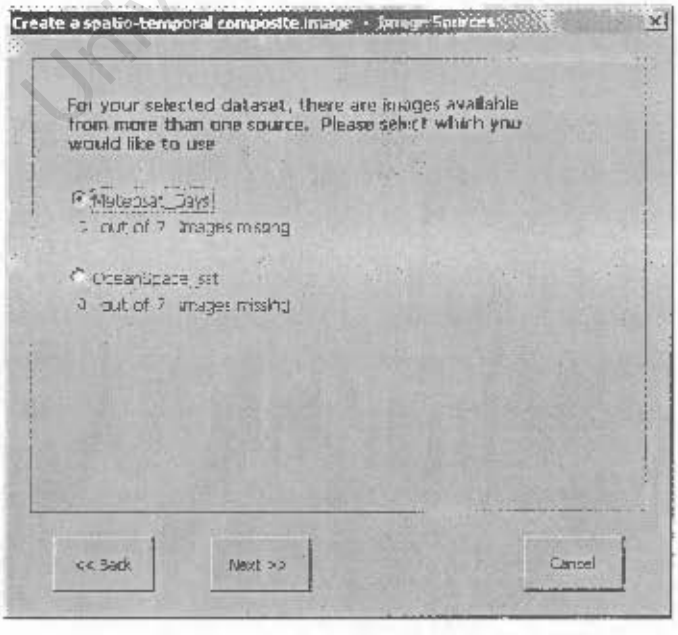


Figure 5.11 Screen for choosing an image source

Another consideration when retrieving available images occurs when the chosen temporal resolution is anything more than daily. If the resolution requested by the user is 2-day, it means that images of two consecutive dates are 'averaged' (described in section 5.5.1) to create a single image for both dates. If there are an odd number of dates in the dataset, the last image will be single and hence won't have another image to be averaged with. Data from the single image could simply be used but to keep it consistent with other images, it was suggested to and agreed by the client to get an image for the following date, and use this to be averaged with the last date. If resolution is 3-day, one or two extra images may be needed depending on the total number of dates in the dataset.

If no images are available for these extra dates, this is accounted for and just the single image is used, or if it is a 3-day resolution and only one extra date is missing, the other two are averaged together instead of all three etc.

5.4 Creating Polygons for Clipping Images

5.4.1 Determining the Buffer Area

The main input for the next few steps is the input point shapefile and what is ultimately needed is to create an area around the points which will ultimately determine the extent of the composite image.

One method to define this area is to create a convex hull around the survey points. The convex hull of a set of points is the smallest convex polygon that encloses all the points [ESR05], an example of which is shown Figure 5.12a. The outer lying points will lie on the boundary of the convex hull polygon.

Ideally however it would better if the area extended for some distance beyond the outer lying points. Creating a 'buffer' around these points would solve this problem. An ArcObject method from the *ITopologicalOperator Interface* is provided to add in a specified buffer distance around the points; an example of this buffering can be seen in Figure 5.12b with a buffer distance of 0.3 decimal degrees.

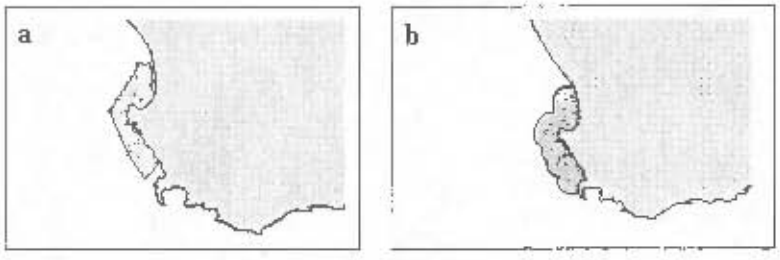


Figure 5.12 a) Convex Hull b) Buffering

Although this operation with this buffer distance works for the above dataset, if we look at the Figure 5.13 below of a different survey, it can be seen that here, this buffer distance is not suitable. When the data points are far apart, the buffer distance is not large enough to produce one continuous area.

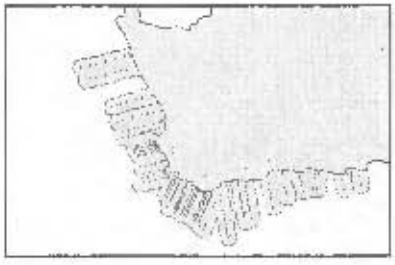


Figure 5.13 Buffer distance too small to create one continuous area

The problem encountered here is therefore how big to create the buffer distance to ensure that one continuous area is created for all possible datasets. The buffer distance can be increased to something much larger, for example in Figures 5.14a and Figure 5.14b below the distance has been increased to 0.7 for two sets of data points. In both of these, there is now one continuous buffer area as required, however while this increased distance may be suitable for the first set of points (Figure 5.14a), for the smaller set (Figure 5.14b) the area may be too large.

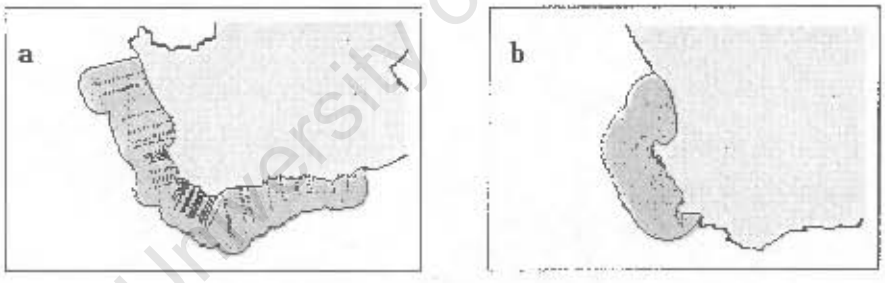


Figure 5.14 a) Buffer area the correct size b) Buffer area too big

A reasonable solution to this problem, and one that has been implemented, is to calculate the mean distance between all the points and use a function of this value as a buffer distance. This value would at least then be standardized but still change to fit the spatial extent of the individual dataset.

Clipping buffer with land polygon

Because the data points lie along the coast, close to land, the buffer area will extend over the land. It therefore needs to be clipped to the shape of the land and this is done using the land polygon shapefile provided by the user. The result of these two steps is a single buffer polygon extending over the ocean only.

5.4.2 Dirichlet Tessellation

In line with the discussion in section 3.3, it is necessary to associate every part of the buffer area with a specific data point nearest to it. This is done by creating Thiessen polygons around all the points in the shapefile.

5.4.2.1 *ConvertToVoronoiRegions* Method

There is an ArcObjects method called *ConvertToVoronoiRegions* which creates Thiessen polygons from a set of points and uses a 'clipper polygon' to define the extent of the Thiessen polygons. This 'clipper polygon' will be the buffer area that was created in the previous step. Although the results of this method appear to be correct, when the polygons are dissolved by date, there are odd lines which appear in places. This problem is illustrated in Figure 5.15. Figure 5.15a shows the Thiessen polygons for a set of points. Figure 5.15b shows the odd lines that appear when dissolving polygons of the same date after using the *ConvertToVoronoiRegions* method to create the Thiessen polygons. An alternative method was found to create the Thiessen polygons using a DLL. When using this method and the days are dissolved, the odd lines do not appear and the resulting polygons appear as they should, illustrated in Figure 5.15c. This indicated that it was the *ConvertToVoronoiRegions* method that was causing the problem and not the dissolve operation.

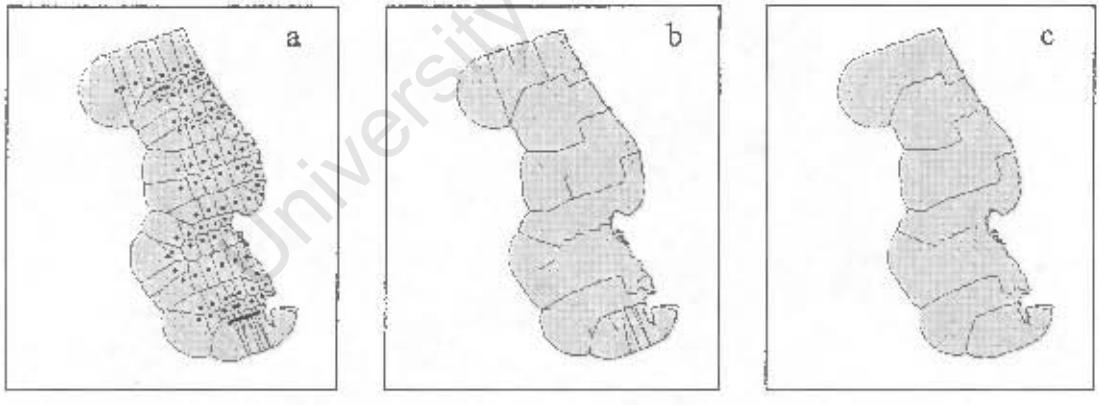


Figure 5.15 a) Thiessen polygons b) Dissolved by date when using the *ConvertToVoronoiRegions* method c) Dissolved by date using DLL

When the Thiessen polygons were examined more closely it was apparent that there were slight overlaps between some adjacent polygons and tiny gaps between others. These overlaps and gaps were not noticeable when viewing the polygons in a normal resolution in ArcMap and only become evident when it is zoomed in to the maximum. These gaps are not actual polygons, but rather small areas between other polygons.

For the above tessellation, in real terms a single Thiessen polygon may span about 50km, while the gaps are mere centimetres in size.

For other uses of these Thiessen polygons, such small anomalies may not matter, however for purposes of this project where it is necessary to dissolve the Thiessen polygons, the gaps and overlaps present a problem. The gaps are empty areas of space and do not have any 'date' attribute. When a dissolve by date is performed, they will not be dissolved into any of their adjacent Thiessen polygons, and are left as empty spaces. It is these empty spaces that appear as the odd lines as in Figure 5.15b.

A search of the ESRI forum revealed that I was not the first person to encounter this problem. There is a post where someone asks: 'I'm using VB 6 to build an ArcMap extension. One of the extension tools creates Thiessen polygons using the ConvertToVoronoiRegions function. The polygons in the output shapefile pOutFC have gaps and slivers. How can these gaps and slivers be avoided or eliminated?' [Nev05]. One reply indicated that the person ended up using the DLL described below and actually reported the problem to ESRI and got a bug number but at present did not know anything on the status of this bug.

A DLL called 'Create Thiessen Polygons 3.0' has been written by Tim Lomas and is available on the ESRI website [Lom03]. He notes that: 'This release improves the topological consistency of the output polygons. It is now possible to "dissolve" the output polygons without first "cleaning" them. The clean functionality is now included as part of the Create Thiessen application'. His words here confirm that Thiessen polygons do have to be 'cleaned' before they can be dissolved.

As this is a DLL, the code is not freely available for use and after extensive searches no other code or functions could be found to create Thiessen polygons without gaps. Two options were therefore remained:

1) To attempt to clean the shapefile of the tessellation created using the *ConvertToVoronoiRegions* method
2) To write my own code to perform the tessellation

5.4.2.2 Removing Gaps and Overlaps

Option one was considered initially. 'Cleaning' a shapefile would essentially involve removing any overlapping polygons, slivers and gaps. There is much discussion on the ESRI forums on different ways to avoid and eliminate these overlaps, gaps and slivers.

There is a sample in the ArcObjects Developers Help called 'Export and Clean Shapefile'. As described in the Developers Help, 'This utility creates a new clean Shapefile based on an input Shapefile. A clean Shapefile is one that has been simplified - its geometry is topologically consistent. For example, a clean polygon shapefile will have no overlapping rings, all exterior and interior rings have the proper orientation, and all non-closed polygons are [ADH04]. Use of this sample was suggested by a few people and seemed to solve their problems, however when run on my tessellated shapefile some slivers still remained.

ETGeoWizards (http://www.ian-ko.com/) is a set of powerful functions that provide ArcGIS users easy ways to manipulate data. The author of this site, Ianko Tchoukanski, notes in an ESRI forum 'In ArcGIS (excluding ArcInfo workstation) there is no standard function that will identify and clean the gaps'. He goes on to suggest using the 'CleanGaps' function of ETGeoWizards to solve this problem. For this function to work properly however, he notes that the gaps need to be entirely closed by real polygons. If the gaps are not entirely closed by real polygons, he says he doesn't think there is a solution other than manual editing [Tc05a]. Because the actual code is not available for this function and some of the gaps were not entirely enclosed by polygons, this method of removing the gaps was ruled out as a solution.

Another suggestion given on the forum was to 'union' the tessellated shapefile containing the gaps and overlaps with the buffer polygon shapefile that was used to determine the extent. The 'union' command 'combines features from different layers into one feature while maintaining the original features and attributes' [ESR05]. The results of this union would produce a shapefile where all the gaps and overlaps would be converted to sliver polygons. As these sliver polygons would be extremely small in relation to the Thiessen polygons they could be identified based on their size and then merged with the larger Thiessen polygons. Determining which Thiessen polygon to merge them with could be based on the Thiessen polygon with the largest adjacent boundary to the sliver polygon or the Thiessen polygon with the largest adjacent area.

This option was investigated, however it was decided that it would not be ideal. It is necessary to predetermine some cut off size for determining which polygons were slivers and which were the actual Thiessen polygons. In testing this option with a few datasets it was found that although it worked for most slivers, occasionally there would be a sliver that was much larger than the set cut-off size and so would not be recognised as a sliver. In addition, this option would require a fair bit of coding and process time. In the end it was decided that the best option would be to write the code to perform the tessellation and so avoid the gap problem completely.

5.4.2.3 Tessellation Algorithm

The Bowyer-Watson Algorithm discussed in section 2.5 forms the basis for the algorithm used to perform the tessellation for this project, but with slight adaptations to suit its implementation in a GIS.

While the Bowyer-Watson Algorithm keeps a list of triangles, my algorithm actually creates circumcircles from these triangles and keeps a list of these circles together with the three points that the circles were made from (called *circleList*). When a new point is added, as with the Bowyer-Watson Algorithm, a check is done to see which circles the point lies within. These circles are removed from the *circleList* and their three points are added to another list (which we will call *buildList*). Once all the circles have been checked, the *buildList* will contain all the points from all the circles that have been removed. Any duplicate points are removed. The points in this *buildList* will be the points together with the newly added point which will be used to create a new set of circles. This step is the equivalent of forming new triangles between the new point and each outside edge of the enclosing polygon in the Bowyer-Watson Algorithm.

Pairs of points from the *buildList* plus the newly added point are then used to create new circles. The pairing of these points is however, important. If we think of the Bowyer-Watson Algorithm, in order to create new triangles (that do not overlap any other triangles), the pair of points used are those two points that lie on either end of one of the polygon boundaries. For example, in Figure 2.14c, P1 and P2 are used together with the newly added point to form a triangle as these two points lie on each end of a boundary of the polygon. In order to replicate this obvious pairing in my algorithm, all the points in the *buildList* need to be sorted according to their spatial location. They are sorted radially so that the first two will become a pair, the 2nd and 3rd, 3rd and 4th etc. A circle is then created with the newly added point and each of these pairs; the new circles then being added to the *circleList*.

Once all the points have been added, the *circleList* will contain a list of circles that do not contain any other points. In effect, the list of three points stored for each circle will be the three points of the triangles that would make the Delaunay triangulation. At this point, it is simply a case of identifying all the circles that pass through each point, getting their centre points, and then joining these centre points to create the Thiessen polygon for that point.

The complete algorithm is described in pseudo-code as follows:

```
Function: MakeThiessenPolygon
Input: point shapefile
Output: Thiessen polygon shapefile

Initialize the circleList (will contain the actual circle and the three points)
Generate three outlying points
Create a circle from these points
Add the circle (& it points) to the circleList

For each point (call NewPoint) in the input shapefile
        Initialize the buildList
        For each circle currently in the circleList
                Calculate the distance between the NewPoint and the centre of the circle
                Calculate the radius of the circle
                If the distance is less than the radius (i.e. the newPoint lies in the circle)
                        Add the three points of the circle to the buildList
                        Remove the circle and its points from the circleList
                End if
                Remove duplicates from the buildList
                Call sortFunction to sort the points in the buildList radially
                Get the sequential pairs of points from this sorted buildList and use
                  together with the NewPoint to create a new circle
                Add this new circle and its three points to circleList
        End for each circle
End for each point

Initialize polygonList (list of Thiessen polygons)

For each point (NewPoint) in the input Shapefile
Initialize a thiessenList (list of circle centre/Thiessen vertices)
        For each circle in circleList
                If the NewPoint is one of the three circleList points
                        Get the centre point of the circle
                        Add this centre point to the thiessenList
                End if
        End for
        Call sortFunction to sort the points in the thiessenList radially
        Make a polygon from the points in the thiessenList
        Add this polygon to the polygonList
End for each point

Make one shapefile with all the polygons in the polygonList.

End Function
```

Figure 5.20 below shows the final tessellation for a set of points using the above algorithm. The odd looking shape of the tessellated area in Figure 5.20a is due to the three outlying points that were added in the first step. Figure 5.20b shows the close-up of the tessellated area around the datapoints.

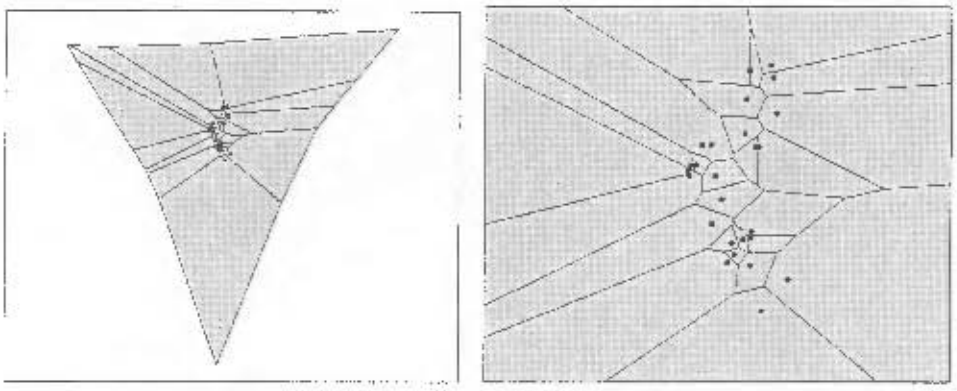


Figure 5.20 a) Tessellation for a set of points b) Close-up view

5.4.2.4 Clipping Problems

As this tessellation extends well beyond the data points it is then necessary to clip it with some user defined area. This will be the 'buffer area' that was created previously. Initially this clip operation was performed using a clip method belonging to the *ITopologicalOperator Interface*. When the Thiessen polygons were dissolved by dates in the next step, the results showed the same problem that was encountered when using the *ConvertToVoronoiRegions* method to perform the tessellation: there were slivers present between some of the adjacent polygons and odd lines appeared when polygons were dissolved (as shown in Figure 5.15b).

It was determined that these slivers were not present before the clip operation and it was therefore this clipping that actually caused the problems. Another clip method was found which belonged to the *IBasicGeoprocessor Interface*, and when this was used no slivers were created.

5.4.2.5 Removing Duplicate Points

After testing of the tessellation code with different datasets it appeared that it worked perfectly for all but a few cases. In these few, no Thiessen polygon was created for the odd point (Figure 5.21a) or odd lines appeared above a correct or almost correct tessellation (Figure 5.21b).

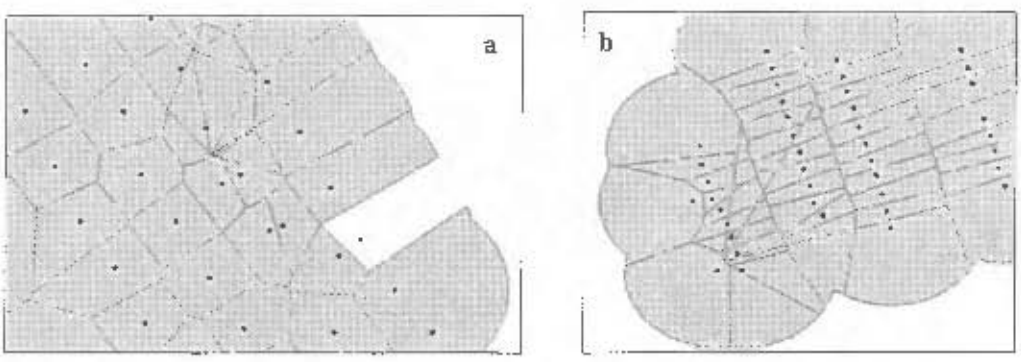


Figure 5.21 a) Point for which no polygon was created b) Odd lines in tessellation

It was determined that these irregularities occurred when there were duplicate points in the input datasets i.e. points with exactly the same spatial locations. This problem was solved by removing the duplicate points before the tessellation algorithm was performed. No ArcObjects method could be found that easily identified and removed duplicate geometric structures. The *ITableSort Interface* however, allows the rows in a shapefile's attribute table to be sorted by a given field. Some input shapefiles have latitude and longitude fields already defined and so these could be used to sort the data. However, as other datasets may not have such fields, it was necessary to create two new fields, one for the latitude and one for longitude and populate them with each point's x and y co-ordinates. These two new fields were then used to perform the sort.

ET GeoWizards has a function to 'Build Thiessen Polygons'. The actual code is not available, however, it is interesting to note that in the description of this function, it says that one of the first steps performed is to 'clean duplicate points' [Tc05b].

5.4.2.6 Spatial Join

Once these Thiessen polygons have been created, they are merely polygons with no other attributes. What is then needed is to associate each of these polygons with the datapoints around which they were created. This will in effect give each Thiessen polygon a 'date' attribute. This operation is performed using a spatial join of the input point shapefile and the newly created Thiessen polygon shapefile. A spatial join is a type of table join operation in which 'fields from one layer's attribute table are appended to another layer's attribute table based on the relative locations of the features in the two layers' [ESR05].

5.4.3 Dissolving Polygons by Date

The final step in this section is to dissolve the Thiessen polygons by their dates. For the Thiessen polygons this involves getting all the polygons of the same date and dissolving their boundaries to create one large polygon. Each of these larger polygons then represents the total area covered on each date and will be used to clip the satellite image of the same date.

If the user chooses a daily resolution, the dissolve is based on each date. For any resolution greater than daily, it is necessary to merge the dates in the shapefile based on the resolution. If the resolution was 2-days, this would require all polygons for every two consecutive dates to be merged into one date. And likewise, for a resolution of 3-days, every three consecutive dates would be merged. An example of this is shown in Table 5.1 below. The new date value is given as the first date in the group.

Resolution: daily	Resolution: 2 days	Resolution: 3 days
01/05/99	01/05/99	01/05/99
01/05/99	01/05/99	01/05/99
01/05/99	01/05/99	01/05/99
02/05/99	01/05/99	01/05/99
02/05/99	01/05/99	01/05/99
03/05/99	03/05/99	01/05/99
03/05/99	03/05/99	01/05/99
03/05/99	03/05/99	01/05/99
04/05/99	03/05/99	04/05/99
04/05/99	03/05/99	04/05/99
05/05/99	05/05/99	04/05/99

Table 5.1 Example date values for daily, 2-day and 3-day resolution

Instead of replacing the values in the existing date field with the new values, a new field was created in the shapefile which was populated with the new values. At the same time a new array is created recording these new dates. For resolutions greater than daily, this array would be used in subsequent steps instead of the original *satDatesArray* created earlier in the program. The dissolve is then performed on these new values. The polygons created from this dissolve operation are then used to clip the satellite images in the next section.

5.5 Raster Processing

5.5.1 Averaging Images if Resolution is Greater than Daily

If the resolution is daily, the original images (which are daily images) will be clipped using the daily dissolved polygons. For any resolutions greater than daily, the raster images need to be averaged. More specifically, this involves getting an average value for two or more images for each pixel and creating a new image based on these averaged values.

The number of images averaged together will be the same as the resolution. So for a 2-day resolution, just as the polygons were dissolved for every two dates, images for every two dates must be averaged together. These images will then be clipped using the dissolved polygon of the same dates.

Through the *IRasterGeometeryProc Interface* there is a Mosaic method which as described in the ArcObjects Developers Help 'combines multiple adjacent datasets into a single output raster dataset. It determines the value of an output cell by averaging the values of the overlapping input cells' [ADH04]. This appeared as an ideal method to use, however two problems existed:

1) This method can only be performed on two images, and so would be useless when needing to average three or more images.

2) It was found that the pixel values of the 'mosaiced' image were incorrect. This problem is described below.

During the mosaic method, each raster is put into a Raster Band Collection. The order in which the rasters were added to the collection appeared to determine the value of the mosaiced pixels. To test this, the order in which the two images were added was changed and random pixels values of the mosaiced raster were checked. These are shown in the Table 5.2 below.

Raster Image	Pixel 1	Pixel 2	Pixel 3	Pixel 4	Pixel 5
Raster1 (original)	1.25	3.42	2.78	No data	1.65
Raster2 (original)	1.90	No data	3.80	0.77	3.67
Merged (R1 added 1st)	**1.25**	**3.42**	**2.78**	**0.77**	**1.65**
Merged (R2 added 1st)	**1.90**	**3.42**	**3.80**	**0.77**	**3.67**

Table 5.2 Pixels values when images are mosaiced using *IRasterGeometeryProc*

It can be see from here that the value of the mosaiced pixel was the value of the pixel from the raster that was added first and not an average of the two pixels. When either of the two was a 'no data' pixel, the other value was used, which is what should happen. The fact that this does occur shows that both rasters are in fact being 'read', but that when it comes to actually averaging two pixels which both have values it is not doing so and takes the first value.

What added to the peculiarity of these results was that there is a sample of code in the Developers Help to perform this operation which should be able to be copied and used as it is given. Even though this was done, the results were still incorrect. The same operation was then attempted but using the raster calculator that is available as part of the ArcMap interface. Here it was found that the 'averaged' pixel values were different to those obtained using the mosaic method, but were still incorrect.

These two factors led me to believe that perhaps in order to perform these operations the raster must be of a particular format. After some investigation, I could not find any other references or solution to this problem. For this reason, as well as the fact that the mosaic method could only be performed on two raster images anyway, it was decided to write functions to average the images from scratch.

The method of averaging raster images involves going through each pixel, averaging the values of the pixels, and writing this value to a new raster. When doing this it had to be kept in mind that there may be images missing. If this averaging operation was attempted on a 'no image' it would cause an error. The following scenarios therefore exist:

1) If no images are present for the averaging operation, (i.e. two consecutive images missing if the resolution was two, or three images missing if the resolution was three), then it is skipped.
2) If one image is missing, then the remaining image/s are used.
3) If all images are available, then averaging occurs as normal.
4) If it was determined earlier in the program that extra days were needed, then these are used here.

In a similar manner as the images are checked for absentees, the same must be done with each pixel when averaging to check for 'no data' pixels. For example, for a resolution of 2-days, if both pixels are 'data' pixels the two are averaged, if one is a 'no data' (cloud) pixel then the other value is used, and if both are 'no data' pixels the new value would remain 'no data'.

Since the code to perform the averaging of these images was fairly lengthy, initially functions were only written to average 2-days and 3-days. These two functions were then used to implement the 4, 6 and 8 day resolutions. So for example, a 4-day resolution involved using the 2-day function to average every two dates, and then using these 2-day composite images and again using the 2-day function to average every two images. It was subsequently realized however that these calculations would only work if there were no 'no data' pixels or if there were the same number of 'no data' pixels in each function.

For example, the six pixel values of **15, 7, 18, 19, 21 and 14** are averaged to **15.66.** If we now divide these into two groups of **15, 7, 18** and **19, 21, 14**, the average of these two groups is **13.33** and **18** respectively. These two values averaged gives **15.66**, the same value as if we'd calculated it in one step.

Now if one of these pixels has a 'no data' value this will not work which the following example illustrates. The pixel values **15, 7, no data, 19, 21 and 14** are averaged to **15.2**. This is because we treat the 'no data' pixel not as a zero but absent and so we disregard it and use the other five numbers to get an average. If we now divide these into two groups of **15, 7, no data** and **19, 21 and 14** and average them we get **11** and **18**. These two values averaged give **14.5**, which is incorrect.

For this reason, the 2-day and 3-day functions could not be used to implement the higher resolutions which needed to be individually coded. Since the code in these averaging functions must check for missing images and 'no data' pixels and account for these by using the reminder of the images or 'data' pixels in the averaging operation, the higher resolution functions must include large amounts of code to

account for all eventualities. Functions were therefore written for a 4-day and 5-day resolution however it was decided that because of the large amount of code needed for the 6-day and 7-day and because these most probably would very seldom be used anyway, they would not be written.

5.5.2 Clipping Images with the Dissolved Polygons

The satellite images are then clipped with the relevant polygons. The polygons which are used are those that were created by dissolving the dates (in the final step of section 5.4.3). If the resolution is daily, then the original satellite images are used, their names being stored in the *satNamesArray*. If resolution is greater than daily then the averaged images are used, their names being stored in the *newNamesArray*. The clipped images are saved and used in the next step.

5.5.3 Merging the Image Clips into One Image

The image clips are then merged together in one image. Because the polygons that are used to define the extent of each clip will not overlap, the image clips will also not overlap and will fit together to form a seemingly single image. This composite image is the final output from the system and is saved in the output folder the user selected.

The name given to the image will reflect the data used to create it. For example the name '10015_ch_ci2' indicates the composite image was created for a dataset named 10015, using chlorophyll images (ch) and at a 2-day resolution (2). The 'ci' indicates a composite image.

If a user chooses to create a second composite image but at a different resolution, the same working folder can be used to save this second image. This is important because all the intermediate shapefiles that were created (discussed in section 4.3.3) during the first run can be used again.

5.5.4 ColourMap

A requirement of the tool is to produce images that are displayed with standard sets of colours for the different parameters (SST/chlorophyll). This would be done by applying a colourmap with prewritten RGB values to the image. The problem however, is that the original satellite images that are used, do not actually have a colourmap present, and even if they did, colourmaps are lost when spatial operations are performed on raster images. Although the raster object does have a colourmap property, it is read-only (ADH05).

Below is an extract from an ESRI forum that was posted by a member of their raster development team confirming this problem:
'The RasterColormap is a coclass, you can create a colormap and populate (it) with the existing colormap values (red, blue, green), but the IRasterDataset:Colormap is read only, you can't apply a colormap to a raster dataset. If you only want to display the extracted image with (an) old colormap, you can read the color for each value from old colormap table and use customized raster unique value render to display, but the defined color is not saved with the raster dataset' [Hon01].

A workaround to this problem as suggested by the comment above is to create a customized renderer to display the raster image. This is achieved using the *RasterClassifyColorRampRenderer Interface* which visualizes a raster as a set of classes. Each class is displayed using a unique colour and each class contains one or more values from within the raster. Using this, the exact colours to be used for the different classes and the range of values for each class can be set. Although this method displays the raster image in ArcMap as required, these changes are still not saved as part of the image. A solution to this problem is to save the layer (containing the composite image) as a .lyr file. Doing this preserves the renderer and so when the image is displayed again it will have retained its required colours.

5.5.5 Calculating the Percentage Cloud Cover

The last operation that needs to be performed is to count the number of 'no data' pixels in the composite image (to determine percentage cloud cover). This value will be useful for the user to decide if he/she is satisfied with the image or would like to increase the resolution in an attempt to decrease the cloud cover.

It is simple enough to count the number of 'no data' pixels in a raster, however, even if there is only small clip of a raster, the whole area outside the 'clip' is still considered part of the image (size is set from original raster images). So for example, Figure 5.22a shows the extent of one of the original satellite images used to create the composite image in Figure 5.22b. A count of the 'no data' pixels in Figure 5.22b would therefore include all the outlying pixels that lie within the extent of the original image as well as any 'no data' pixels in the composite image itself. There is no way to distinguish the two and therefore get a count of the 'no data' pixels in the composite image alone.

A solution to this problem is to get the buffer polygon that was used to define the extent of the composite image (Figure 5.22c) and convert this to raster (Figure 5.22d). This new raster polygon will in effect cover exactly the same area as the composite

image, the only difference being that all the cells are given a value of 0. This allows us to determine the total number of pixels in the composite image by counting the number of pixels with a value of 0. From this we can calculate the number of 'no data' pixels in the remaining area around the composite image and finally the actual number of 'no data' pixels in the composite image.

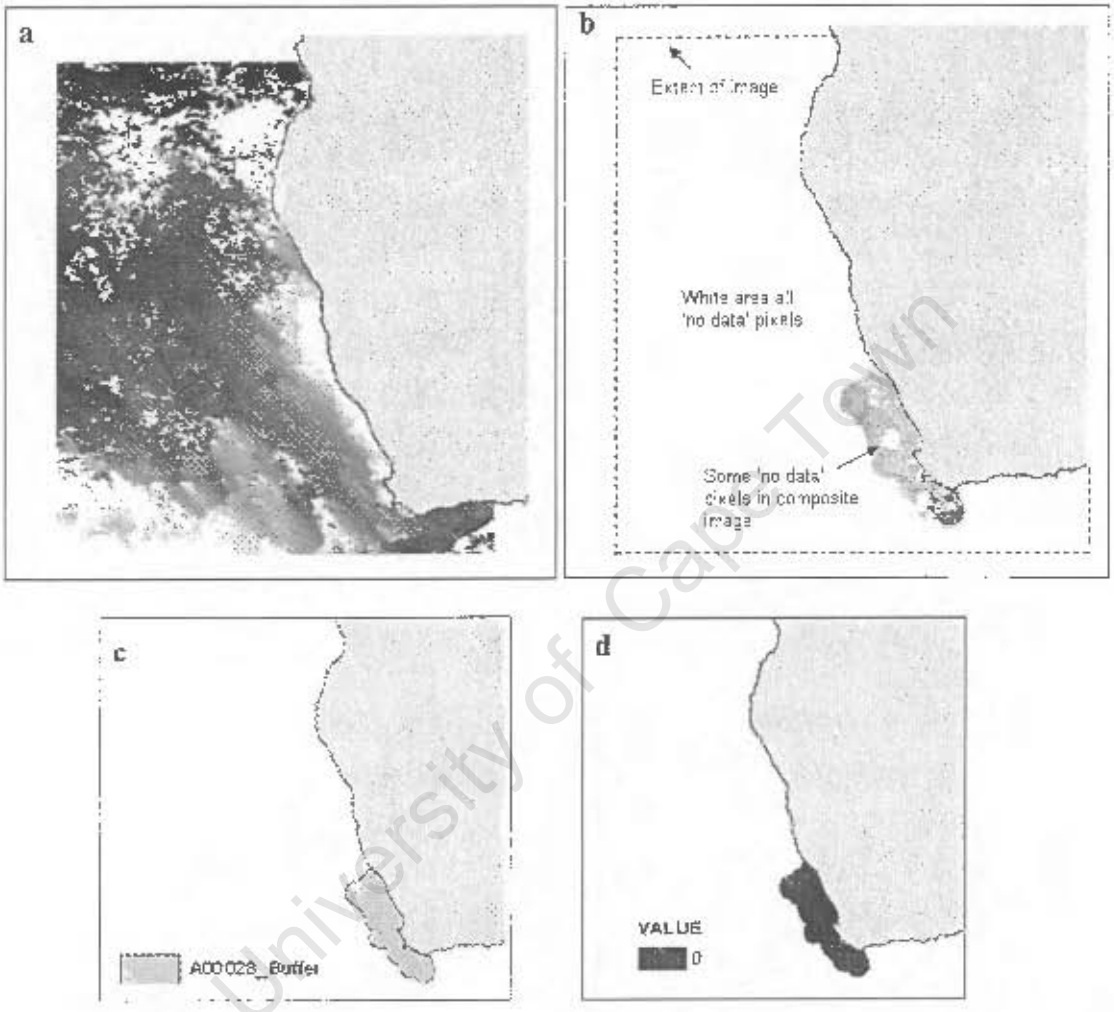


Figure 5.22 a) Original image b) Composite image showing extent of actual image c) Buffer polygon used to create composite image d) Polygon converted to a raster

5.5.6 Displaying a Results Screen

Once the final composite image has been displayed in ArcMap and the percentage cloud cover calculated, a final screen is displayed to the user (Figure 5.23). This contains the following information:

- The file name of the composite image.
- The folder it was saved in.
- The percentage cloud cover for the image.

This new composite image is now available for adding to any map in the same way as any normal raster file would be used in GIS.

The user is also given the option of creating another composite image using the same parameters (i.e. same input shapefile, type of image, output folder) but at a different resolution. The process will then be repeated but without needing to obtain information from the user.

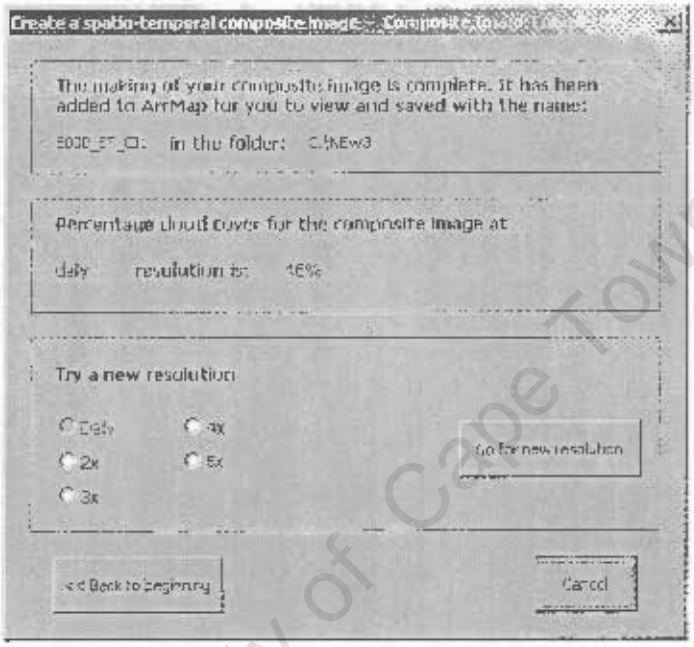


Figure 5.23 Final results screen

5.5 Summary

5.5.1 Interface

The GUI was implemented through a series of small, simple screens. The screens were kept simple, consistent and easy to use. Although advanced users may find having to repeatedly go through all the screens frustrating if creating multiple composite images, they are necessary to obtain all the information required to create the desired composite image. The implementation of a single, larger, more complex screen to obtain the required information was considered, however it is not possible to obtain all the information in one step. Thus, if the GUI consisted of a single screen, its contents would have to be changed three times to get all the required information. The nature of the information required lent itself to having a series of screens and this is what was implemented in the final product.

5.5.2 Making the Tool Generic

A metadatabase for checking image availability goes a long way to automating a process that would otherwise be time consuming for the user. Since a metadatabase had already been set up for all the satellite images available at MCM, the tool was implemented according to the format and structure of this database. As only a few of the tables and fields from the MCM metadatabase are used, only these would be required in any other metadatabase that was setup for use with the tool.

There is a fair amount of functionality in the tool dedicated to the date field. This may at first appear unnecessary, however the date field (date of each data point) is fundamental to creating the composite image. It is therefore vital to ensure that the dates are recognized by the tool. Unfortunately the manner in which the dates are recorded in the shapefile is beyond the control of the application. Although attempts can be made to automatically determine the correct format of the dates which are typed as Strings, ultimately confirmation of this is required from the user. For this reason, this feature was removed and the user asked to supply this information. Although it may be a bit tedious, it is necessary to ensure that the tool is generic and will run successfully using any input data.

The clipping operation (of the buffer area with the land polygon) is done to limit the extent of the composite image to the sea. The clipping function could potentially be used to delimit very specific areas where a composite image is required, for example, for a small section of the coast.

5.5.3 Problems with ArcObjects

Tessellation is widely used in geographical operations and it was therefore surprising to discover that this ArcObjects method does not work correctly in that there are slight gaps and overlaps between adjacent tessellated polygons. Most people using a tessellation created using the ArcObject method will probably not dissolve the polygons and therefore not have any problems. Having written my own tessellation function, not only did my code create accurate tessellations, being identical to those produced from a DLL provided on the ESRI website, but my code also created the tessellations far quicker than the DLL.

A problem was also found with the *IRasterGeometryProc Interface*, Mosaic method in that the value of the pixels in raster images were not averaged correctly. For this reason, as well as the fact that there are no methods available to average more than two images, functions were written to perform all the averaging operations.

6 System Testing

An important part of developing any system is testing and evaluating the final product. Testing the functionality of the system was done by using the tool with a variety of input data in an attempt to identify and fix any bugs that may have been present and checking the accuracy of the output of the product. Finally, a test was done to evaluate the success of the product in terms of time i.e. how much time is saved by using the tool rather than creating a composite image manually. Each of these is discussed in this chapter.

6.1 Robustness Testing

During the development of the tool, each function that was written obviously needed to be tested on some data to ensure that it was working correctly. This meant that the tool was continuously being tested as it was implemented. For the most part, the data used for testing during the development was a shapefile called 'Acoustics' provided by MCM. The near-final product was therefore error free when using this shapefile as input. As already noted, it was important to ensure that the tool could be used for a variety of data or shapefiles. During the implementation a few other shapefiles were therefore also used to run the tool. A number of problems were encountered when these shapefiles were used and alterations were therefore made to the design and implementation of the tool as and when these problems were encountered. Once the tool was deemed complete, a number of new shapefiles were used as input in a final testing stage. The main problems discovered during the implementation and final testing stages are discussed below.

6.1.1 Missing Dates in Input Shapefile

Some input shapefiles had missing dates. For example data points may have existed for 8th, 9th, 12th, 14th August etc., with no points for the 10th, 11th and 13th. If a daily temporal resolution was used to create the composite image, these missing days would not make any difference. If however, the temporal resolution was anything more than daily, satellite images of consecutive images would have to be averaged. For a 2-day resolution, this would mean that images for the 12th and 14th would be averaged when in fact they are not actually consecutive dates. This problem was discussed with the client, who did not think there was any need to for me to check and account for such an eventuality. There were two reasons for this. First, most composite images would be created using a daily resolution anyway where this would not be a problem. Second, the majority of their data would not have any dates missing, it being a rare occurrence rather than the norm. The application was therefore

not amended in any way to check for missing dates, although a future improvement to the tool could do so.

6.1.2 File Names for Raster Images

It was discovered that there was a limit to the number of characters that could be used in the file name when creating and saving raster images. Spaces in the file name also caused errors. Code was therefore included to check the length of the file name and if too long the user is required to choose a shorter name. If any spaces are found in the file name, these are removed.

6.1.3 Datapoints Incorrectly Georeferenced

In a few of the shapefiles tested, there were the odd datapoints that had been incorrectly georeferenced so that they were located over the land and not the sea. For example, see Figure 6.1 which shows a single point on the land. If these incorrect datapoints were sampled on a day when no other datapoints were sampled this would cause an error in the program. This was because once the Thiessen polygons have been created for all the points, they are clipped using the land polygon. The result would be that the Thiessen polygon for this misplaced point would be clipped out. Once the Thiessen polygons are dissolved by date, the number of dissolved polygons would be one short of the number of dates originally obtained from the input shapefile. This problem was discussed with the client who said that I should not worry about it because the input shapefiles provided should be checked for such errors before they are used. Again, a future improvement to the tool could incorporate a system to check for such misplaced points before any processing begins. This could be done by using the land polygon provided by the user to indicate areas where no points should lie and remove these from the original set of points before processing begins.

Figure 6.1 Dataset with an incorrectly georeferenced point located over the land

6.1.4 Buffer Operation Not Working

In a single shapefile there was a problem performing the buffer operation. Although it created a buffer shapefile, and in the attribute table of this file it appeared as if there was a polygon (see Figure 6.2 below), the shapefile was actually empty. An error therefore occurred during the tessellation when the buffer area was used to clip the Thiessen polygon. The error received is shown in Figure 6.3.

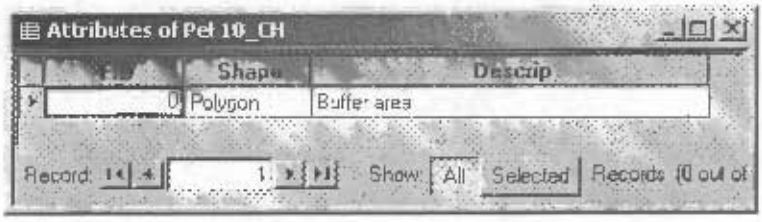


Figure 6.2 Attribute table for the buffer polygon

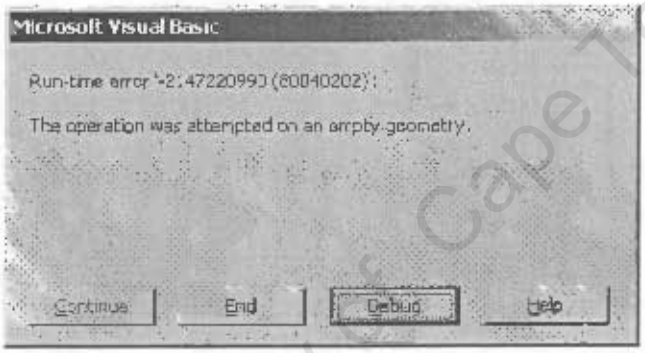


Figure 6.3 Error message received when trying to clip Thiessen polygon with the buffer polygon

A number of things were done in an attempt to determine the source of and solve this problem. When creating the buffer area, all the points are added to a point collection. A check revealed the points had in fact been added to the collection, indicating that this was not the problem. Using the point collection, one can also create a convex hull around the data points. The convex hull and buffer methods are from the same interface (*ITopologicalOperator*) and use the same set of points. The convex hull was created without any problems which suggested that there was some problem with the buffer method itself and not with any operations leading up to it.

The buffer distance was then changed to see if this would make a difference, which it did not. An attempt to create a buffer area around the data points manually in the ArcMap interface was successful. In the end, the source of the problem could not be identified. As a fix, a line of code was added to the function which checked to see if the 'geometry' was empty. If this was the case, then a convex hull was created around

the points and this used instead of a buffer. A message was also displayed informing the user that this had occurred.

6.1.5 Conclusions

After extensive testing of the tool using a number of different shapefiles as input, for these files at least, the system appeared to be bug-free. Almost all of the problems or potential problems that were discovered were either fixed or as was the case with the incorrectly georeferenced points, considered beyond the scope of the system to account for. The only major issue that remained unresolved was the buffer operation which did not function correctly for a single shapefile but for which a remedy included should this occur.

The tool can therefore be considered a success in terms both of the user being able to use a tool that is free of errors and making it generic so that it can be used with any input data.

6.2 Accuracy Testing

Accuracy testing was done to check that the extent and data in the output of the tool i.e. the composite image, was correct.

6.2.1 Testing the 'Averaging' Functions

Testing was done to check that all the functions that averaged the satellite images produced accurate results. In order to do this, the pixel values in the original satellite images were compared with the pixel values in the composite image. This can easily be done by viewing all the images in ArcMap. There is an 'Identify' command in the tool's tool bar as shown in Figure 6.4 below.

Figure 6.4 Tools toolbar showing the 'Identify' command

This pointer can be placed at any location on a map displayed in ArcMap and will show the values of each of the layers that are displayed for that specific location. This means that the composite image and the original images can be displayed together and the values for individual pixels compared.

Figure 6.5 below shows five layers created for a dataset called 'B030' and displayed in ArcMap. The first two dates of the 'B030' dataset were 22/10/1995 and 23/10/1995. The layers are:

1) The dissolved polygons for a 2-day resolution.
2) A composite image.
3) An original satellite image called 'b19951022' (taken on 22/10/1995).
4) An original satellite image called 'b19951023' (taken on 23/10/1995).
5) A polygon showing South Africa..

The top-most polygon in the composite image represents the area covered over the two dates. Placing the 'Identify' pointer over any spot in this top polygon will give us the pixel values for the two original images and the composite images as shown in Figure 6.6 below. These values can be checked to see if the value of the pixel in the composite image (18.9) is indeed an average of the pixel values from the two original images (19.27 and 18.71), which it is.

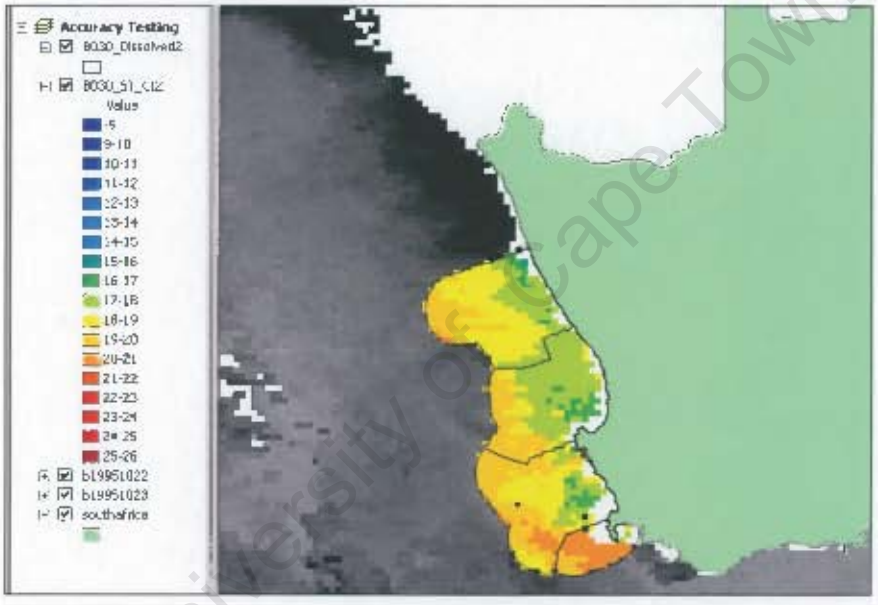


Figure 6.5 ArcMap interface showing maps with five layers

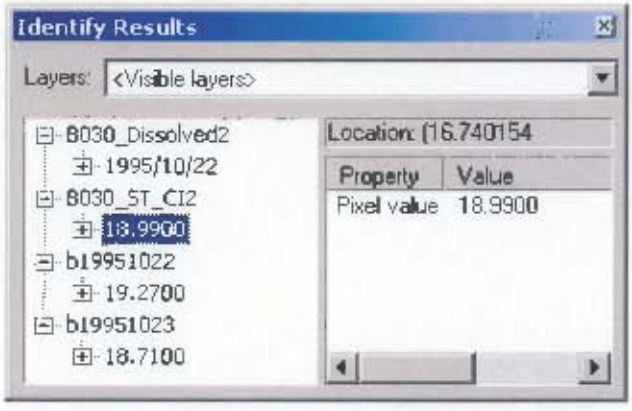


Figure 6.6 Results from the identify command showing pixels values for the different layers

Using this method, the pixel values were checked for a number of different composite images at different resolutions and all values were found to be correct.

6.2.2 Clipping Images with Polygons Correctly

A check was done to make sure that the image clips were exactly the same size as the polygons which were used to clip them. This was done by displaying the polygons in one layer in ArcMap, overlaying them with the individual image clips in another layer, and then visually checking that the image clips had the same extent as the polygons. In some of the datasets tested there were no problems, however, in others it was found that the shape of the image clips were not identical to the polygons used to clip them. On closer inspection it was found that these problems only occurred when the polygons did not consist of one continuous area.

This problem is illustrated in the diagrams below which show the dissolved polygons for a dataset called A00028 in one layer and individual image clips in another layer. The blue outlines a polygon for a single date. Figure 6.7a shows that when the polygon consists of a single, continuous area the image is clipped perfectly. In Figure 6.7b, the highlighted polygon is not continuous and here there are odd pixels which are present in the image clip which lie outside the polygon.

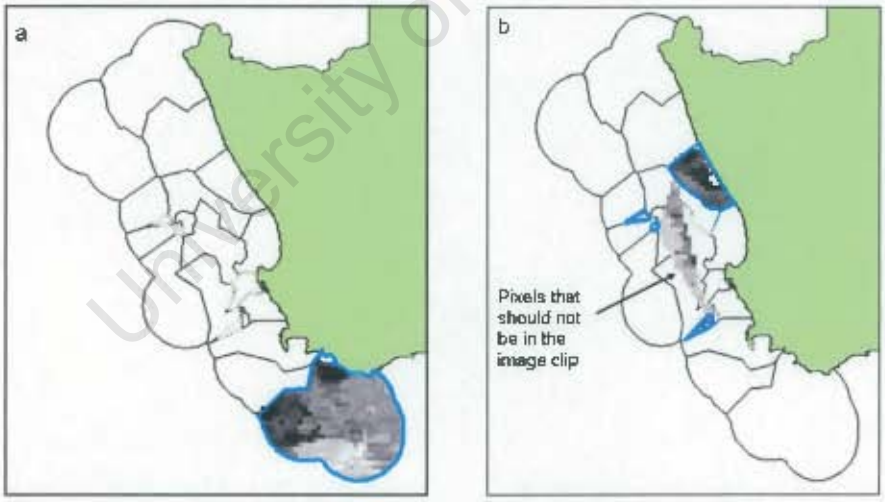


Figure 6.7 a) Correctly clipped image b) Incorrectly clipped image when polygon is not continuous

Examining every polygon and image clip for this particular dataset, it was found that all the polygons that were not continuous clipped incorrectly while all the single area polygons were fine. This clipping operation is done through the *IExtractionOp Interface*, using the Polygon method which 'Extracts the cells of a raster based on a polygon' [ADH04]. A search on the ESRI forums for similar problems encountered

with this interface did not reveal much although in one post one person wrote 'I'm trying to extract the pixels contained in a polygon using the polygon method of IExtractionOp.... The extraction works but the result seems quite wrong. I guess the out raster should have the shape of the polygon,no? In any case it doesn't, the result is very strange.' [Gre01]. No answer was however provided to this post. I then posted a message myself explaining my specific problem and again did not receive any feedback. No solution or explanation for this problem was therefore found suggesting a possible bug in either the *IExtractionOp Interface* or more specifically with the Polygon method of this interface.

6.2.3 Comparing Composite Images Created by the Tool to Those Created Manually in ArcView 3

As a final test to check that the output of the tool was accurate, a composite image that was created by the tool was compared to one created by the client. The one created by the client was done manually using ArcView 3. As well as providing me with the actual composite image, the client was able to give me the 'buffer' shapefile used to determine the extent of the image as well as the shapefile containing the tessellation. These were also compared with the same shapefiles created by the tool. The same point shapefile and satellite images were used as input. The results of this comparison are discussed below.

Buffer area

The diagrams below show layers containing the 'buffer areas' used to determine the extent of the composite image. Figure 6.8a shows the area created by the client and Figure 6.8b that created by the tool. The dark points indicate the input point shapefile. As can be seen these two areas look very similar.

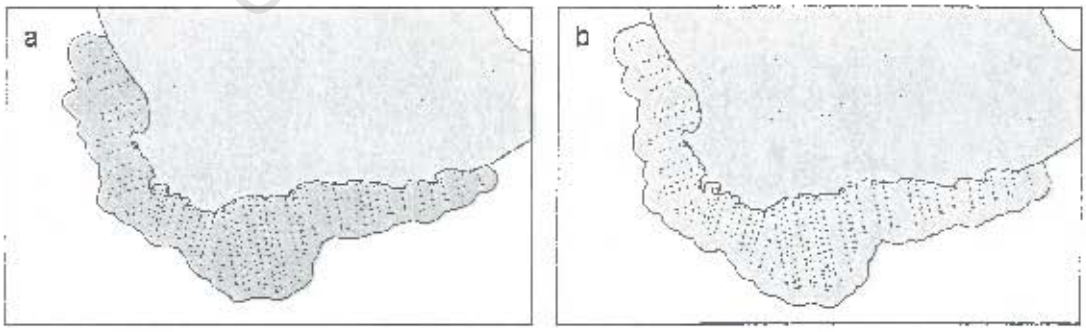


Figure 6.8 a) Extent from client b) Extent, created by buffering from tool

If we overlay these two layers, the difference between the two becomes more apparent. This is shown in Figure 6.9 where that created by the client in blue overlays

the beige extent created by the tool. It can be seen here that the tool's extent is slightly larger than the clients, but this difference is minor. The general shape of the extent is very similar. Differences are due to the fact that in the tool this extent is obtained by creating a buffer around the points where the outskirts of the buffer area is a set distance from the outlying points. The client on the other hand created this extent in an all together different way: by density kernel calculations. Such slight differences are not however important and would only become problematic if there were significant differences between the two extents.

Figure 6.9 Tool and client 'extent' layers overlaid.

Tessellation

The tessellations for these points are shown in the diagrams below. Again, the tessellation created by the client is shown in Figure 6.10a, while that created by the tool is shown in Figure 6.10b. As can be seen the two layers look very similar.

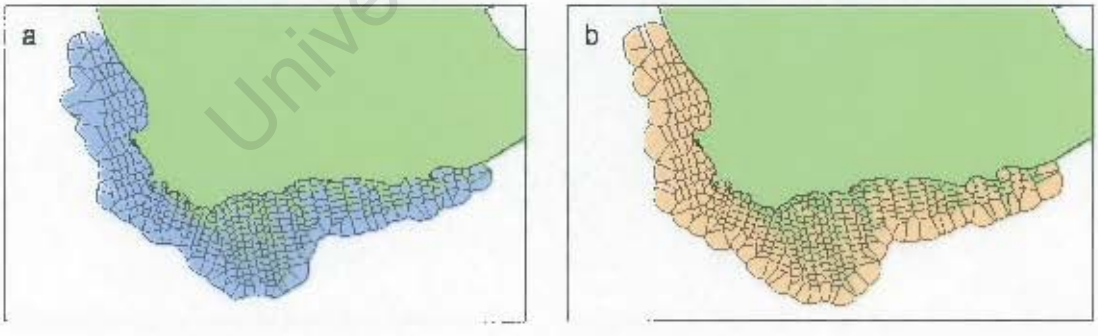


Figure 6.10a) Tessellation from client b) Tessellation from tool

To make sure that the two tessellations were in fact the same, the two layers were overlaid, the results of which are shown in Figure 6.11. The background colours of the layers have been removed and only the outlines of the tessellation polygons are visible. The client tessellation is shown in red, which overlays the tool's tessellation in black. It can be seen that there is very little black visible because it lies directly

underneath the red lines. This shows that the tessellation created by the tool was exactly the same as that from the client, the only difference being that the tool's tessellation extended a bit beyond that of the clients. This was because the 'extent' shown in Figure 6.9 used to determine the limits of the tessellation were slightly different, as explained in the previous section.

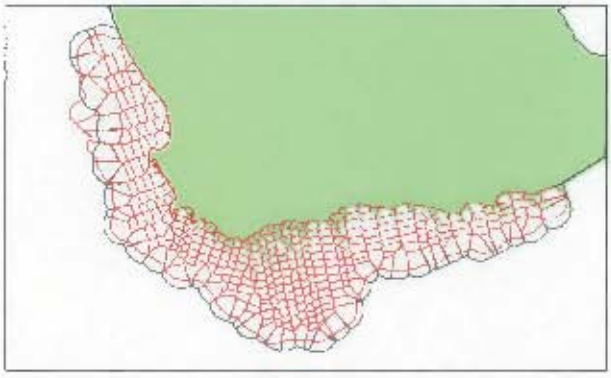


Figure 6.11 Tool and client tessellation layers overlaid

Composite Image

The final daily composite image created by the client is shown in Figure 6.12a below, and by the tool in Figure 6.12b. Visually the two images look identical apart from the tool's image having a slightly larger extent. This was confirmed by overlaying the two images and using the 'Identify Results' command (described in section 6.2.1) to check that the pixel values were the same in each image, which they were.

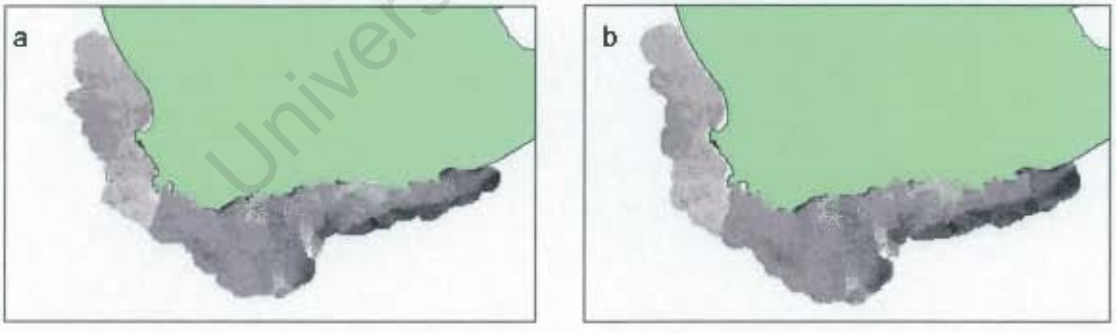


Figure 6.12 a) Daily composite created by client b) daily composite created by tool

In conclusion, this comparison has shown that the intermediate shapefiles (extent and tessellation) as well as the final daily spatio-temporal composite image created by the client in ArcView 3 and the tool are identical. Presuming therefore, that the composite image created by client previously in ArcView 3 is accurate, it can be concluded that the composite image crated by the tool, too is accurate.

6.2.4 Conclusions

The averaging functions were all found to produce accurate results. This was concluded after testing each averaging function (i.e. 2-day to 5-day) using a number of different datasets. The problem with the clipping operation producing inaccurate results was not resolved. Although this is a fairly serious problem and one that should be investigated further, it should be remembered that this problem will only occur when the points associated with one date are spatially located among points of different dates, so that when Thiessen polygons are created and dissolved by date they create discontinuous polygons. When data is sampled by continually moving to new areas, as is usual, this problem will not occur; it arises only when surveyors re-visit regions already sampled. The final test for accuracy by comparing composite images created manually by the client and automatically by the tool showed that the two were almost identical, the only difference being a slight variation in the extent of the images.

6.3 Performance Evaluation

The final test conducted was to evaluate the performance of the tool. It should be noted that the performance was not a focus during the tool's development and there were no stipulations made by the client in this regard. These experiments were merely conducted to try and evaluate the gain from having the tool.

6.3.1 Total Time for Creating Composite Image Manually vs. Using the Tool

The aim of this experiment was to evaluate the amount of time that is saved by using the tool over performing the same task manually in ArcMap.

Seven point shapefiles were used as input and for each a composite image was created manually in ArcMap and another by using the tool. The shapefiles ranged in size i.e. the number of actual points and the number of different days, as these would affect how long it would take to create the composite images. All composite images were created for a daily resolution. The time for the tool was measured from the time the first screen is opened to the time the final results screen is displayed and therefore includes the time taken for the user to enter all the required information as well as the actual processing time. The results of this experiment are shown in Table 6.1 below.

Shapefile	No. of points	No. of days	Total Time (Manual)	Total Time (Tool)
A074	27	3	10 min, 42 sec	1 min, 25 sec
1015	47	10	14 min, 03 sec	2 min, 09 sec
B030	97	7	24 min, 46 sec	2 min, 30 sec
A00028	107	17	38 min, 37 sec	3 min, 12 sec
A065	165	10	39 min, 22 sec	2 min, 59 sec
A083	207	12	40 min, 45 sec	3 min, 27 sec
A092	272	18	45 min, 24 sec	4 min, 22 sec

Table 6.1 Times taken to create composite images manually and by using the tool

What is clear from the above results is that regardless of whether the composite image was created manually or using the tool, the time taken increases as number of points and days in the input shapefile increases. The results also show that for all the shapefiles, the time taken to create the composite image using the tool was far quicker than doing it manually. There was a saving of just over nine minutes for the smallest shapefile which only had 27 points spanning three days, while for the largest shapefile which had 272 points spanning 18 days, 41 minutes were saved.

When creating this image manually, I had at hand a detailed list of steps for all the operations. A user attempting to create this image without such detailed instructions would most probably take much longer. This extra time would obviously depend on the user's prior ArcMap experience. For example, when I started the project and was attempting to create the composite image manually before starting to develop the tool, I spent a number of hours trying to work out how to perform some of the operations manually.

It should also be noted that while the times taken to create the tessellations around the points using the functions I wrote for the tool took merely a matter of seconds, when doing this manually, a DLL is used. This DLL takes a fair amount of time and increases quite substantially as the number of points in the shapefile increases. This will obviously have a big influence on the total time taken for the manual operations.

6.3.2 Performance of the Tessellation Operation using DLL vs. the Tool

When creating a composite image manually in ArcMap, a DLL downloaded from the ESRI website is used to perform the tessellation. As has been discussed previously, as this was a DLL, the source code was not available and I was forced to write a function from scratch to perform the tessellation. A small test was done to compare the performance of the DLL versus my own code. When running the DLL, the total time taken to perform the tessellation was given as part of the results and so these

times were easily available and used for the experiment. To calculate the time taken to perform the tessellation in my own code, markers were placed just before the start and on completion of the 'tessellation' functions and the time measured for the program to run between these markers. The same shapefiles used in the previous experiment were used for this test. The results are shown in table 6.2 below.

Shapefile	No. of points	Time for Tessellation using DLL	Time for Tessellation using my own code
A074	27	3 min, 25 sec	11 sec
1015	47	1 min, 7 sec	23 sec
B030	97	6 min, 46 sec	39 sec
A00028	107	17 min, 48 sec	40 sec
A065	165	20 min, 12 sec	1 min, 01 sec
A083	207	21 min, 17 sec	1 min, 15 sec
A092	272	20 min, 59 sec	1 min, 37 sec

Table 6.2 Times taken perform the tessellations using the DLL and my own code

As with the previous experiment, the total times for the tessellations using both methods generally increased as the number of points in the shapefile increased. What is fairly surprising and which is obvious from the results above, is how much quicker my own code took to perform the tessellations compared to the DLL.

6.3.3 Tool's Comparative Performance at Different Resolutions

A final test was done to evaluate the tool's performance when creating composite images at different resolutions. Four of the shapefiles used in the experiment in section 6.3.1 were used here. In addition to the daily resolutions already measured, the time taken to complete the composite images at 2-day, 3-day, 4-day and 5-day resolution were also measured and these shown in Table 6.3 below.

Resolution	B030	A00028	10015	A065
Daily	2 min, 30 sec	3 min, 12 sec	1 min, 55 sec	2 min, 59 sec
2-day	2 min, 06 sec	2 min, 48 sec	2 min, 06 sec	3 min, 11 sec
3-day	1 min, 49 sec	2 min, 27 sec	2 min, 08 sec	3 min, 08 sec
4-day	1 min, 42 sec	2 min, 52 sec	2 min, 01 sec	2 min, 54 sec
5-day	1 min, 39 sec	2 min, 11 sec	1 min, 43 sec	2 min, 39 sec

Table 6.3 Times taken to create composite images at different resolutions

The results from this experiment show that the performance of the tool remains fairly consistent regardless of the resolution used. Using a daily resolution means that some operations (e.g. clipping the satellite image with the 'day' polygon) must be repeated for each date in the shapefile which increases the time, however no averaging

functions are involved. The higher resolutions will involve averaging the satellite images which will add to the total time, but time will then be saved because other operations are not repeated for every date. For the shapefiles tested above, creating a composite image at a 5-day resolution was done the fastest.

6.3.4 Conclusions

In conclusion, the experiments have shown that the tool is much faster at creating composite images than those created manually in ArcMap. This holds true for shapefiles of a range of sizes (number of points and number of different days). It was also shown that the code written to perform the tessellation specifically for our tool out-performed the DLL which would be used when creating the composite images manually. Finally, a small experiment showed that the performance of the tool remained consistent regardless of what resolution a composite image is created at.

6.4 Summary

Testing the system involved robustness tests, accuracy tests and a performance evaluation. During robustness testing, the tool was run using a large number of different shapefiles from various sources as input in an attempt to identify any bugs or problems in the code or design of the tool. A number of problems were identified. Some of these were easily fixed, while others (e.g. missing dates in input shapefile, incorrectly georeferenced datapoints) were identified by the client as being beyond the scope of the application to account for and hence no fix was implemented. Although there is no way to foresee every problem that may occur, performing such tests did hopefully go a long way to identifying and fixing the majority of difficulties that might arise in practice. Accuracy testing showed that the tool produced accurate results for its various outputs including the 'averaging' operations, the 'clipping' operations and the final composite image. The only problem that was found was when the raster images were clipped with discontinuous polygons and the resulting clipped image contained pixels lying outside the polygon. No explanation or solution to the problem was found, which could be due to a bug in the interface or method used in the clipping operation. A final test was done to assess the performance of the tool. Although performance requirements were not part of the specification, reasonable run times were clearly desirable and so this was measured during testing. Tests showed that significant amounts of time are saved by using the tool over manually creating a composite image. Together, the three sets of tests therefore show that the tool is fast and produces accurate results. The extensive testing on different shapefiles during validation means that the tool should be able to be used with any input shapefile without encountering any problems.

7 User Evaluation

Not only is it important to test a system's functionality as was discussed in the previous chapter, but it is also very important to test its usability. The details of the user testing conducted on the tool are discussed in this chapter.

7.1 Aims

The tool was developed in order to automate a process that when performed manually was very time-consuming, complicated and difficult for users without considerable prior GIS experience. The aim of the evaluation was twofold: firstly to evaluate the user interface of the tool and secondly to evaluate the success of the tool in making it usable for both experienced and inexperienced GIS users.

7.1.1 Interface Evaluation

The first aim of the usability tests was to evaluate the actual user interface of the tool. This would include determining aspects of the interface that could potentially cause difficulties for users, how much they liked it and where improvements could be made. The best way to perform this evaluation would be to get the users to actually use the tool to create a composite image. This would allow them not only to see the different screens of the interface, but also use them in the context of a real example.

7.1.2 Ease of Using the Tool

The second aim of the evaluation was to evaluate the success of the tool for inexperienced GIS users i.e. to determine how difficult or easy they found the tool. Although an evaluation of the users using the tool alone would go some way to gauge its success, a better method would be to attempt to compare the ease with which users found performing a task manually using the GIS software (ArcMap) to using the tool.

7.2 Methods and Experiment Details

7.2.1 Evaluation Rationale

Evaluation of the tool or GIS software required using the appropriate GIS software (ArcView 8.3). To use this software requires relevant licences which are expensive. Even at university there are a limited number of computers in any one location which have such licences. The possibility of conducting experiments or evaluations with a number of users concurrently was therefore never a viable option and it was decided to perform evaluations with a single user at a time. Added to this, performing the evaluations one-on-one would allow the users to be closely observed which wouldn't be possible were there many users at the same time.

As one of the reasons for developing the tool was so that inexperienced GIS users could easily create the composite images, it was decided to obtain users who had a range of GIS experience. Users would also be selected with a range of backgrounds and computer experience as this may affect how easy they find performing the tasks. For example, computer scientists may find using new software much easier than people with little experience using computers, and people from a scientific background may understand the concept of a composite image better than those from a non-science background.

Since it was very unlikely that any potential users would have heard of a spatio-temporal composite image, it would therefore require first explaining to them the concept. Added to this, since some of the potential users may never have used a GIS before, they would require at least some explanation and guidance when performing the tasks, and given the known difficulty of using GIS software without any training it was suspected that some users would require a fair amount of guidance. For these reasons, it was decided that the evaluations would have to involve at least some interaction with the users.

Taking into account all the above considerations, it was decided to set the users a task using the tool to create a composite image. From this the user would be required to evaluate the interface. They would then be required to complete a task manually. They would be timed for each task to see how long it took them to complete. They would also be observed while performing the tasks and notes would be taken covering what questions they asked, comments they made and steps which they struggled with. Once the tasks were completed, users would be asked questions which would include both qualitative and quantitative information.

A pilot run of the evaluation was conducted on two users in order to get feedback on and make improvements to the structure and layout of the questionnaire, the task itself and the instructions for the task. Once changes had been made to the experiment and questionnaire, an attempt would be made to evaluate at least ten users. On completion of the evaluation, appropriate changes, if necessary, would be made to the interface based on observations and the user's comments and suggestions. If any major changes were made, a follow-up evaluation would be conducted to determine whether the changes were effective.

The tasks and questionnaire are shown in Appendix C. The information sheet given to the users with screenshots of all the screens in the tool is shown in Appendix D.

7.2.2 Introduction to Evaluation

The evaluation began by asking the users a number of questions aimed at getting background information including their computer and GIS experience. Those users that had not heard of or used a GIS previously were given a brief overview of how it works, including showing them what layers are, how they are displayed, and how different operations could be performed on the data/layers. A brief overview was given to all users of the concept of a spatio-temporal composite image. This included showing the users in ArcMap what a potential input shapefile would look like i.e. a set of point data with date values, how these points could be converted into polygons for each date and then how satellite images would be clipped using these areas and then merged to create a composite image. A brief outline was also given on the potential problem with cloud cover and how this could be reduced by increasing the temporal resolution. Once the user had a basic understanding of what was being done and the purpose of the tool, the evaluation tasks could begin.

7.2.3 Task 1. Interface Evaluation

The first task involved the users creating a composite image using the tool. Since these users would most likely not have their own data or shapefiles that could be used as input, they were provided with all the required files. The name and location of these shapefiles as well as the parameters that were to be used for the image were given to the users in the instructions. The point shapefile that was provided would require confirmation from the user to clarify the format of the dates. This was chosen specifically so that the screen used to do so could be evaluated.

Users were observed while they were performing the task and any operations that they appeared to struggle with were noted. The users were allowed to ask questions for any problems they had; these questions were also noted. Users were also timed to see how long they took to complete the task.

After completing the first task the users were required to answer questions about the interface. Users were asked if they had difficulties with any of the screens. Screen shots of each screen were given to the users when answering this question. Users were asked to make comments and suggest improvements.

7.2.4 Task 2. Tool vs. Manual Evaluation

The second part of the evaluation tried to evaluate the tool as a whole i.e. how easy it was to use and how easy it was compared to a manual task. First users were asked questions on ease of use based on their experience with the first task.

The second aspect was to determine how easy users found performing the tasks manually in ArcMap. Users had to complete some of the operations that are done when creating a composite image e.g. displaying the input point shapefile and land polygon shapefile, creating a buffer area around the points, dissolving Thiessen polygons by date etc. The users were encouraged to use the help function if they encountered problems.

Users were also timed to see how long they would take. Because completing all the tasks manually may have taken a very long time for some users, it was decided that instead of timing each user to see how long they took to complete everything, they would be given 20 minutes and it would be noted how many tasks they had completed in full in this time.

On completion users were asked how easy they found performing the manual tasks and asked to compare this to using the tool. The times recorded for the manual and tool tasks were also compared.

7.2.5 Follow-up Evaluation

Since there were a number of changes made to the interface a follow-up evaluation was conducted. A number of users, different to those used in the initial evaluation were chosen. An attempt was made to find users with a range of backgrounds. These users performed exactly the same task using the tool from the initial experiment, except that the tool and interface now included the changes. As with the initial evaluation, they were given a brief overview of GIS and spatio-temporal composite images beforehand. Users were timed as well being observed and any questions they asked or comments they made noted. On completion of the task they were asked the same questions from the initial evaluation.

7.3 Results

The results from the initial and follow-up evaluations are discussed below.

7.3.1 Users' Backgrounds

Eleven users were tested in the first evaluation. The users came from a range of backgrounds including professionals working in Science, professionals working in IT, professionals working in other fields, undergraduate and postgraduate students studying Marine Biology/Zoology and a student completing a BA. The age of the users ranged from 23 to 53 years.

The users' responses to their level of expertise at computers are shown in Figure 7.1 Most of the users indicated that they felt very confident using computers. Since most of these users are working in IT or GIS related fields or are students doing Science degrees this was not surprising since they are likely to be using computers on a daily basis for a range of applications. The users that were fairly confident and the single user that felt not very confident tended to be those whose professions/degrees did not require much use of computers e.g. medical doctor, BA student.

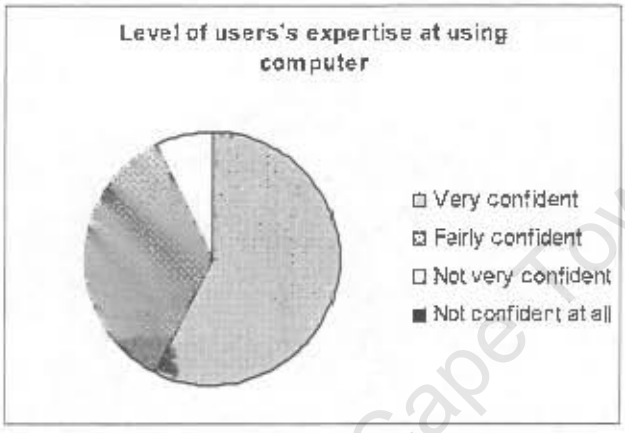


Figure 7.1 Users' level of computer expertise

The users were asked if they knew of GIS. The results of this question are illustrated in Figure 7.2.

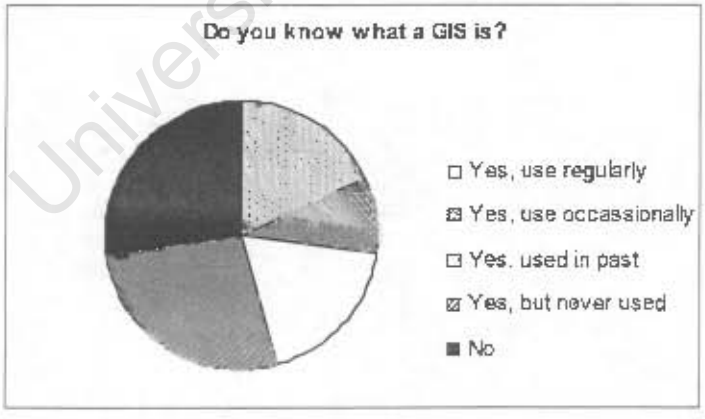


Figure 7.2 Users familiarity of GIS

Of the five users that had used a GIS before, when asked whether they had used ArcGIS in particular, they all had or still did. This suggests that ArcGIS software is widely used.

7.3.2 Interface Evaluation

Users' responses to problems they had with the different screens are shown in Figure 7.3. Screens 3, 4 and 6 presented no problems to the users, while for screens 1, 2 and 5, users did experience some problems.

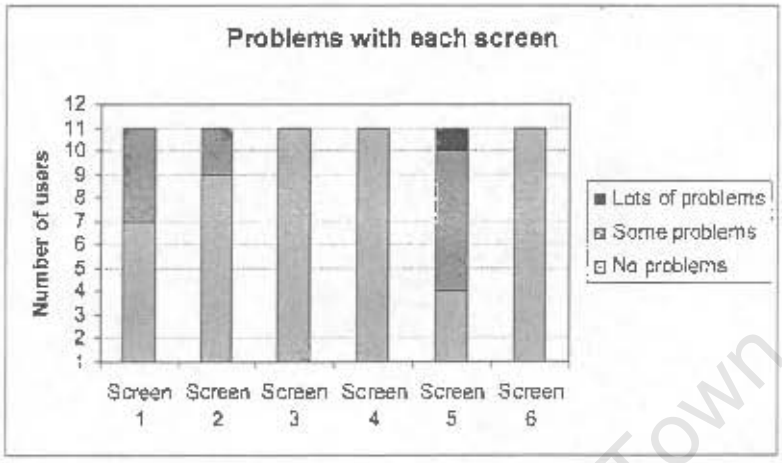


Figure 7.3 Users' responses to which screens they had problems with

Through these responses, the comments made by the users during the task and in the questionnaire afterwards, and the direct observation made of the users, the following issues concerning each screen were noted:

Screen1 (Input/Output files):
All but one of the users had no problems browsing for the input file and output folder. This is probably because the users are familiar with the Common Dialogs used for browsing as they are standard Windows features.

When selecting the actual input file, four of the users were unsure of which file to select, confused by the different extensions. They were looking for the file name without any extensions. Three of these were those that had never used a GIS.

Screen 2 (Check for land polygon):
All but two users used the 'Browse' button to indicate the land polygon shapefile rather than entering the name of the file in the text box. Those users that selected the 'Browsing' option did not have any problems. Both the users who entered the name in the text box had a problem: one user entered the shapefile name (southafrica) with a space between the 'south' and 'africa' which caused an error, while the other user entered the name with a capital letter (e.g. Southafrica) which also caused an error. It had to be pointed out to these users that the name had to be entered exactly as it was written down. The one user succeeded on the second attempt while the other opted to

browse for the file instead. These are the same two users who indicated they had some problems with this screen in Figure 7.5 (screen 2).

Screen 3 (Select field names) and Screen 4 (Select parameters):
No problems were encountered by any of the users for either of these screens.

Screen 5 (Format date field):
The date the users were given was: 1995-10-22 00:00:00
They needed to enter the format as: yyyyxmmxddxxxxxxxxx
(This was the original interface for this screen and not the one used in the final product)

This screen presented the most problems for the users, with seven out of the eleven users indicating they had some or lots of problems with this screen. Of those users that indicated problems, three commented that the instructions were not very clear. Other questions asked by the users were:

- whether the format needed to be entered as capital letters
- whether the time part of the date needed to be entered as well

While entering the format, a count was done for each user as to the number of attempts it took to enter the format correctly. Only four users entered it correctly the first time round as shown in Figure 7.4 below.

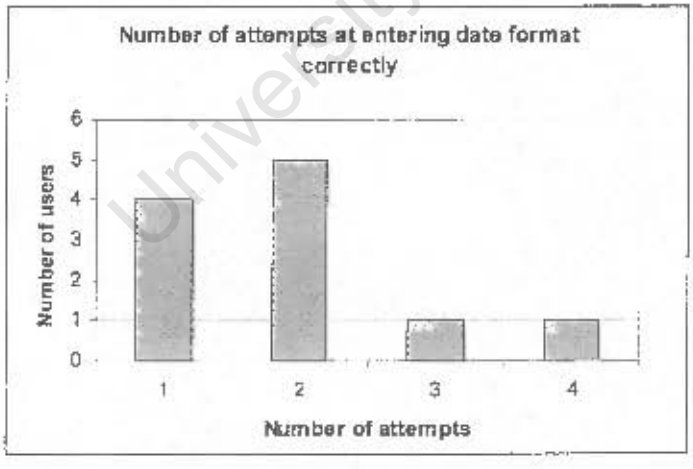


Figure 7.4 Number of attempts by users to enter format correctly

The format entered by these unsuccessful users was incorrect for a range of reasons including:

- 'x' not entered for the space between the 'date' and 'time'.
- Incorrect number of 'x's entered for the time.
- '-' entered instead of 'x'.
- General mistyping e.g. entering only three 'y's instead of four.

Screen 6 (Composite image completed):
No problems were encountered by any of the users for this screen.

General comments made by the users for improvements to the screens included:

- When the users have to browse for a shapefile, to display only the shapefile name without any extensions, rather than having the different file extensions all displayed.
- Telling the user that there can't be spaces when typing in the land polygon shapefile name
- Provide a clearer explanation on how to enter the format of the date.

After the information is entered on Screen 5, it takes a bit of time for the composite image to be made. Many of the users were unsure what was happening i.e. whether the computer was still processing or whether it had frozen. Some of the comments made by the users in this regard included 'How do I know if I pressed the 'Next' button', 'Is the computer thinking?', 'Is the computer busy?' and 'How long must I wait?'. A number of users suggested that the user should be told that the computer is busy. Suggestions for this included a progress bar or just a message telling the user that will have to wait a while for the process to complete.

7.3.3 Ease of using Tool

Performing task with tool

Generally users had very few problems using the tool. The time taken by the users to complete the task varied from 5 to 31 minutes, the averaging being 10 minutes and 18 seconds. In comparison I completed the task in 2 minutes and 30 seconds. Although the users' times are far greater than my time, it should be remembered that while the users were completing the task they were asking me questions which added some time to the total. The range in times can also be explained by the different levels of experience using computers and GIS.

Users were asked how easy they found using the tool in general. Their responses to this question are shown in Figure 7.5. It can be seen that seven users indicated 'Very easy', three indicated 'Easy' and only a single user indicated 'Not very easy'. The user that indicated 'Not very easy' was the user that expressed a very low level of expertise at using a computer. The groups of users indicating 'Very easy' and 'Easy' consisted of users with a range of GIS experience. This suggests that prior GIS experience did not guarantee that users encountered no problems (problems entering the date format probably accounted for this). It also indicates that users without prior GIS experience still found the tool easy to use.

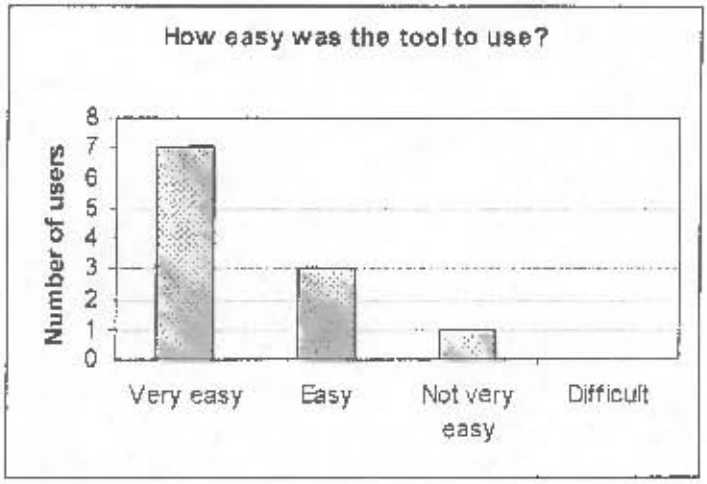


Figure 7.5 Users' responses to how easy the tool was to use

Performing the task manually

Generally the users did not get very far in the manual task, the number of tasks completed varying from three to seven out of eight, the average being 4.7 (i.e. step four). The results of this are shown in Figure 7.6. The two users who used GIS regularly completed the most steps. Both these users did however, get stuck on the last task, which was clipping the satellite image with a polygon. This is a fairly complicated operation and requires a bit of research or reading up to determine what to do. Most of the users asked lots of questions about how to do things. I tried not to answer their questions directly but encouraged them to use the Help function. If, after some time they were still stuck, I pointed them in the right direction. Apart from the two regular GIS users, everyone found this task fairly to very difficult. Even the users who had used ArcMap in the past, battled with some of the steps.

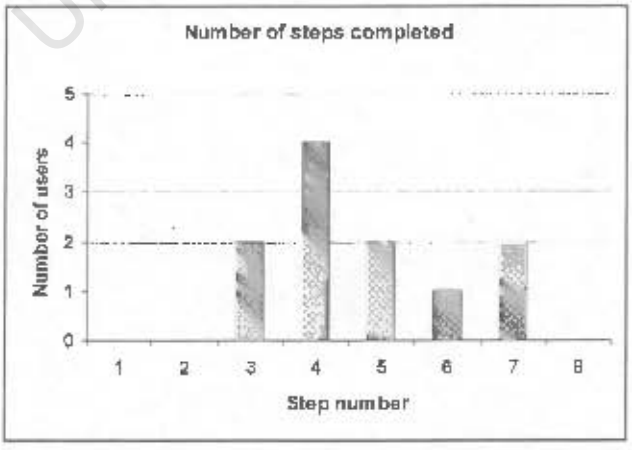


Figure 7.6 Number of steps completed for all users for the manual task

When asked to rate how easy they found the task, none of the users found it very easy. The two regular GIS users indicated it was fairly easy with a few problems, while the remainder of the users thought it was not every easy or difficult. The number of users choosing each category is show in Figure 7.4 below.

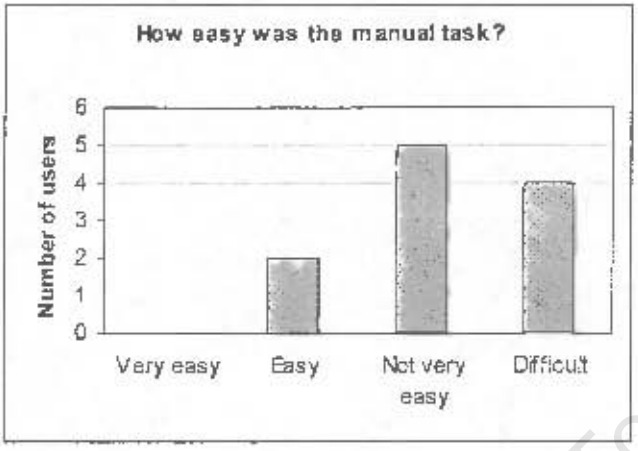


Figure 7.4 Users' responses to how easy the manual task was

Comparison

A comparison of the users' responses to how easy the manual task and tool were to use (Figure 7.8) shows how the users seemed to find the tool much easier to use. This was confirmed by the fact that when asked this in question 7, all the users indicated that the tool was much easier. Some of the users' responses as to reasons for saying the tool was easier to use included:

- 'I only had to click a few buttons and choose a few things in the tool. In the other task nothing was easy to do'
- 'It took me ages to figure out how to do the tasks in ArcMap. Even using Help was difficult because I didn't understand properly what I was supposed to be doing. The tool was much easier because I didn't have to figure out anything, I just had to click buttons and everything was done for me'
- 'When I was using the tool, I knew exactly what to do, and if I made a mistake there was a message so I knew what I'd done wrong. Things in ArcMap weren't so obvious. I had to look around a long time before I knew what to do'
- 'Even though I have used ArcMap before, some of the tasks I had no clue how to do. In the tool, I didn't need to find out how to do anything'

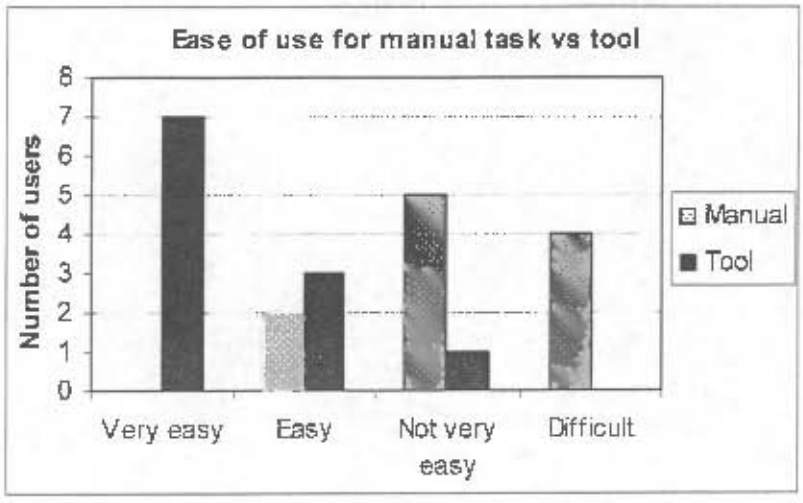


Figure 7.8 Comparing users ratings of ease of use of manual task versus the tool

7.3.4 General

When the users were asked if they would use this tool in the future, seven users indicated 'yes, no problem', four indicated 'yes, but would prefer someone to help or check' and only one said they would ask someone else instead. The user indicating they would ask someone else instead was the user who indicated they were not confident using computers, while the users who indicated they were confident in using the tool in the future without any help, were generally those that had used a GIS before.

The final question asked the users if they thought the tool could be applied in their own subject or field. Almost all the users said no. One user who is studying pelicans, including their population dynamics and movements as part of her PhD said she could see how the tool could be useful, but for her research it would not be applicable. Part of her research may involve tracking the pelican's movements and although this data could be potentially used as the input point data, she did not think that any environmental parameters that could be measured by satellite images would affect the pelican's movement.

7.4 Conclusions from Initial Evaluation

Conclusions that can be drawn from the results of this evaluation include:

1) Users found the manual task difficult, confirming the fact that using a GIS by inexperienced or untrained users is not easy.
2) Users found the tool easy to use, and indicated that is was much easier than the manual task.

In general, the design of the interface for the tool was considered adequate, although there were a few screens where problems were identified and changes or improvements were made. These were:

1) Users were confused when browsing and selecting shapefiles as they didn't know which file (extension) to select. The screen was therefore changed and a note included telling the user that they could select any of the files of the correct name, regardless of the extension.
2) Users had problems when entering the shapefile name in the text box to indicate the land polygon shapefile. This option was subsequently removed from the interface.
3) Users did not have a clear understanding of what they had to do when required to indicate the format of the date. The instructions on this screen were therefore changed so that this was clearer. Two examples were also added to the screen which showed a date and the format it should be entered as. One of these examples is shown in the screen shot below.

In attempt to make the instructions even clearer, the use of an 'X' to indicate 'non-date' characters was changed to a '*'. This was used because '*'s are often used when typing passwords for example, where they indicate the position of a character which can't be seen
4) Users were unsure of when the computer was busy processing. A message was therefore added asking the user to please wait a few moments while the computer is busy

7.5 Follow-up Evaluation

Once these changes had been made, a follow-up was done with three of the users. The users chosen were those that had the most problems during the evaluation. Because the follow-up was done a while after the original evaluation, the users were shown the original screens and reminded of the problems they had, and then shown the modified screens. They were then asked to comment on whether they thought the improvements were adequate. All three users indicated that this was the case, with one of the users commenting that the instructions on the 'Date format' screen were much clearer and a vast improvement.

In addition to following up with these users, six new users were recruited to perform the first task from the initial evaluation using the tool with the changes. Two of these users are doing post-graduate degrees in Marine Biology; both these users indicated

they were confident using computers and had used a GIS in the past but had not used it recently. Three users are professionals working in non-scientific fields. Two of these users indicated they were fairly confident and the other one very confident using computers. All three had heard of a GIS but hadn't used one. The final user is a bar manager. He indicated he was not very confident using computers and had never heard of a GIS.

Users performed the first task and were then asked to indicate the problems they had with each of the screens. The results are shown in Figure 7.9 below.

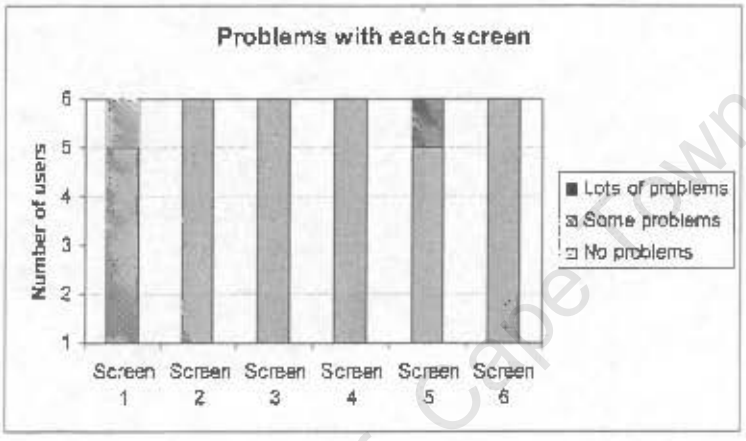


Figure 7.9 Problems users had with each screen in follow-up evaluation

From the results shown above, observations of the users and comments they made while performing the task and comments users made in the questionnaire after completing the task, the following points should be noted:

- Only a single user indicated they had some problems with first screen. This user does not use computers regularly and battled slightly browsing for the correct file as he was looking for the file without an extension.
- In Screen 2, unlike in the initial evaluation, the option of indicating the shapefile by typing the name in a text box had been removed and all users had to use browse for the file. No users had any problems with this; the users who battled with Screen 1 now knew exactly what he was doing.
- As in the initial evaluation, no users had any problems with Screens 3 or 4.
- Users generally took a bit of time reading the instructions on screen 5. A single user started asking me a question about what to do, however I asked them to read the instructions carefully and attempt to proceed without my help. All but one user managed to enter the format correctly the first time round. The user who entered the format incorrectly, had entered one to few

'*'s. I questioned him about this and found that he had not entered a * for the space between the date and time, even though this was included in the instructions.

The results of this follow-up evaluation have shown that the changes made to the interface do reduce the number of problems the users have using them. Removing the text box option for indicating the shapefile in Screen 2 not only makes the screen less cluttered but also eliminates the chance of experiencing problems that may be encountered when typing the name in the textbox. Added to this when the user has to indicate the file and folder in Screen 1, both only have 'Browse' options, and so in an attempt to keep the screens consistent it is appropriate to only have the Browse option on Screen 2.

The changes made to Screen 5 where users must indicate the format of the date, appear to have made a big difference to the users being able to understand what is required of them. Although users need to read the instructions carefully which may take a bit of time, they appear to be much better understood.

The final change made to the interface was including a message asking the user to please wait while the computer was busy processing. None of the users in the follow-up evaluation made any comments or asked any questions about having to wait or whether the computer was still busy, suggesting that the including the message was a successful addition to the interface.

7.6 Conclusions

In conclusion therefore, the tool can be considered a success in terms of its ease of use for both experienced and non-experienced GIS users. This was in stark contrast to performing manual tasks in ArcMap which users generally found very difficult. The evaluation of the user interface revealed that the design and information on the screens was detailed and clear enough so that the users knew what they were doing thus making the tool easy to use. A few areas where improvements could be made were identified by users. Changes were subsequently made to address these problems and included in the interface of the final product. A follow-up evaluation indicated that the changes made to the interface were a success and reduced the number of problems users encountered.

8 Conclusions

Some of the conclusions that can be drawn with regard to this project are discussed in this final section. After summarizing the achievements and success of the work, we review the main problems encountered with ArcObjects and end with ideas for future improvements.

8.1 Summary of Main Achievements

On completion of this project it was found that it was feasible to develop a tool that could automatically create spatio-temporal composite images. It was also possible to develop the tool generically so that it could be used for a variety of input data and not just data of the client at MCM. The user is therefore able to provide any input point data with dates and choose the type of satellite image and resolution with which they wish to create the image, and the tool will automatically format the data in the input shapefile correctly, check if there are satellite images available for the required dates, and if so create a spatio-temporal composite image at the specified resolution. It was also found that using the tool significantly reduced the amount of time taken to create the images compared to doing it manually and that the tool could be easily used by even inexperienced GIS users. Finally, although the client at MCM never made too many stipulations as to how he wanted the tool or its interface designed, he was happy with the final interface and output. The final confirmation that the tool successfully created accurate spatio-temporal composite images was when a comparison was done to an image he had created using ArcView 3 and it was found that the two were identical.

8.2 Success of the Project

The system can be considered a success in terms of both its functionality and usability. As was discussed, it has been developed so that it is generic. All possible variation in the format and contents of the input shapefile has been accounted for in the design so that the tool should run successfully for any input data. After testing the tool using a number of different shapefiles as input, it appears to be bug-free. In terms of accuracy and correctness, all the averaging functions produce accurate results and the final composite image appears as it should. The only problem that was not resolved was when satellite images were clipped using non-continuous polygons. This produced a clipped image where there were pixels that lay outside the extent of the polygon. This appears to be an inherent problem in the ArcObjects method as all possible reasons for this occurring were investigated including posting on the ESRI forums, without any success. The tool can also be considered a success in terms of its performance: the time taken to create a composite image was shown to be much faster

than creating a composite image manually in ArcMap. For a particular dataset, I was able to create an image in 1 minute and 25 seconds using the tool, compared to 10 minutes 42 seconds doing it manually in ArcMap.

In terms of usability, the tool was found to be easy to use, regardless of a user's background and prior experience using GIS. This was in sharp contrast to performing the tasks manually in ArcMap which users found very difficult. The user interface was kept simple and consistent. There were a few aspects of the interface which users had some problems with. Slight modifications were therefore made to the design which are now incorporated into the final product. In a usability test on the final product, one subject experienced a slight problem and the others had no difficulty at all.

One of the aims when developing the tool was to automate the process of creating the composite images as much as possible. One of the biggest obstacles for doing this was deciding on the best way to find the correct satellite images (as determined by the dates of the input point data and the type of satellite image selected by the user). One possible solution was to require the user to indicate the name and location of each image required. This however would require extensive input from the user which could take a considerable amount of time, especially for large datasets which may require the names of dozens of images In order to automate this potentially tedious task, it was decided to make use of a metadatabase to check for image availability and location. The metadatabase contains details of all images that are available to be used including the type of image (e.g. STT, chlorophyll), its source, its name, date and where it is saved. Using this information the tool automatically checks through all the images to determine if images of the required type and date are available.

A further aim when developing the tool was to make it generic as possible so that input point data from any source and of any format could be used. All possible measures were taken to identify and include features that would make the tool generic so that any point data could be used as input. These features include:

- Allowing the input shapefile to have any name (the tool checks for potential name problems e.g. length/spaces and remedies them).
- Allowing the shapefile to contain single or multiple datasets
- The field that represents the 'date' and the field that distinguishes the different datasets (if a multiple dataset) can have any name.
- The dates can be typed as Strings or Dates and can be in any format.

- The input shapefile can contain points with exactly the same spatial location. Duplicate point cause problems when performing the tessellation however the tool detects and removes these.
- The points in the shapefile do not need to be in chronological order
- Any satellite images that can be used in a GIS are supported and the tool detects if these differ in extent or resolution.

After including all these features in the tool and testing it using a number of different sources of input data it was found to run successfully for each test. The tool can therefore be considered a success in terms of making it generic.

8.3 Problems with ArcObjects

There were a number of ArcObject methods that did not appear to work correctly. The problems encountered were:

1) When using the *ConvertToVoronoiRegions* method to create Thiessen polygons around the points, gaps and overlaps between polygons were present. These caused problems when dissolving adjacent polygons. Although a DLL is provided on the ESRI website to create these polygons, this could not be used for the tool as the source code is not available. This problem was overcome by writing my own function to create the Thiessen polygons. What is interesting to note is that my function appeared to work much more efficiently than the DLL. As an example, for one point shapefile, using the DLL to create Thiessen polygons took 6 minutes and 45 seconds, while my function took 21 seconds to complete.
2) After the Thiessen polygons have been created it is required to clip them with the 'buffer area'. When this clip operation was done through the *ITopologicalOperator Interface*, there were slivers present in the output, suggesting a possible fault here. When the *IBasicGeoprocessor Interface* was used to perform the clipping, the output was fine and no slivers were present.
3) When using the *IExtractionOp Interface* to clip the satellite images using polygons, when the polygons were not continuous, there was an error in the output in that pixels were included in the clipped image that lay outside the polygon. No solution to this problem was found.
4) The Mosaic method from *IRasterGeometryProc Interface* did not produce accurate results. The values of the pixels in the output did not appear to be averaged even though it says that the method 'determines the value of an output call by averaging the values of the overlapping input cells' [ADH04]. This problem was overcome by writing my own function to 'mosaic' or average images.

Although all the above presented problems for me, I cannot say for certain that they are bugs in the ArcObject methods. For all issues described above, I did however make a concerted effort to try and identify and fix the problems by searching extensively on the ERSI forums as well as posting my own questions there. I did find that other people frequently encountered similar problems also posting these on the forums. In general, the ESRI forums were a very helpful resource although there were postings I made which either got no response or the suggested solutions did not directly solve my problems. Apart from the problem with the *IExtractionOp Interface*, clip operation for which no solution was found, for all the other problems encountered alternate methods were used or the functions were written from scratch to perform the operations.

8.4 Limitations and Future Improvements

As was mentioned the tool has been developed to use a metadatabase to check for image availability. Although this feature does have the benefit of automating the process of finding appropriate images, the metadatabase does need to be kept up-to-date otherwise new images which are available may be missed. Metadatabases are however, useful features to have in that they allow information on large amount of data to be recorded for easy referencing. They are also useful for people to browse or query to check what's available and get details on the files e.g. resolution and format.

The tool was developed specifically for MCM, which deals with marine based data only and for the most part their data is located along coastlines, and specifically along the South African coastline. The user is thus required to provide a land polygon in order to clip the buffer area to an extent that would lie over the sea only. Data located around islands or in mid-ocean can be used without any problems as long as a polygon is supplied for the clipping operation. For islands, all the individual land masses in the island group can be represented as a single polygon and so this does not present a problem. For mid-ocean data, a polygon is still required, so a polygon must be supplied which has different location to the data. In this case, although the clipping function will still run, no actual clipping will occur because the data and polygon do not overlap spatially. Using the tool for land based data would in essence be no different than using it for data located in the ocean. As long as it was point data with date attributes and there were satellite images available measuring a relevant parameter it could be used in exactly the same manner.

As has been discussed previously the tool was implemented so that only daily satellite images are used to create the composite images. It may however occur, as was the case with the images at MCM, that there may be temporal composite images already available to be used. These would normally be 5-day, weekly or monthly composite images. Why not then allow the tool to use these images as well? The main reason for not implementing this feature is that in most cases the user would choose to create the composite image at a daily resolution. Using 5-day temporal resolution and hence the 5-day composites would almost defeat the point of creating the spatio-temporal composite images. The amount of additional coding required to implement a feature that would most likely very seldom be used did not seem practical. Having said that however, it could occur that for the dates of a particular dataset, there may be no daily images available while there may be 5-day composites. In this case, it would be better to use these 5-day composites rather than not have any images available and therefore not being able to create any sort of composite image. In the event that a 5-day resolution is acceptable and if there were often 5-day or weekly composite images available for dates where no daily images were available, then it would definitely be worthwhile adding this feature to the tool.

8.5 Summary

This project investigated the feasibility of creating a tool to generate spatio-temporal composite images automatically. We discovered that although there were a number of problems with several ArcObject methods, these could be overcome by writing one's own code to perform the operations. The successful running of the final product showed that it was feasible to develop a tool to automate the process. It has also been shown that the tool has met the stipulated requirements: the process of creating spatio-temporal composite images has been automated so that minimal input is required of the user; the images are created quickly and the time taken is far less than would be possible for even the most experienced GIS user using the normal GIS interface; the output of the tool i.e. the composite images are correct and accurate; the tool was extensively tested and free of errors; many feature have successfully been included in the tool to make it generic; the tool has a simple, easy to use interface and finally the tool was easily used by even inexperienced GIS users.

References

[ADH04] ArcObject Developers Help. ESRI. 2005. http://arcobjectsonline.esri.com/

[Bar99] Barale, V. Integrated Geographical and Environmental Remotely-sensed Data on Marginal and Enclosed Basins: The Mediterranean Case. Chapter 13. In: *Marine and Coastal Geographical Information Systems*. Ed. Wright, D. and Bartlett, D. Taylor & Francis, London, 1999.

[BH98] Brodziak, J. and Hendrickson, L. An analysis of environmental effects on survey catches of squids, *Loligo pealei* and *Illex illecebrosus* in the northwest Atlantic. *Fish. Bull* 97:9-24, 1998.

[Bou89] Bourke, P. Efficient Triangulation Algorithm Suitable for Terrain Modelling. 1989. http://astronomy.swin.edu.au/~pbourke /terrain /triangulate/. Accessed 21 March 2005.

[CL99] Cataldo, M and Labini, G.S. Monitoring Bluefin tuna habits with the help of satellite environmental data: ERS experience waiting for ENVISAT. 1999. http://envisat.esa.int/pub/ESA_DOC/gothenburg /191Catal.pdf. Accessed 11April 2005.

[Col99] Cole, J. Environmental conditions, satellite imagery, and clupeoid recruitment in the northern Benguela upwelling system. *Fisheries Oceanography* 8(1):1-25. 1999. http://www.blackwellsynergy.com /doi/abs/10.1111/j .1365-2419.2004.00324.x. Accessed 11 April 2005.

[CV00] Cole, J. and Villacastin, C. Sea surface temperature variability in the northern Benguela upwelling system, and implications for fisheries research. *International Journal of Remote Sensing* 21(8): 597-1617. 2000. http://journalsonline.tandf.co.uk/app/ home/contribution.asp?wasp=5a9baf1ae7334c2fb400bcb5341dbc88&referrer=parent&backto=issue,4,18;journal,110,169;linkingpublicationresults,1:100669,1. Accessed 7 February 2005.

[DF00] Demarcq, H. and Faure, V. Coastal upwelling and associated retention indices derived from satellite SST. Application to *Octopus vulgaris* recruitment. *Oceanologica Acta* 23(4): 391- 408. 2000. http://www.cephbase.utmb.edu/refdb/pdf/6968.pdf. Accessed 7 April 2005.

[Dra03] Drago, A. Addressing the need of marine observations for fisheries. CapeMalta.Net. 2003. http://www.capemalta.net/medgoos/docs/APS%20Paper.pdf. Accessed 18 April 2005.

[Dud03] Dudycha, D.J. Scope of Remote Sensing. 2003. http://www.fes.uwaterloo.ca/crs/ geog165/scopers.htm. Accessed 16 April 2005.

[ESR05] ESRI Support Center. GIS Dictionary. 2005. http://support.esri.com/index.cfm?fa=knowledgebase.gisDictionary.gateway Accessed 13 April 2005.

[GIS05] GIS.com. The Guide to Geographic Information Systems. 2005. http://www.gis.com/whatisgis/. Accessed 12 March 2005.

[Gre01] Greuter, G. IextractionOp bugs. July 24 2001, Message posted to http://forums.esri.com/Thread.asp?c=93&f=992&t=44554#108168 Accessed 12 June 2005.

[GMR04] Gulf of Maine Research Institute. GMA: Space Available: Remote Sensing. 2004. http://octopus.gma.org/surfing/sensing/sensing.html Accessed 18 April 2005.

[Gui00] Guilmoto, C.Z. The spatial analysis of sub-population, technical paper prepared for the workshop on socio-cultural factors affecting demographic behaviour, Unesco and UNFPS, Paris. 2000. http://www.demographie.net/ird/guilmoto/unesco.pdf. Accessed 18 April 2005.

[Hal04] Halpern, K. The 2004 ESRI LBS Conference: A Global Carrier Launch Comparison. Pg 1. 2004. http://www.jlocationservices.com/LBSArticles/ ESRI%20UC%202004%20Article.pdf. Accessed 23 April 2005.

[Hon01] Hong. Applying an existing Colormap. April 24 2001, Message posted to http://forums.esri.com/ Thread. asp?c=91&f=915 &t= 39654#96697. Accessed 30 November 2004.

[JAR96] Japan Association of Remote Sensing. Concept of Remote Sensing. 1996. http://www.profc.udec.cl/~gabriel/tutoriales/rsnote/cp1/cp1-1.htm. Accessed 16 April 2005.

[Kin01] King, S.D. Solway Firth GDSPDS - Remote Sensing. 2001. http://www.abdn.ac.uk/ ~geo402/rs.htm. Accessed 19 April 2005.

[Kul00] Kuliniewicz, P. Windows API Guide: Glossary. Pietschsoft. 2000. www.pietschsoft.com/programming/vbapi/ref/glossary.html. Accessed 11 September 2004.

[Lie01] Liew, S. D. Principles of Remote Sensing - Centre for Remote Imaging, Sensing and Processing, CRISP. 2001. http://www.crisp. nus.edu.sg/~research/tutorial/ spacebrn.htm. Accessed 16April 2005.

[Lom03] Lomas, T. ArcScripts Details – ESRI Support. Create Thiessen Polygons 3.0. 2003. http://arcscripts.esri.com/details.asp?dbid=11958 Accessed 7 October 2004.

[Luc99] Lucas, A. Representation of Variability in Marine Environmental Data. Chapter 5. In: *Marine and Coastal Geographical Information Systems.* Ed. Wright, D. and Bartlett. D. Taylor & Francis, London. 1999.

[Mea99] Meaden, G.J. Applications of GIS to Fisheries Management. Chapter 15. In: *Marine and Coastal Geographical Information Systems*. Ed. Wright, D. and Bartlett. D. Taylor & Francis, London. 1999.

[MC05a] Microsoft Corporation. The Microsoft Scripting Runtime Object Library. 2005. http://msdn.microsoft.com/library/ default .asp ?url= /library/en-us/modcore/ html/ deconthemicrosoftscriptingruntime objectlibrary.asp. Accessed 24 September 2004.

[MC05b] Microsoft Corporation. Microsoft OLE DB Provider for Microsoft Jet. 2005. http://msdn.microsoft.com/library/default.asp?url=/library/en-us/ado270/htm/ mdrefjetprovspec.asp. Accessed 21 October 2004.

[Nev05] Nevo, A. How to avoid/fix/ gaps/slivers in Voronoi polygons. 28 January 2005. Message posted to http://forums.esri.com/Thread.asp ?c=93&f=993&t=91773#253418. Accessed 3 October 2004.

[NOA03] NOAA. Fisheries Northeast Fisheries Science Center. Large Marine Ecosystems of the World: LME 29: Benguela Current. 2003. http://na.nefsc.noaa.gov/lme/ text/lme29.htm. Accessed 19 April 2005.

[OA98] Oceans Alive. The Scientist at Sea. Remote Sensing. 1998. http://www.mos.org/oceans/scientist/sensing.html. Accessed 19 April 2005.

[PHH98] Painting, S., Hutchings, L., Huggett, J.A., Korrûbell J.L., Richardson, A.J. and Verheye, H.M. Environmental and biological monitoring for forecasting anchovy recruitment in the southern Benguela upwelling region. *Fisheries Oceanography* 7(3-4):364. 1998.

[RGH98] Rantajävi, E., Gran, G., Hällfors, S. and Olsonen, R. Effects of environmental factors on the phytoplankton community in the Gulf of Finland – unattended high frequency measurements and multivariate analyses. *Hydrobiologia* 363(1-3): 27-139. 1998. http://www. springerlink.com/app/home/contribution.asp?wasp=ccccef98935524009 8281da6e544aa9ec&referrer=parent&backto=issue,11,30;journal,175, 196;linkingpublicationresults,1:100271,1. Accessed 14 April 2005.

[Sho04] Short, N.M. Remote Sensing Tutorial Introduction. 2004. http://rst.gsfc.nasa.gov/ Intro/Part2_1.html. Accessed 14April 2005.

[SO98] Shannon, L.V. and O'Toole, M.J. Integrated overview of the oceanography and environmental variability of the Benguela Current region. Second Regional Workshop of the Benguela Current Large Marine Ecosystem (BCLME). BCLME Thematic Report, 2. 58pp. 1998.

[Tc05a] Tchoukanski, I. Filling gaps between polygons? 21 November 2005. Message posted to http://forums.esri.com/ Thread.asp? c=93&f= 987 &t=79110#212410. Accessed 12 May 2005.

[Tc05b] Tchoukanski, I. Build Thiessen Polygons Wizard. 2005. http://www.ian-ko.com/ ET_GeoWizards /UserGuide /thiessen Polygons.htm. Accessed 1 October 2004.

[US05] Uniblue Systems. shell32.dll – DLL Information. 2005. http://www.liutilities.com/ products/wintaskspro/dlllibrary/shell32/ Accessed 24 May 2005.

[Val02] Valavanis, V.D. Geographic Information Systems in Oceanography and Fisheries. Taylor & Francis, London. 2002.

[WGI02] Wyoming Geographic Information Science Center. University of Wyoming. 2002. http://www.wygisc.uwyo.edu/gis. Accessed 14 April 2005.

[WQ00] Williams, E.H. and Quinn, T.J. Pacific herring, *Clupea pallasi*, recruitment in the Bering Sea and north-east Pacific Ocean, II: relationships to environmental variables and implications for forecasting. *Fisheries Oceanography* 9(4). 2000. http://www.blackwell- synergy.com/doi/abs/10.1046/ j.13652419 .2000.00146 .x?journalCode=fog. Accessed 19 April 2005.

[Zei02] Zeiler, M. Exploring ArcObjects Vol.1 – Applications and Cartography. Published by ESRI, Redlands. 2002.

Appendix A: Sample Code for Vector Data Processing

```
' creates a buffer around a set of points
Public Function CreateBuffer(sInputFileName As String) As IGeometry

    Dim pWorkspace As IWorkspace              ' set up workspace
    Dim pFact As IWorkspaceFactory
    Dim pWorkspace As IFeatureWorkspace

    Set pFact = New ShapefileWorkspaceFactory
    Set pWorkspace = pFact.OpenFromFile("c:\temp", 0)
    Set pWorkspace = pWorkspace

    Dim pFeatureclass As IFeatureClass
    Dim pFeatureCursor As IFeatureCursor
    Dim pFeature As IFeature

    Set pFeatureclass = pWorkspace.OpenFeatureClass(sInputFileName)
    Set pFeatureCursor = pFeatureclass.Search(Nothing, False)
    Set pFeature = pFeatureCursor.NextFeature

    ' Set up multipoint for constructing buffer
    Dim pMultiPoint As IMultipoint
    Dim pPointCollection As IPointCollection
    Dim pPoint As IPoint
    Dim pGeometry As IGeometry

    Set pMultiPoint = New Multipoint
    Set pPointCollection = pMultiPoint

' Loop through each point, project it and add it to the pointcollection
    Do While Not pFeature Is Nothing
        Set pGeometry = pFeature.Shape
        Set pPoint = pGeometry
        pPointCollection.AddPoint pPoint
        Set pFeature = pFeatureCursor.NextFeature
    Loop

' Set up for getting the buffer of the points
    Dim pTopo As ITopologicalOperator
    Dim pBufferGeo As IGeometry

    Set pTopo = pMultiPoint
    Set pBufferGeo = pTopo.Buffer(0.7)  'with the required buffer distance

    CreateBuffer = pBufferGeo  ' return the buffer geometry

' Memory cleanup
    Set pFeatureCursor = Nothing
    Set pFeature = Nothing
    Set pMultiPoint = Nothing
    Set pPointCollection = Nothing
    Set pGeometry = Nothing
    Set pPoint = Nothing

End Function
```

Appendix B: Sample Code for Raster Data Processing

```
Public Sub JoinTwoImages(sImageNameA As String, sImageNameB As String)

    ' set up raster workspace
    Dim pRasterWorkspaceFactory As IWorkspaceFactory
    Dim pRasterWorkspace As IRasterWorkspace
    Set pRasterWorkspaceFactory = New RasterWorkspaceFactory
    Set pRasterWorkspace = pRasterWorkspaceFactory.OpenFromFile("c:\temp\images", 0)

    ' open the rasters
    Dim pRDS1 As IRasterDataset
    Dim pRDS2 As IRasterDataset
    Set pRDS1 = pRasterWorkspace.OpenRasterDataset(sImageNameA)
    Set pRDS2 = pRasterWorkspace.OpenRasterDataset(sImageNameB)

    ' set up safeArray for first raster
    Dim pRBandCol As IRasterBandCollection

    Dim pRaster1 As IRaster
    Dim pRProps1 As IRasterProps
    Dim pSize1 As IPnt
    Dim pBlock1 As IPixelBlock
    Dim pRPnt1 As IPnt
    Dim pSA1 As Variant

    Set pRBandCol = pRDS1
    Set pRaster1 = pRDS1.CreateDefaultRaster
    Set pRProps1 = pRaster1
    Set pSize1 = New DblPnt
    pSize1.SetCoords pRProps1.Width, pRProps1.Height
    Set pBlock1 = pRaster1.CreatePixelBlock(pSize1)
    Set pRPnt1 = New DblPnt
    pRPnt1.SetCoords 0, 0
    pRaster1.Read pRPnt1, pBlock1
    pSA1 = pBlock1.SafeArray(0)
    ' set up safeArray for second raster
    Dim pRaster2 As IRaster
    Dim pRProps2 As IRasterProps
    Dim pSize2 As IPnt
    Dim pBlock2 As IPixelBlock
    Dim pRPnt2 As IPnt
    Dim pSA2 As Variant

    Set pRaster2 = pRDS2.CreateDefaultRaster
    Set pRProps2 = pRaster2
    Set pSize2 = New DblPnt
    pSize2.SetCoords pRProps2.Width, pRProps2.Height
    Set pBlock2 = pRaster2.CreatePixelBlock(pSize2)
    Set pRPnt2 = New DblPnt
    pRPnt2.SetCoords 0, 0
    pRaster2.Read pRPnt2, pBlock2
    pSA2 = pBlock2.SafeArray(0)

    'get parameters from one of the input images to create new raster
    Dim pPointOrig As esriCore.IPoint
    Set pPointOrig = New esriCore.Point
    pPointOrig.PutCoords pRProps1.Extent.xMin, pRProps1.Extent.yMin
    Set pPointOrig.SpatialReference = pRProps1.SpatialReference

    Dim lCol As Long
    Dim lRow As Long
    Dim csize As Double
    Dim lNumBands As Long
    Dim pOutSR As ISpatialReference

    Set pOutSR = pRProps1.SpatialReference  ' Get the spatial reference of the input
    lCol = pRProps1.Width                   ' number columns
    lRow = pRProps1.Height                  ' number rows
    csize = pRProps1.MeanCellSize.x         ' cell size
    lNumBands = 1                           ' number of bands
```

```
    ' create new raster data set
    Dim pRDSNew As IRasterDataset
    Dim pRasterNew As IRaster
    Dim pRPropsNew As IRasterProps
    Dim pBlockNew As IPixelBlock
    Dim pSANew As Variant
    Dim sNewName As String  'name for the new image
    sNewName = "NewImage"

    Set pRasterWorkspace = pOutRasterWorkspace
    Set pRDSNew = pRasterWorkspace.CreateRasterDataset(sNewName, "GRID", _
                pPointOrig, lCol, lRow, csize, _
                 csize, lNumBands, PT_FLOAT, pOutSR, True)

    ' set up safeArray for new raster
    Dim pBandNew As IRasterBand
    Dim pBandColNew As IRasterBandCollection
    Dim pRawPixelNew As IRawPixels
    Dim pSizeNew As IPnt

    Set pRasterNew = pRDSNew.CreateDefaultRaster
    Set pRPropsNew = pRasterNew
    Set pBandColNew = pRasterNew
    Set pBandNew = pBandColNew.Item(0)
    Set pRawPixelNew = pBandNew
    Set pSizeNew = New DblPnt
    pSizeNew.SetCoords pRPropsNew.Width, pRPropsNew.Height
    Set pBlockNew = pRawPixelNew.CreatePixelBlock(pSizeNew)
    pSANew = pBlockNew.SafeArray(0)

    'next section calculates the new pixel values to put in newArray
    Dim noDataValue As Variant                ' get the 'no data' value
    noDataValue = pRProps2.noDataValue(0)

   Dim vPixelValue1 As Variant        ' to hold the value of the pixel from ImageA
   Dim vPixelValue2 As Variant        ' to hold the value of the pixel from ImageB
   Dim vPixelValueNew As Variant      ' to hold the newly calculated pixel value
   Dim i As Long
   Dim j As Long

    'loop through each pixel in the two arrays to get their values
    For i = 0 To pSize1.x - 1
        For j = 0 To pSize1.y - 1
            vPixelValue1 = pSA1(i, j)
            vPixelValue2 = pSA2(i, j)
            vPixelValueNew = 0

           ' calculate new pixel value from the two values
            If Not vPixelValue1 = noDataValue And Not vPixelValue2 = noDataValue Then
               vPixelValueNew = (vPixelValue1 + vPixelValue2) / 2   'if both are data pixels
            Else      ' if only one has a data pixels, use that value
                If vPixelValue1 = noDataValue And Not vPixelValue2 = noDataValue Then
                    vPixelValueNew = vPixelValue2
                Else
                    If vPixelValue2 = noDataValue And Not vPixelValue1 = noDataValue Then
                        vPixelValueNew = vPixelValue1
                    Else     ' if both are no data, then set to no data
                        vPixelValueNew = pSA2(i, j)
                    End If
                End If
             End If

            pSANew(i, j) = vPixelValueNew
        Next j
    Next i

    ' Create a DblPnt to hold the top left corner
    Dim pPnt1 As IPnt
    Set pPnt1 = New DblPnt
    pPnt1.SetCoords 0, 0

    ' Write the PixelBlock to raster band
    pRawPixelNew.Write pPnt1, pBlockNew

End Sub
```

Appendix C: User Evaluation Questionnaire

Please answer these questions:

Name:
Age:
Profession:
If student, degree studying:

1. What do you consider your level of expertise at using a computer? (circle)

   A. I am very confident using computers. I find performing new tasks and using new programs/software easy
   B. I am fairy confident, I find performing new tasks/programs a bit of a challenge but don't mind doing them
   C. I am not very confident. I find performing new tasks/programs a big challenge
   D. I am not confident at all. I find performing new tasks/programs terrifying

2. Do you know what GIS (Geographical Information System) is?

   A. Yes, I use it regularly as part of my work or studying
   B. Yes, I use it occasionally as part of my work or studying
   C. Yes, I have used it in the past but not recently
   D. Yes, I have heard of it but never used it
   E. No, have never heard of it

If you circled A or B in the previous question, please answer the next question:

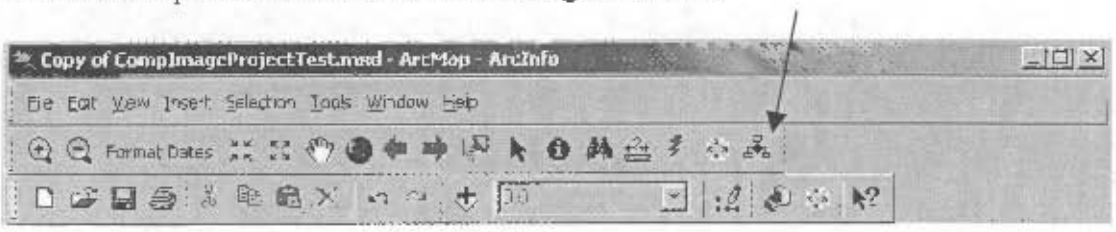
3. Have you used any of the ArcGIS products (e.g. ArcMap)?

   A. Yes, use it regularly
   B. Yes, but do not use it regularly
   C. I have never used it

< Explain and show to user basics of ArcMap if user is unfamiliar to GIS and to all users what a composite image is>

Task 1.

You are going to try and create a composite using the tool. The data you need is listed below. To start the tool, click this button  which appears on one of the tool bars in the ArcMap interface as shown in the diagram below.



Use the information given to you below to create the image. You will be faced with a series of screens. Enter the required information and click the 'Next' button to take you to the next screen.

Input:

- Input shapefile is called **B030** in is stored in **c:\temp**, you will use all the data in this shapefile
- Choose any folder for the output, c:\temp is fine
- Land polygon is called **southafrica** and stored in **c:\temp**
- The date field is called **'Date_"**
- You want to create a image for **temperature** at a **daily** resolution

Questions for task 1:

4. Did you have problems with any of the screens? Please rate each screen and comment on specific problems in the space afterwards

Screen 1: Input/Output files

A. No problems at all
B. Some things I had a problem with
C. I did not know what I was doing

Screen 2: Check for land polygon

A. No problems at all
B. Some things I had a problem with
C.I did not know what I was doing

Screen 3: Select field names

A. No problems at all
B. Some things I had a problem with
C.I did not know what I was doing

Screen 4: Select parameters

A. No problems at all
B. Some things I had a problem with
C.I did not know what I was doing

Screen 5: Format date field

A. No problems at all
B. Some things I had a problem with
C.I did not know what I was doing

Screen 6: Composite image completed

A. No problems at all
B. Some things I had a problem with
C.I did not know what I was doing

Comments about any of the screens above:

_____ _____
_____ _____
_____ _____
_____ _____

5. In general how easy did you find creating the composite image using the tool?

A. Very easy, no problems as all
B. Easy but a few things I was unsure about
C. Not very easy, lots of things I was unsure about
D. I didn't really know what I was doing

Task 2.
You are now going to perform a few simple tasks in ArcMap. You have 15 minutes, try and do as much as you can

<Give the user 15 minutes and see how far they get>
< note what step they get to>

The steps you must follow are listed below: (use default names when saving the files)

1) Add the **B030** and **southafrica** layers to ArcMap (located in **c:\temp)**
2) Create a buffer area around the points in **B030,** specify the distance at **40km**
3) Add the **B030_thiessen** layer to ArcMap (located in **c:\temp)**
4) Do a spatial join between the **tessellation(B030_thiessen)** layer and **B030** layer
5) **Dissolve** the polygons in the tessellation layer by date
6) Select first polygon in the dissolved layer and create a new layer from this polygon
7) Add the satellite image named **sst19961022 to ArcMap** (located in **c:\temp)**
8) **Clip** this image with the single polygon in the layer you created in step 6.

Questions for task 2:

6. How easy did you find completing the above tasks manually using ArcMap?

A. Very easy, no problems as all
B. Easy but a few things I was unsure about
C. Not very easy, lots of things I was unsure about
D. I didn't really know what I was doing

7. How easy to use was the tool compared to using the tasks you performed in ArcMap interface?

A. The tool was much easier
B. The tool was slightly easier
C. About the same
D. Using ArcMap was easier

Explain:

_____

_____

General questions:

8. Any other comments about the tool in general or suggestions for improvements?

_____

_____

9. Would you use this tool in the future if you were asked to produce such an image?

A. Yes, no problem
B. Yes, but would prefer someone to help or check me
C. No, would ask someone else instead

10. Are you aware of any way in which the tool could be applied in your own subject or field?

A. Yes
B. No

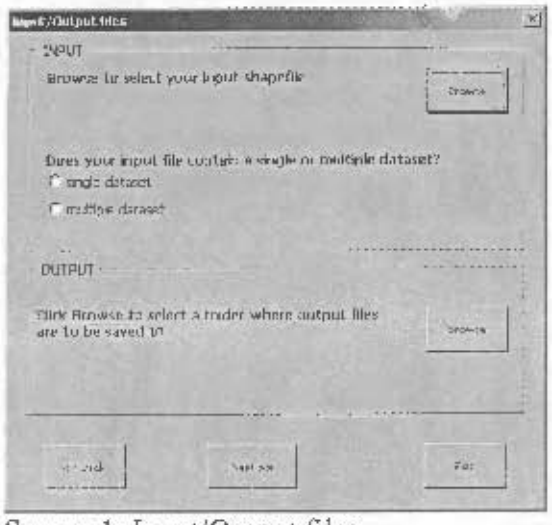
If Yes, explain

_____

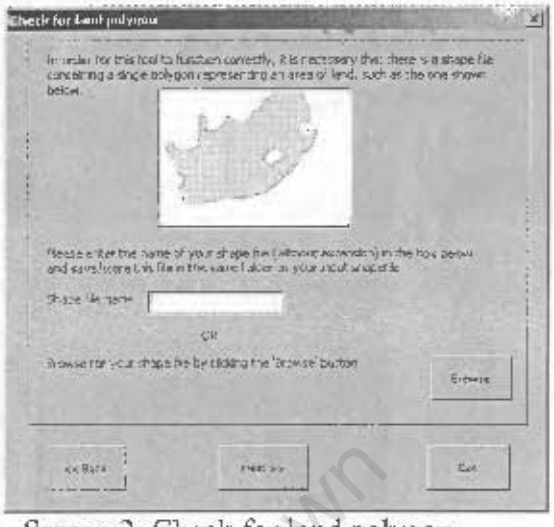
_____

Thank-you for taking your time to help evaluate this tool

Appendix D: Screen Shots used in Initial User Evaluation



Screen 1: Input/Output files



Screen 2: Check for land polygon

Screen 3: Select field names

Screen 4: Select parameters

Screen 5: Format date field

Screen 6: Composite image completed