# Forced Landing Technologies for Unmanned Aerial Vehicles: Towards Safer Operations

Dr Luis Mejias[1], Dr Daniel Fitzgerald[2], Pillar Eng[1] and Xi Liu[1]
*Australian Research Centre for Aerospace Automation*[3]
*Australia*

## 1. Abstract

While using unmanned systems in combat is not new, what will be new in the foreseeable future is how such systems are used and integrated in the civilian space. The potential use of Unmanned Aerial Vehicles in civil and commercial applications is becoming a fact, and is receiving considerable attention by industry and the research community. The majority of Unmanned Aerial Vehicles performing civilian tasks are restricted to flying only in segregated space, and not within the National Airspace. The areas that UAVs are restricted to flying in are typically not above populated areas, which in turn are one of the areas most useful for civilian applications. The reasoning behind the current restrictions is mainly due to the fact that current UAV technologies are not able to demonstrate an Equivalent Level of Safety to manned aircraft, particularly in the case of an engine failure which would require an *emergency* or *forced landing.*

This chapter will preset and guide the reader through a number of developments that would facilitate the integration of UAVs into the National Airspace. Algorithms for UAV Sense-and-Avoid and Force Landings are recognized as two major enabling technologies that will allow the integration of UAVs in the civilian airspace.

The following sections will describe some of the techniques that are currently being tested at the Australian Research Centre for Aerospace Automation (ARCAA), which places emphasis on the detection of candidate landing sites using computer vision, the planning of the descent path/trajectory for the UAV, and the decision making process behind the selection of the final landing site.

## 2. Introduction

The team at the Australian Research Centre for Aerospace Automation (ARCAA) has been researching UAV systems that aims to overcome many of the current impediments facing the widespread integration of UAVs into civillian airspace. One of these impediments that the group identified in 2003 was how to allow a UAV to perform an emergency landing.

[1] Dr Luis Mejias, Pillar Eng and Xi Liu are with the Queensland University of Technology
[2] Dr Daniel Fitzgerald is with the ICT centre CSIRO
[3] ARCAA is a joint venture between Queensland University of Technology and CSIRO

An emergency or forced landing (in the case of unpowered flight), is where the aircraft is required to perform an unplanned landing due to the occurrence of some onboard emergency, such as an engine failure. This capability is an inherent component for benchmarking the performance of the manned aviation industry, therefore the group identified this as a key impediment to overcome in order to allow UAVs to operate over populated areas in civilian airspace (Fitzgerald, Walker et al. 2005; Fitzgerald 2007). Hence, it is believed that UAVs must therefore be provided with the ability to safely terminate the flight through a range of emergency scenarios. A UAV plummeting uncontrollably into the middle of a busy freeway or a school yard is a risk that the public will be unwilling to except. In this context, a UAV emergency landing system will be an important component towards enabling routine missions in civilian environments.

To date, no commercial system is available that allows a UAV to decide autonomously on the safest area to land in an unknown environment. Safety systems currently available to UAVs only allow the aircraft to fly towards a pre-defined safe landing area from a database of known safe landing locations. However, these systems must be preprogrammed with up-to-date information, thus requiring a continuous communications link between a human operator and the air vehicle to ensure that the latter will not attempt to land at an unsuitable location.

An alternative would be to have a system onboard the UAV that can process information in a similar way to a human pilot in emergency situations that require the aircraft to land. Therefore, the objective of this research is to develop an onboard capability that allows the UAV to select a suitable landing site, and then manoeuvre the UAV autonomously to land at this location. If this functionality is realised, it will bring UAVs one step closer to flying in civilian airspace above populated areas.

The research described in (Fitzgerald 2007) has reduced the technical risks for a vision-based emergency landing system and hence in the past year, a number of research programs have begun at ARCAA to complement this research. It has now been proposed that a complete prototype system suitable for flight trials be developed. A range of flight test scenarios will be evaluated on the prototype system (encompassing a range of altitudes and different terrain), and will be conducted with the relevant approvals from the Civil Aviation Safety Authority (CASA) of Australia.

This chapter will describe the different research programs and results to date, and how these will be integrated to form a complete prototype system ready for flight testing.

These research programs can be classified into the three broad areas of:

- Visual identification and classification of UAV forced landing sites;
- Trajectory planning and tracking for autonomous aircraft forced landings; and
- Multilevel decision-making for high-level reasoning during the descent

These form the basis of this chapter and their use in developing a real-time implementation and system for flight testing.

## 3. Machine Vision Landing Site Selection

Over a 3 year period a computer vision based approach has been developed and optimised. This approach, which mimics human processes, identifies emergency landing sites that are obstacle free. More specifically, the landing sites are chosen based on their size, shape and slope as well as their surface type classification. Subsequent algorithms have been developed that allow automatic classification of the candidate landing site's surface (based on back propagation neural networks). At the time of writing we are implementing the approach described in this section in real time and are conducting flight trials of a small UAV to further demonstrate this approach.

The remainder of this section will describe this vision-based techniques to find potential UAV landings sites from aerial imagery. For a detailed description of this work please refer to (Fitzgerald 2007).

### 3.1 Candidate Landing Site Selection Framework

The aim for the selection of candidate landing sites for the UAV forced landing problem is to locate regions from aerial imagery that were of similar texture, free of obstacles and large enough to land in. Regions that met the criteria would be further classified according to their surface type to aid in the choice of the most suitable landing region.

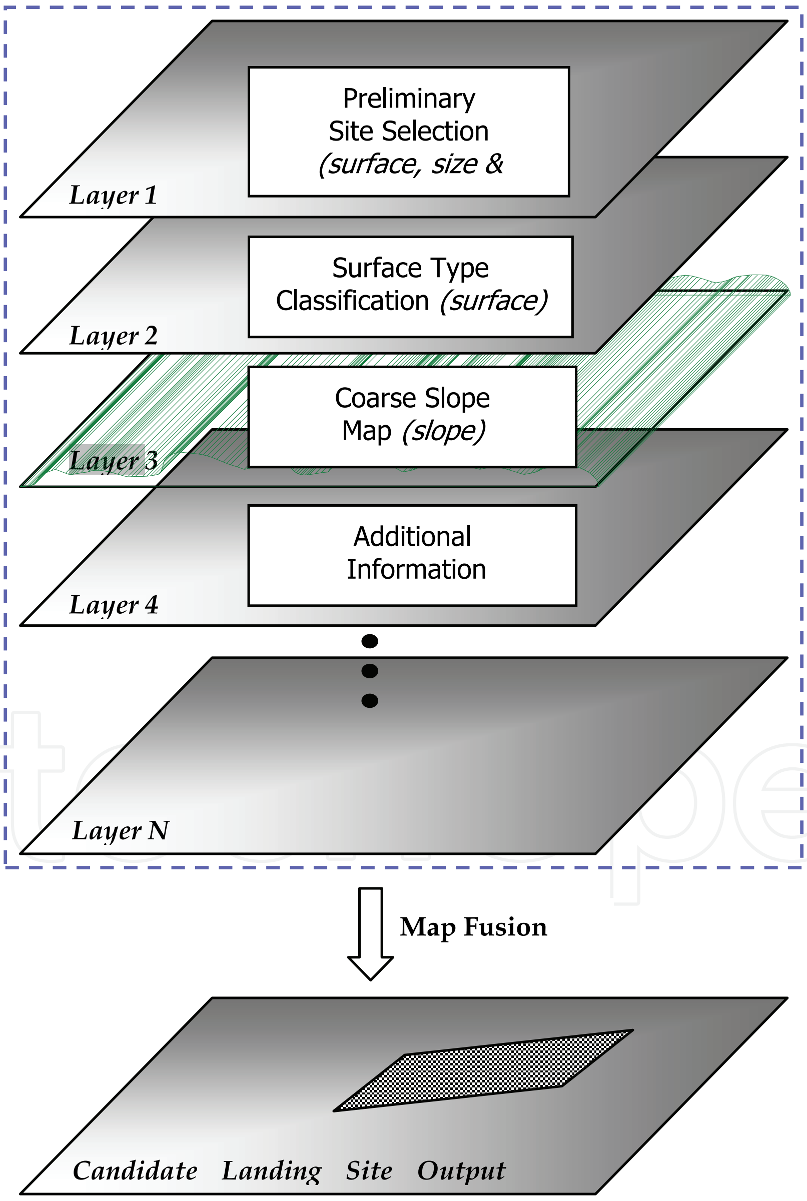

Preliminary Site Selection (surface, size &
Layer 1
Surface Type Classification (surface)
Layer 2
Coarse Slope Map (slope)
Layer 3
Additional Information
Layer 4
Layer N
Map Fusion
Candidate Landing Site Output

Figure 1. Candidate Landing Site Selection Architecture

This criterion for UAV landing site selection is based on the elements that a human pilot considers during a forced landing. Section 4 describes a number of factors that a human pilot considers when selecting the most appropriate landing site - Size; Shape; Slope; Surface; Surroundings; and S(c)ivilisation. These factors are more commonly known as the six S' and are cues used by pilots when selecting a landing site during a forced landing. The approach that has been developed for selecting a candidate landing site for a UAV forced landing has been approached with these factors.

With these human factors in mind, Figure 1 illustrates the architecture of the approach developed to locate candidate UAV forced landing sites.

This architecture incorporates many of the elements that human pilots use when deciding upon the most suitable forced landing site. It also leaves open the ability to add additional layers of information at a later stage if information is available. Information from each layer can be fused together by using a number of linguistic or fuzzy rules. For example areas free of objects (identified in the Preliminary Site Selection layer), with a *flat* slope (Coarse Slope Map layer) and classified as *grass* (Surface Type Classification layer) would be given a *very safe* output map value. This final output map has similarities to the T-Hazard map in (Howard & Seraji, 2001, 2004) where a number of data sources are fused together with linguistic rules.

The remaining indicators used by human pilots in a forced landing situation, that are not considered in the architecture of Figure 1 are civilisation, wind and surroundings.

Civilisation is not a valid indicator for choosing candidate landing sites for the UAV forced landing problem, as the civilisation criterion is an indication of proximity to life critical services for a human pilot. It is important for a human to be as close as is safely possible to populated areas, so that any medical attention can be administered as quickly as possible after a forced landing. A UAV does not have this problem as there are no humans on board. The concern for a UAV forced landing is to avoid people entirely. Here the best option is to head for large areas free of obstacles.

Wind is the final additional indictor included by (CASA, 2001) that is not covered by the traditional six S' used by pilots when selecting a suitable landing site during a forced landing. Knowledge of wind direction is important for a pilot's decision on the final approach direction (for a conventional landing) and also to determine the maximum range the aircraft is able to glide to. A tailwind can increase the distance an aircraft is able to glide, however a head wind will reduce this range. Wind will be discussed in Section 5.

Errors in the maximum glide range estimate of the UAV introduced by wind has been mitigated by introducing a buffer between the theoretical maximum glide distance (based on the particular UAV) and the range of landing sites included in the output of the coarse slope map generation discussed next. Any decrease in the ranges to potential landing sites considered can be used. A figure of 85 % of the theoretical maximum range distance has been used for this research. Additionally, a decreasing scale can also be adopted for the inclusion of landing sites as you move away from the UAV's current position. The result is that suitable areas close by can be given more weight that areas towards the extremities of the UAV's glide range.

Finally, surroundings refer to the identification of objects in the image such as trees, buildings and powerlines. For the human forced landing case, objects such as powerlines or fences often cannot be seen, but are inferred by other objects in the image. For example, the presence of a house or building is likely to indicate a power line nearby, just as a road is likely to indicate the presence of fences near by. Humans use this knowledge in the choice for an appropriate approach path to the final landing site.

The elements discussed above are part of the multilevel decision making and determining the most appropriate approach path to the chosen landing site. These areas of research are covered in Sections 4 and 5 of this chapter, respectively.

The information from the 3 (or more) layers can be fused together by a set of linguitic rules or by some other technique resulting in a weighted map of potential landing sites. A higher level decision making process (described in Section 4) will take these information to decide the best landing site.

### 3.1.1 Preliminary Site Selection

The preliminary site selection layer is responsible for extracting regions from the aerial image that are large enough for a UAV forced landing and that do not contain obstacles. The approach extracts these areas directly, without the need for image segmentation. This results in a process that is fast and suits the forced landing application specifically.

The approach was broken down into two steps – a region sectioning and a geometric acceptance. These techniques now will be described briefly.

*Region Sectioning Phase*

The region sectioning phase is responsible for finding areas in the image that are of similar texture and that are free of objects. The approach uses two measures that are augmented together to create a map of suitable areas.

The first measure uses a well known technique, the Canny edge detector [Canny, 1986] on the entire image, followed by a line expansion algorithm. It was observed then further assumed that regions in the image that contained no edges corresponded to areas that contained no obstacles. Additionally, since boundaries between different objects – for instance grass and bitumen – usually have a distinct border, areas with no edges should corresponded to areas of similar texture (ie: the same object, for example a grass field).

This assumption was made after studying a number of edge gradient maps similar to the one shown in Figure 2. Peaks in the plot correspond to edges in the image. The figure shows clear evidence that the areas free of obstacles in Test Image 1 (refer top Figure 4) correspond to areas with a low number of edges. Subsequent observations on different images were made to verify this assumption. Additionally, clear borders (corresponding the edges in the Canny edge detection image) could be observed between different regions in the image which is also desired for the forced landing problem.

A line expansion algorithm immediately follows the edge detection, and involves the examination of the pixels of all edges found. For each pixel found, the algorithm inspects the surrounding pixels within a certain search radius. If another edge pixel is found, the algorithm will set all pixels within this radius to a "1". This is shown below in Figure 3.

This calculation is performed by knowing how much distance each pixel equates to on the ground (pixel ground resolution). Based on a number of assumptions the pixel ground resolution for the image can be determined from:

- Height above ground (for example: 2500 ft; approx 762 meters);
- Image dimensions (for example: 720 x 576 pixels);
- Camera viewing angle (for example: 35.0 x 26.1 degrees).

The pixel resolution values are used at different stages of the algorithm to determine measures such as the landing site pixel dimensions and line expansion radius values. These can be altered according to the landing requirements of the UAV – dependant on the UAV class.

The edge detection measuring layer outputs a layer map that contains a 1 for every pixel location corresponding to an edge. The following figure shows a number of images with the edge detection measure shown (Canny edge detection plus the line expansion algorithm).

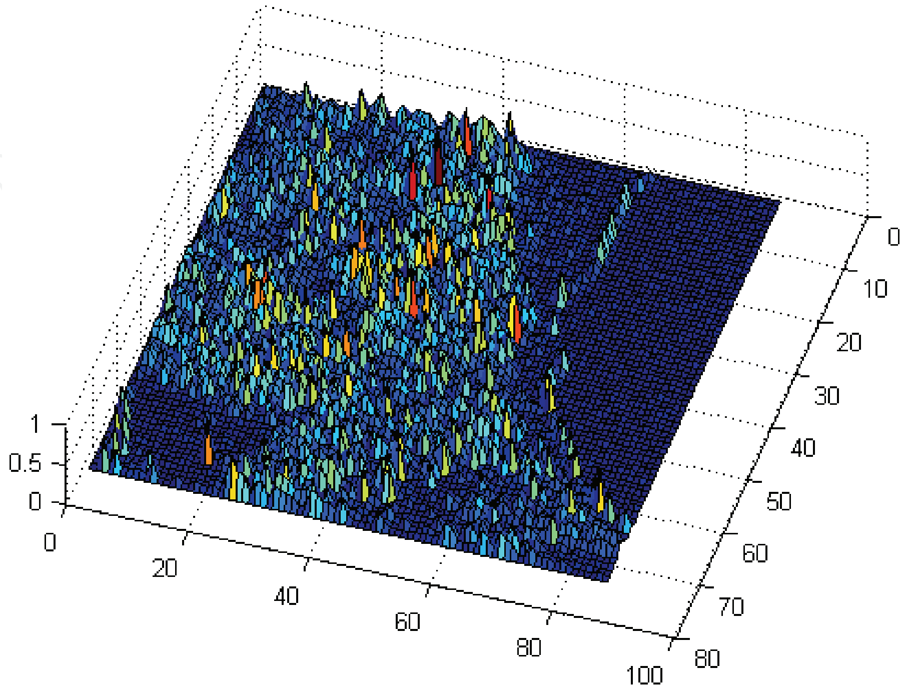
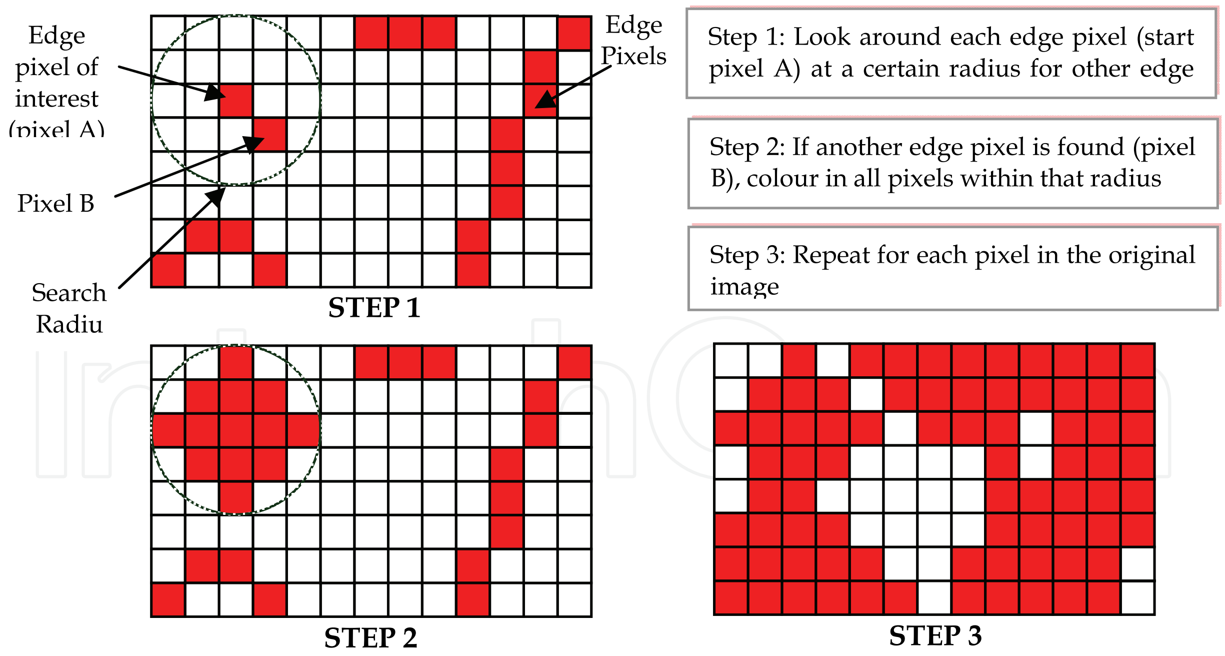


Figure 2. Edge Gradient Map of Test Image 1



Step 1: Look around each edge pixel (start pixel A) at a certain radius for other edge

Step 2: If another edge pixel is found (pixel B), colour in all pixels within that radius

Step 3: Repeat for each pixel in the original image

Figure 3. Line Expansion Algorithm

This final step of the algorithm ensures a suitable boundary is placed between obstacles detected and potentially safe areas to land in. The search radius size in this algorithm can be altered depending on the UAV's height above ground level, to maintain this suitable safety zone.

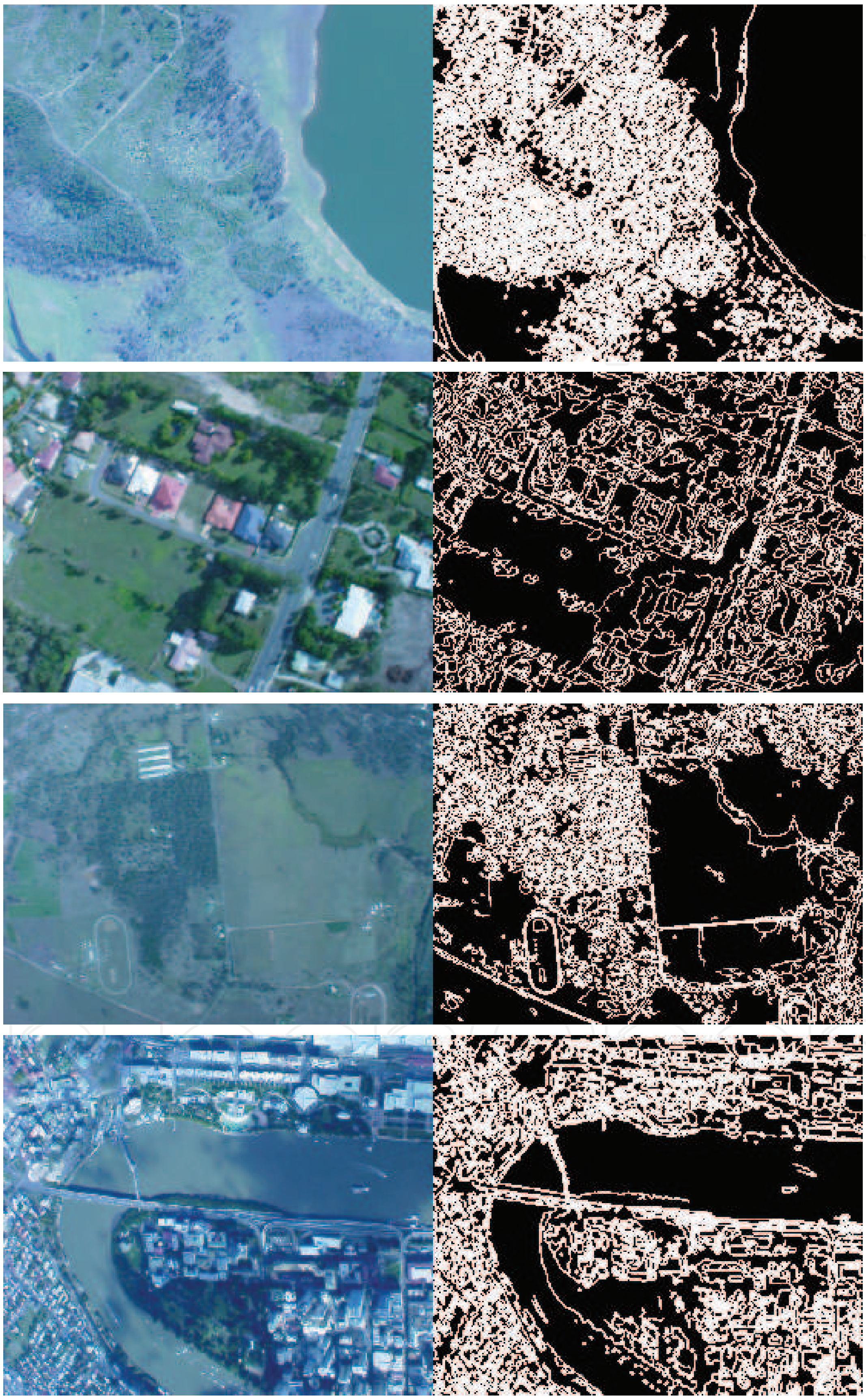


Figure 4. Edge Detection Measure Output for a Number of Test Images

The second measure was an intensity measure that was required to help distinguish between man-made and natural objects in the image. It was observed through experimentation that man-made regions in the images often exhibited the highest intensity values in the image. These objects included roof tops, cars and sheds – anything that had a high reflectance. Natural surfaces such as grass and trees generally were of lower intensity.

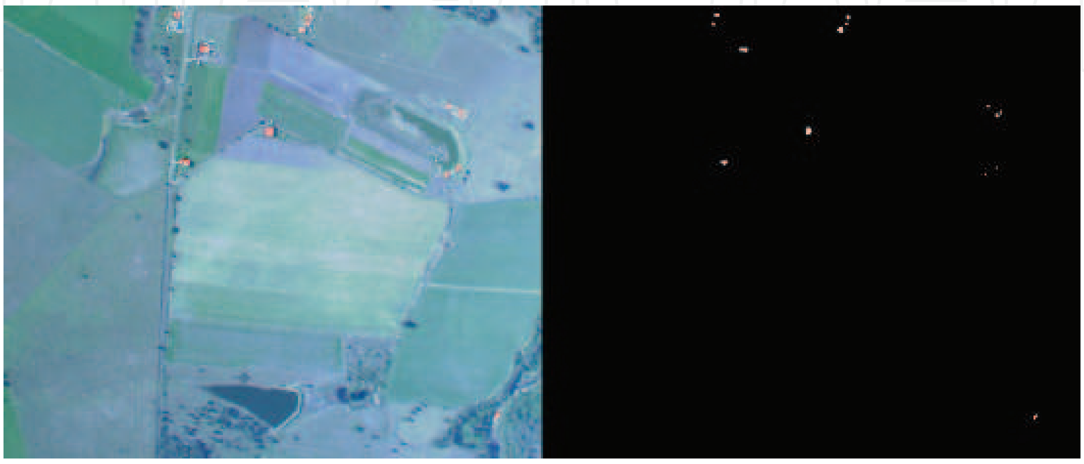
The intensity measure was formulated by dividing the intensity image space into 10 equal partitions and labelling any pixel in the highest valued intensity region a 1 and the other lower intensity valued pixels a value of 0. Any pixel with a value of 1 was considered to be a man-made region and assigned a *very unsafe* value and would not be considered as a potential UAV forced landing site. The pixels with intensity values in the top 10% of the intensity scale were removed from landing site consideration as they were assumed to be man made objects.

The following figure (Figure 5) shows some of the test image dataset with the made-man areas identified by the intensity measure shown overlaid on the original images. The intensity measure images are shown also for clarity.

In summary, the edge detection measure is used for the identification of objects in the image based on the assumption that edges occur between boundaries of objects. This assumption is valid under the condition that there is sufficient contrast between objects or regions in the image and that the spatial resolution is large enough. It follows that as the height above the ground decreases, the better the algorithm is at defining boundaries between objects and detecting smaller objects.

The intensity measure aims to eliminate some of the man made objects in the scene. These usually correspond to white building or roof tops, as these are most likely to reflect the sun the most to the airborne sensor. It is important to note that the algorithm does not perform any intensity stretching, that is, intensities are not scaled. This means that the algorithm remains valid over differing terrain, where a large number of man-made objects can be eliminated in dense urban environments or only a few objects eliminated in rural regions. Note also that the algorithm appears to work well accross the greatly differing altitudes used in our testing. One disadvatage, is that on more overcast days or at earlier or later times of the day, some objects may not be detected.

The two measures were combined by performing a bit OR across each individual pixel in the two measure maps. The resultant output map pixels will have a "1" in every location that had a "1" in either the edge detection measure or the intensity measure map – *unsafe* pixels will have a value of "1" and *safe* pixels will have a value of "0".



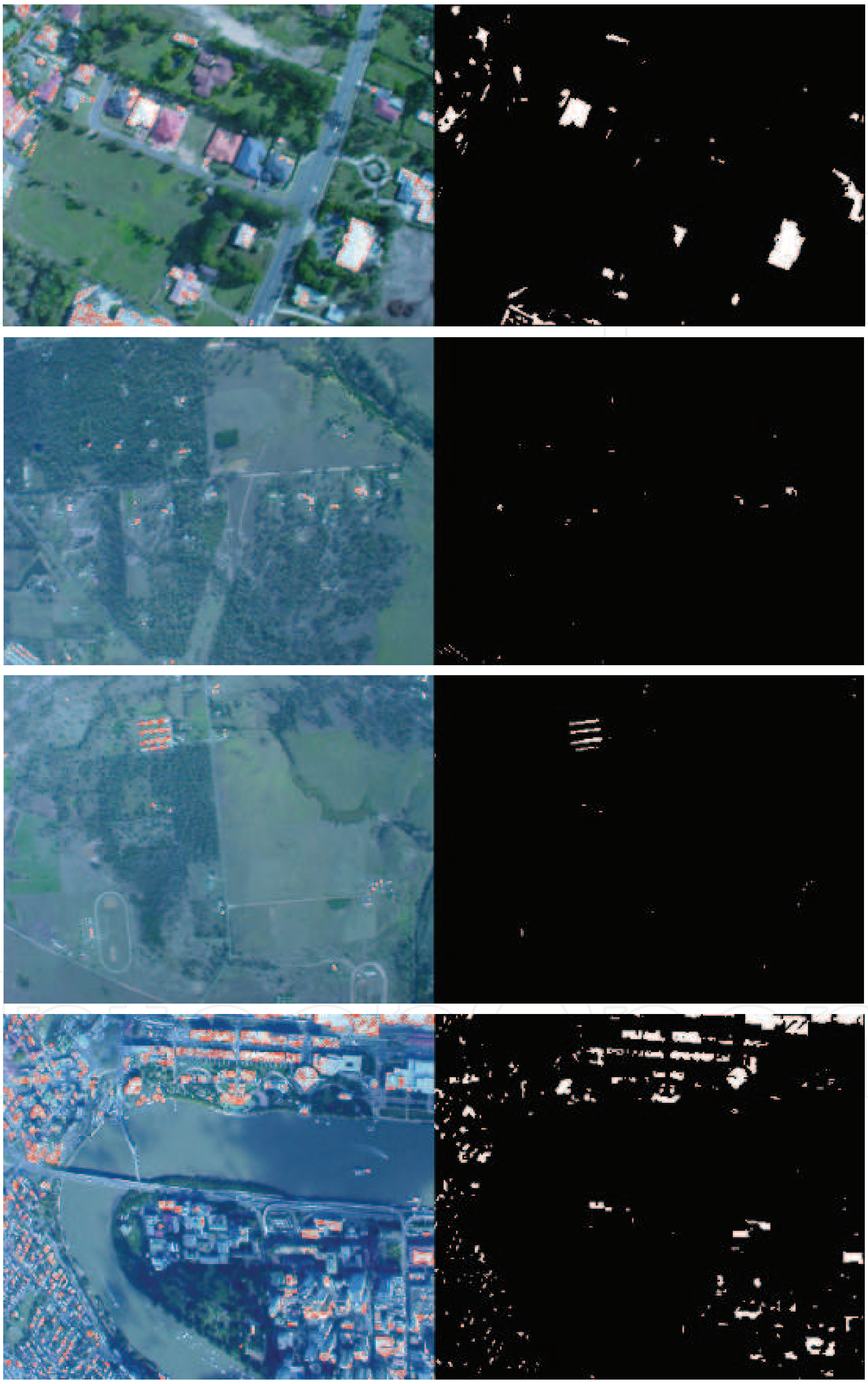


Figure 5. Intensity Measure Result for a Number of Test Images

The Geometric Acceptance phase then takes this map and looks for areas of appropriate size and shape for a UAV forced landing that contain pixels with values only equal to "0".

*Geometric Acceptance Phase*

The geometric acceptance phase locates landing areas of a given size and shape suitable for a UAV forced landing. All areas that have a suitable size and shape will be then labelled as candidate landing sites and will be considered in the next phase of surface type classification.

The algorithm to perform the geometric test in the geometric acceptance phase involves the use of four masks. The masks are rectangular in shape and scalable (size is determined by the pixel resolution calculation). They are also rotated in a number of orientations, catering for approaches to the candidate landing site from different directions. The four masks (labelled A-D) are shown in the figure below.

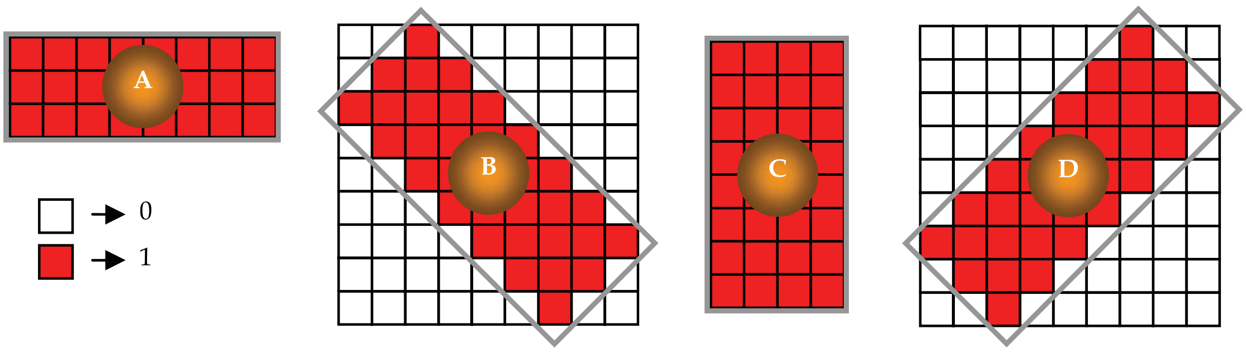


Figure 6. Landing Site Matrix Mask Definitions

Additional mask rotations could have been used, however four masks were chosen, as it was observed that the four masks gave adequate coverage. The example outputs shown in Figure 7 illustrates this point, with the output of the Preliminary Site Selection layer map shown for each test image. As can be seen the masks perform well in extracting every possible area clear of obstacles available. Additionally, the use of only four masks keeps the processing time to a minimum.

The masks are individually moved over the output map from the combination of the edge detection and intensity measure described in the previous section. The image area that the mask passes over is scanned to determine whether or not the area contains entirely *safe* pixels. If all pixels in the area are SAFE, then the area is marked as a candidate landing location.

To perform the scanning check, each mask is represented by a matrix with a "1", indicating that it is part of the mask and a "0" representing that it is not part of the mask (refer Figure 6). For instance, mask B, contains both "0" and "1" elements – the elements that are marked as "1" are members of the mask. For a particular region being tested, if all pixels under the pixels in the mask containing a "1" are *safe* pixels (value of "0") then the pixels in this region on the final Preliminary Site Selection map are set to a *safe*value. This process is repeated across the entire image.

Figure 7 shows some example outputs of the Preliminary Site Selection algorithm. *Safe* areas are shown in white, and the *unsafe* areas in black. Results based on analysis of hundreds of images are summarised at the end of this chapter.

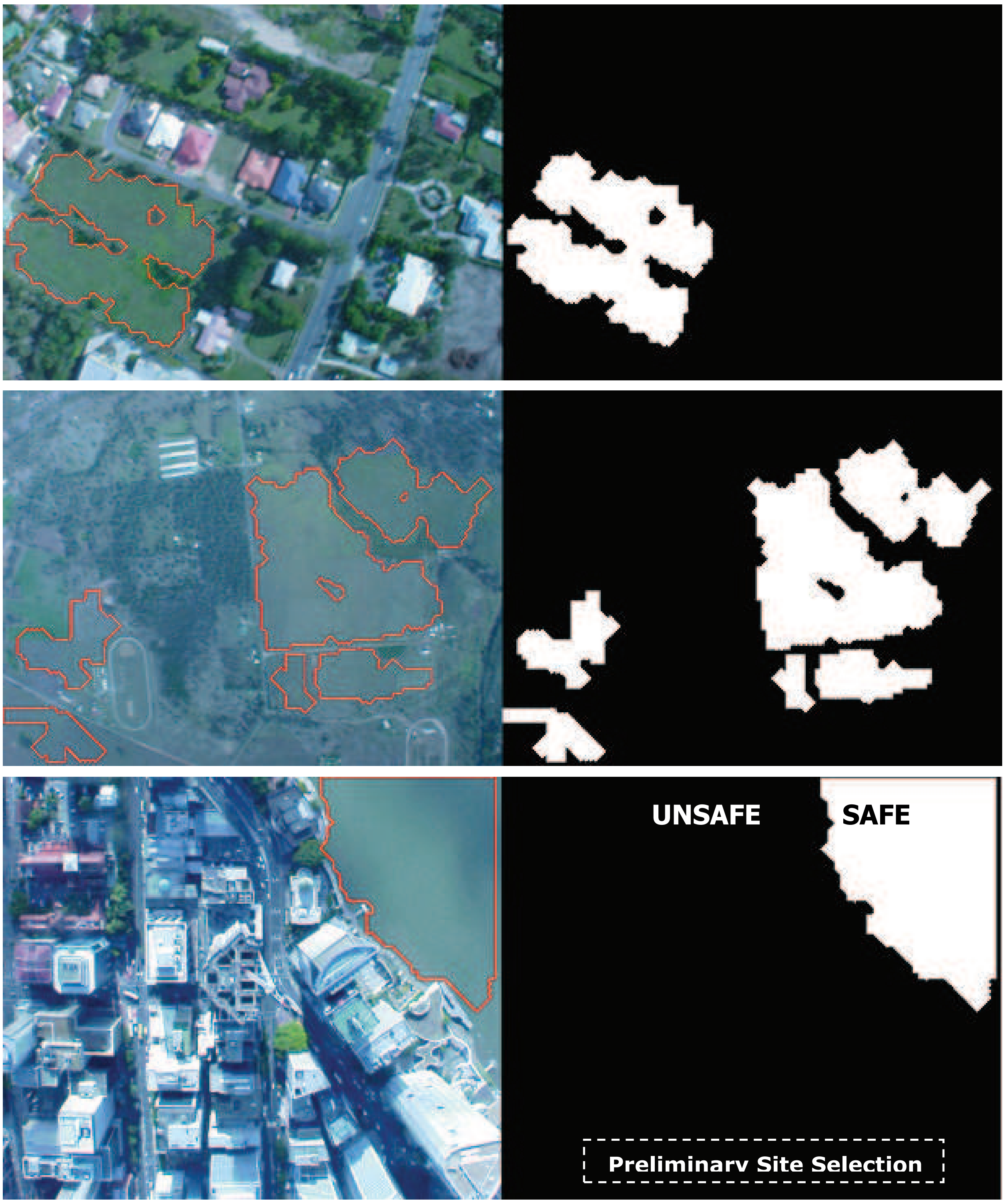


Figure 7. Preliminary Site Selection Output Map Examples

### 3.1.2 Surface Type Classification

The machine vision techniques presented in Section 3.1.1, locate candidate landing sites based on their size, shape and slope. The surface type classification layer is then concerned with labelling each candidate landing site with the surface type so the best landing site can be chosen from those available. This problem falls into the well studied areas of texture classification, pattern classification and the field of automated image indexing.

Our novel controbutions in this area fall in the areas of developing generic semi-automated approaches for developing classification systems. In particular, we automate the process of splitting our dataset into subclasses from generic classes (for example, *grass*, *trees*, *water*) to improve classification accuracy of the system. This approach is good as a human operator can tell easily the difference between class samples such as *grass* or *water*, etc, given the contextual information of the whole seen, however can find it difficult to decide on a number of suitable sub-classes and in turn which samples belong to a particular subclass. Examples of subclasses that a human may find it hard to distinguish between for say *grass*, could be *gree grass* and *brown grass*.

Another important contribution lies in finding a suitable selection of input features for the classification system. We present a method for finding optimal features (the input feature optimisation algorithm – IFO), where the feature space is automatically studied for nearly 90 different features from the literature for each subclass, and the algorithm determines which features are good at distinguishing between subclasses.

Using these algorithms with a back-propogation neural network classifier yielded improvements over our original methods tested by around 10%. Full details of the approach can be found in (D.L. Fitzgerald, 2007) and detailed results to date are given at the end of this chapter.

### 3.1.3 Coarse Slope Estimation

This section addresses the gathering of a coarse slop estimate for the immediate area of interest for the UAV forced landing, and does fall outside the machine vision area, but is included in this section for completeness.

The incorporation of slope into the selection of a suitable forced landing area is a fundamental component of the forced landing process. Slope is one of the key indicators that a human pilot uses to select a forced landing site. A human pilot uses depth of perception and other visual cues to determine the slope of the terrain below. Inferring slope from vision based methods has been well studied in the literature however a robust solution to the problem is still yet to be found. Some of the problems include occlusions of features from shadows and other objects, effects of differing lighting conditions and also problems with reconstruction of 3D from higher altitudes. This particular research is tackling the problem of candidate landing site selection from relatively high altitudes, thus methods based solely on machine vision are seen to not offer the robustness in the design for the aforementioned reasons.

Coarse slope estimation in our sense, refers to finding the slope of the ground in the surrounding area. It is *coarse*, as we are talking about freely available height DEM (digital elevation map) data at a resolution of 3 arc-seconds (latitude and longitude). This corresponds to height elevation readings at approximately 90m spacings. The data that was sourced to demonstrate this technique was from the National Geospatial-Intelligence Agency (NGA) and NASA Shuttle Radar Topography Mission (SRTM). The heights are referenced to the WGS84 geoid which is useful as GPS altitude is referenced to this same geoid. The idea of estimating the slope is to discount areas in the surrounding region.

The augmentation of the vision payload with the GPS and DEM data will provide a number of useful pieces of information that will assist in the selection of a candidate landing site for a UAV forced landing. The following is a summary of the information that has been identified as useful for this problem:

- Calculating pixel resolution;
- Calculating the coarse slope map for each image frame; and
- Defining a slope map for the global operational area.

The knowledge of the pixel resolution in the image is important, because this information is used to define and construct the landing site dimensions for finding sites large enough to land in. If a particular UAV requires 20x100m to land in for example, then this must be converted into the appropriate pixel dimensions for the Preliminary Site Selection algorithm as described above.

The information from the coarse slope map will contain measures of slope for the pixels in each image frame and will be able to be fused directly to the other layers of information to provide a final output map of candidate landing sites.

Finally, a slope map for the global operational area will allow higher order navigation processes to guide the UAV toward areas within the glide range where the slope of the terrain is suitable. These areas may be outside the field of view of the vision sensor and so this slope information plays a vital role in assisting mission planning type decisions for navigation. This information becomes increasingly important if no suitable candidate landing sites exist below the UAV's current position, with the knowledge that the UAV only has a finite time before it reaches the ground.

The full details of this research are presented in (D.L. Fitzgerald, 2007), however we will briefly run over the basical concepts.

*Calculating Pixel Resolution*

Pixel resolution is based on the field of view of the camera, the number of pixels in the camera's image plane and the height above the ground being viewed. It follows that for this problem some averaging of the terrain heights within the field of view of the camera is required to generate the best estimate of pixel resolution for the complete image frame.

Note the following figure.

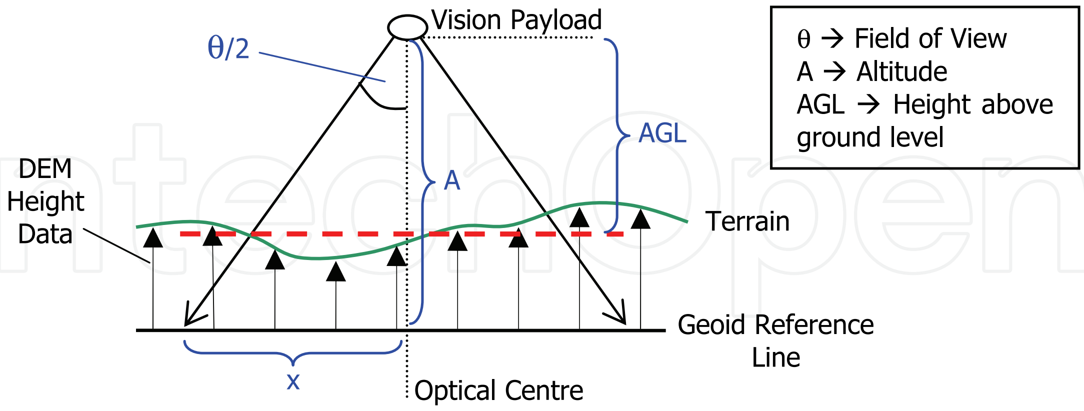


Figure 8. Finding the Average Height Above Ground

The first step is to determine which of the DEM data points are going to be included in the calculation for the average height above ground level (AGL). The points chosen will be those that fall within the view of the camera, extended to the geoid reference line. On the above figure, this would include the DEM Height Data arrows within the x range indicated on both sides of the optical centre of the vision payload.

The pixel resolution can then be determined from this estimate of the height above ground. This calculation uses the distance spanned across the image (based on the AGL) divided by the number of pixels across the image. This final calculation used to determine the pixel resolution is shown in (1).

$$\Delta Px = \frac{(A - H_{av})\tan\left(\frac{\theta}{2}\right)}{\frac{NumPixels}{2}} = \frac{2(A - H_{av})\tan\left(\frac{\theta}{2}\right)}{NumPixels} \tag{1}$$

*Calculating the coarse slope map for each image frame*

The purpose of the coarse slope map is to provide measures of slope for the pixels in each the image frame that can be fused directly with the other layers of information. The problem is to solve where each DEM data point is projected to on the image plane. When laid out appropriately as in Figure 7, it can be seen that the projection problem involves the solution to 2 similar triangles.

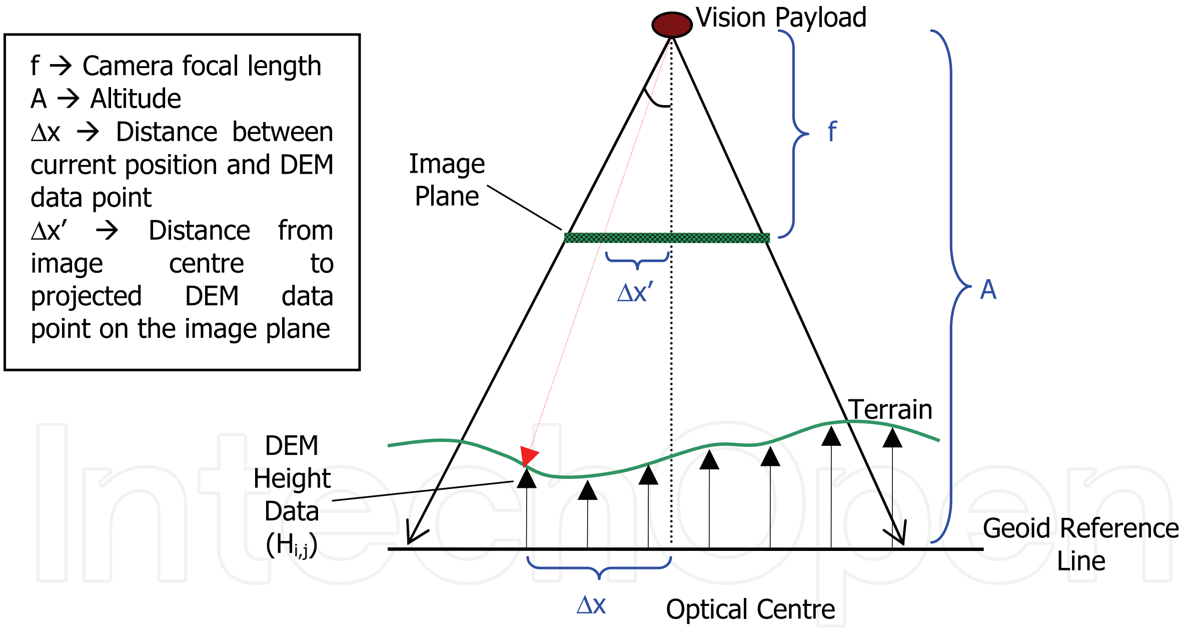

f → Camera focal length
A → Altitude
Δx → Distance between current position and DEM data point
Δx′ → Distance from image centre to projected DEM data point on the image plane
Vision Payload
f
Image Plane
Δx′
A
Terrain
DEM Height Data (H_{i,j})
Geoid Reference Line
Δx
Optical Centre

Figure 9. Solving the Terrain Projection Problem

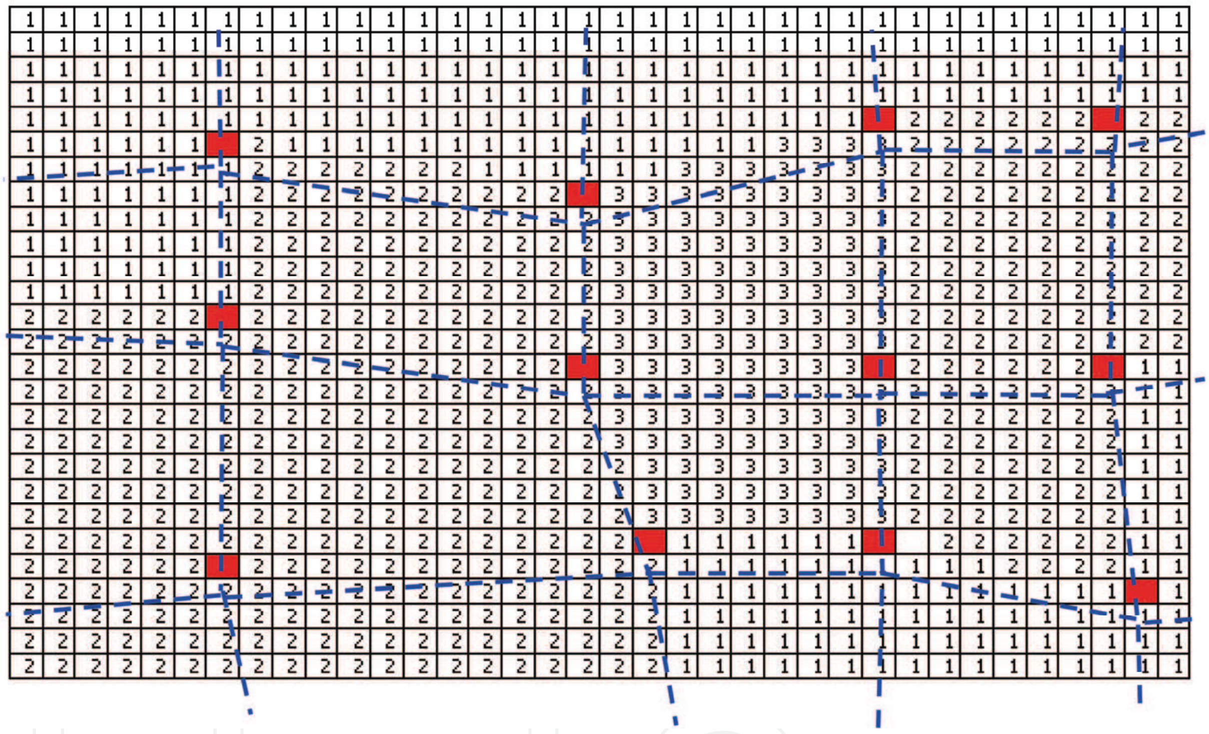
The position of the DEM data point as projected onto the image plane is solved in the x and y directions by the following equations:

$$\Delta x' = f.\frac{\Delta x}{A - H_{i,j}} \tag{2}$$

$$\Delta y' = f \cdot \frac{\Delta y}{A - H_{i,j}} \tag{3}$$

The Δx and Δy positions are added or subtracted from the position of the centre of the image (optical centre) to obtain the location for each height data point. The distribution of these points is totally dependent on the contour of the terrain below. DEM data points are continued to be projected onto the image plane and beyond the boundary, such that all pixels in the image are able to be assigned a slope value as discussed below.

The next step is to determine the slope measures that lie between the data points. This involves looking at the sets of 4 DEM data points and determining the maximum slope between them. Once this is done all pixels that lie between these 4 points are labelled with the appropriate slope measure. An example of this is shown in the following figure.



| | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 🟥 | 2 | 2 | 2 | 2 | 2 | 2 | 🟥 | 2 | 2 |
| 1 | 1 | 1 | 1 | 1 | 1 | 🟥 | 2 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 3 | 3 | 3 | 3 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 |
| 1 | 1 | 1 | 1 | 1 | 1 | 1 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 1 | 1 | 1 | 1 | 1 | 1 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 |
| 1 | 1 | 1 | 1 | 1 | 1 | 1 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 🟥 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 |
| 1 | 1 | 1 | 1 | 1 | 1 | 1 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 |
| 1 | 1 | 1 | 1 | 1 | 1 | 1 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 |
| 1 | 1 | 1 | 1 | 1 | 1 | 1 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 |
| 1 | 1 | 1 | 1 | 1 | 1 | 1 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 |
| 2 | 2 | 2 | 2 | 2 | 2 | 🟥 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 |
| 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 |
| 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 🟥 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 🟥 | 2 | 2 | 2 | 2 | 2 | 2 | 🟥 | 1 | 1 |
| 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 1 | 1 |
| 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 1 | 1 |
| 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 1 | 1 |
| 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 1 | 1 |
| 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 1 | 1 |
| 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 1 | 1 |
| 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 🟥 | 1 | 1 | 1 | 1 | 1 | 1 | 🟥 |  | 2 | 2 | 2 | 2 | 2 | 2 | 1 | 1 |
| 2 | 2 | 2 | 2 | 2 | 2 | 🟥 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 1 | 1 | 1 | 1 | 1 | 1 |  | 1 | 1 | 1 | 2 | 2 | 2 | 2 | 1 | 1 |
| 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 🟥 | 1 |
| 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

Figure 10. Example Image Slope Map Output

The highlighted pixels/squares labelled in the figure above indicate a DEM data point location. Pixels between these data points are assigned a value depending on the maximum slope in the region. A "1" indicates a Flat region, "2" indicates a Sloped region and a "3" indicates a "Steep" region. Dotted lines have been overlaid to highlight the connections between the DEM data points and the slope boundary locations.

*Defining a slope map for the global operational area*

A slope map for the global operational area will allow higher order navigation processes to guide the UAV toward areas within the glide range where the slope of the terrain is suitable. These areas may be outside the field of view of the vision sensor and so this slope information plays a vital role in assisting mission planning type decisions for navigation. This information becomes increasingly important if no suitable candidate landing sites exist below the UAV's current position, with the knowledge that the UAV only has a finite time before it reaches the ground.

This is an extension of the previous theory presented in this chapter. The only additional input required is the glide performance of the aircraft, and once this is defined, the maximum horizontal distance from the current position can be solved.
The image below shows the height map above the S 260 30″ 0′; E 1520 30″ 0′ area at an altitude of 400m using a function that we have written.

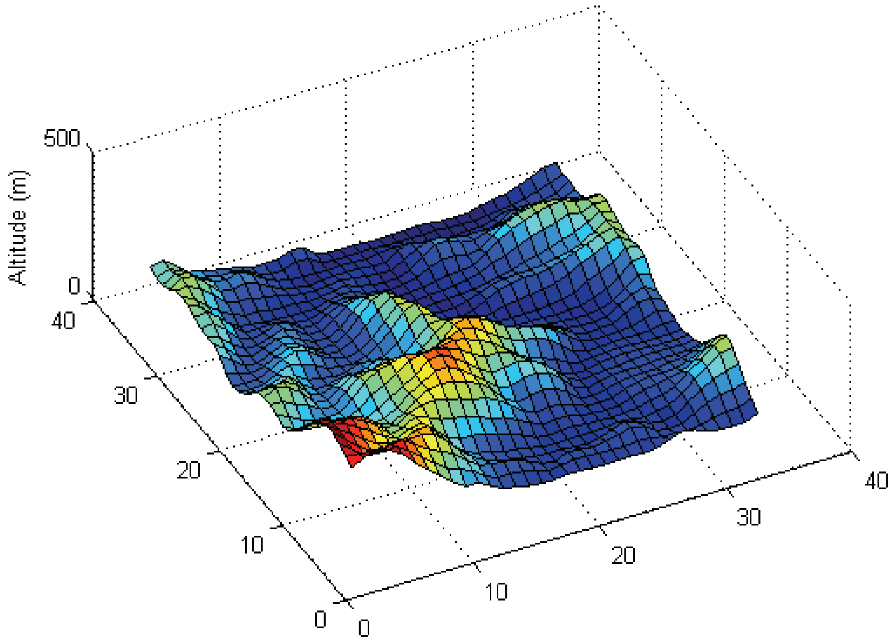


Figure 11. Example Height Map at 400m altitude

The corresponding slope map was then constructed for this entire area and is shown in Figure 11 & Figure 12. Note how the sloped and flat areas are related in the two figures.

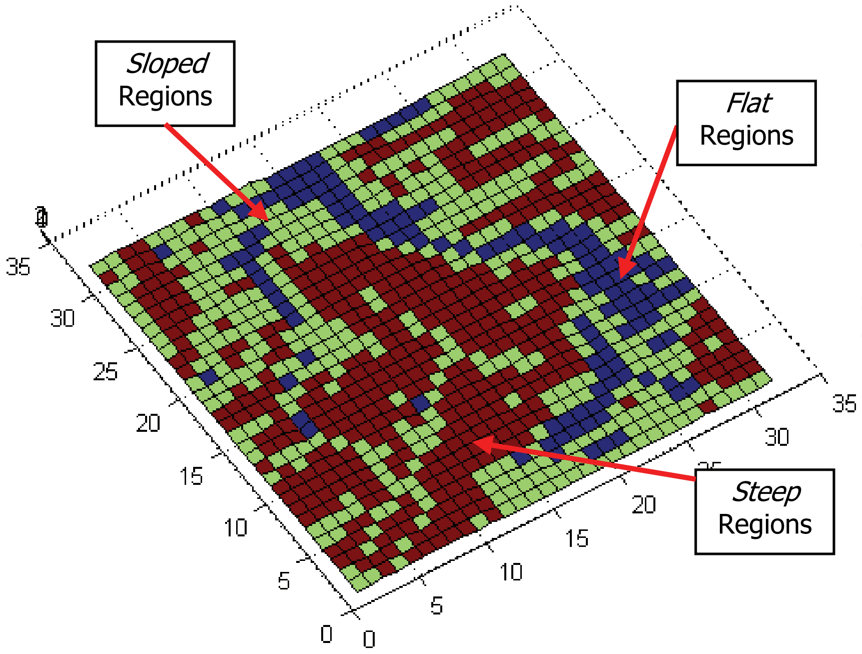


Figure 12. Example Slope Map at 400m Altitude

As discussed, the glide performance of the aircraft will dictate how far the aircraft can travel before reaching the ground, and this can then be used to define the slope map for the operational area.

### 3.2 Results

A series of flight trials were performed using a Cessna 172 aircraft with a 100% success rate for locating large open landing areas. 92% of these large open landing areas were considered to be completely free of obstacles, with only 8% having small obstacles such as trees. These obstacles were missed by the algorithm due to the resolution of the camera vs the current height above ground. As the UAV descends however, these obstacles would be detected and the descent planner could take the appropriate action.

The surfaces of these large open areas were also classified to accuracies of over 93% - for example grass, water, etc. This classification information will be used by the descent planner to select the most suitable landing site from the ones available.

The recommendation is that additional information be used to select the most appropriate landing area to compliment the research. Based on this recommendation, it is proposed that additional maps will be produced by the landing site subsystem to highlight keep-out areas such as roads and buildings.

**Summary**

This section has presented the techniques we have developed to find potential landing site areas for a fixed wing UAV. The testing to date has involved the use of manned aircraft to evaluate the Landing Site Selection algorithm and surface classification techniques and algorithms we have developed. The next phase is to implement these algorithms in real time and evaluate the performance on a small UAV in a forced landing scenario.

## 4. Decision making: Choosing the right landing site

One of the most important tasks in the initial stages of a forced landing is to decide on a feasible landing site, and then how to best approach this landing site. These two aspects are closely related to the multicriteria decision analysis and the trajectory planning and tracking component of the overall approach, respectively. This section will shed light on the main concepts behind the challenging decision-making process, which in reality is continuosly validated and updated throughout much of the descent should new information yield a more appropriate landing site.

### 4.1 Multiple Criteria

According to the Australian Civil Aviation Safety Authority’s latest Visual Flight Rules flight guide (CASA 2001), there are seven criteria to selecting the optimum site for a manned aircraft forced landing. These include:

- Wind
- Surrounding
- Size and Shape
- Surface and Slope
- S(c)ivilisation

When applied in the context of UAVs, many of these factors still hold their significance, and a number of other variables also come into consideration which are not explicitly stated for piloted aircraft. These include the aircraft dynamics, the uncertainty of sensor data and the method of estimating wind.

Also to be considered is the geometrical relationship between the various candidate sites. As the aircraft descends, the number of available landing sites will rapidly decrease. Thus, it is generally better to glide towards several possible sites in close proximity than to one that is isolated, as this keeps multiple landing site options open for as long as possible. This is important so as to have several alternatives if obstacles are detected on the candidate landing sites at lower altitudes.

The number of structures and the population density that lies in the descent path to each site must also be accounted for if applicable, as it would be safer to fly over empty terrain than a populated area, in case further mishaps occur. These points, along with other factors which remain to be identified, will be evaluated to reach an optimal, verifiable decision on which candidate landing site the aircraft should aim for.

Further investigations will also be conducted in order to identify any other elements that affect this decision process, possibly including surveys and simulations involving experienced pilots and/or UAV controllers.

### 4.2 Multiple Objectives

The complexity of the forced landing decision process due to multiple criteria is further increased by multiple objectives that must be met. In many cases, these objectives may be conflicting, and thus compromises must be made such that the most critical objective/s could be achieved.

According to the Civil UAV Capability Assessment (Cox, Nagy et al. 2004), in the event of an emergency landing the UAV needs to be able to respond according to the following objectives and in the following order:

1. Minimize expectation of human casualty;
2. Minimize external property damage;
3. Maximize the chance of aircraft survival; and
4. Maximize the chance of payload survival.

In many scenarios, the best landing site for meeting Objectives 3 and 4 may compromise the more important objectives (1 and/or 2), or vice versa. This complex trade-off between the risks and uncertainties involved with each possible choice is but one example of a difficult problem that the multi-criteria decision-making system must face.

### 4.3 Decision Making

The Decision Making module will initially have predeveloped contingency plans from map data to give fast, reflex responses to emergencies. These contingency plans will guide the aircraft towards known landing sites initially, or large flat areas identified from slope map data. The Guidance and Navigation module (discussed in the next section) will constantly make estimates of the wind speed and direction, which will be taken as input for decision making. The aircraft dynamics will also be known and necessary restraints applied when judging the feasibility of a decision. As the aircraft descends, the vision-based Landing Site Selection module will continously analyse the terrain that the aircraft is flying over. Possible landing sites, buildings, and roads will be identified, including the associated uncertainties of objects in each map. With this information the Decision Making module will be able to continuously validate and update its decision in real-time.

It is expected that uncertainties will reduce as the aircraft descends, however the options available will also reduce. It may be very likely that an initially selected landing site will eventually be deemed unsuitable by the Landing Site Selection subsystem, and an alternative must be sought after. It is the responsibility of the Decision Making subsystem to be prepared for such situations by maximizing the number of alternative choices available.

The research in this area is focussing on the development of a multi-agent based architecture, where multiple events require layered decision schemes. Different software agents that handle different events during the landing process will be in constant interaction and communication throughout the descent in order to handle all the different events.

From the literature review, it was concluded that there are essentially two broad classes of multi-criteria decision analysis methods; one follows the outranking philosophy and builds a set of outranking relations between each pair of alternatives, then aggregate that according to some suitable technique. The other essentially involves determining utility/value functions for each criterion, and determining the 'utility' of each alternative based on each criterion, then aggregating those with a suitable technique to find the overall utility of the alternative.

Many of the existing techniques are not designed for 'decision making'; rather they are intended as 'decision aid' methods, and hence some only generate additional information for the human decision maker to make the final decision with. Decision making is in many ways a subjective matter, as discussed earlier, in most cases there is no 'best' decision, and it is subject on the preference information given by human decision makers. Due to the nature of the forced landing situation, where decisions made could potentially lead to damage to property or even harm life, it is critical then that the decision making system to be developed must be based on justifiable and generally accepted preference data. This means that the technique chosen should require preference data that is clear and understandable by people who don't understand the mathematics of method, and also that the technique should be as transparent as possible for purposes of accountability.

Additional requirements used to evaluate the various techniques include the ability to handle uncertainty in terms of input data, and the assumptions made regarding the decision problem. A number of the discussed techniques are currently under trial, such as PROMETHEE [Brans, 2005] and MAUT [Dyer, 2005]. Promethee is an outranking method that requires relatively simple preference data in terms of criterion weights and preference functions. Maut which is based in Expected Utility Theorem makes the assumption of independence, which essentially means that only the probability distribution of risks of individual criterion are considered, and they don't affect each other. This may be unrealistic for the forced landing scenario, yet it can be addressed by using fuzzy Choquet Integrals, which addresses synergy and redundancy between criteria.

The technique of most interest does not readily fit in to either of the main families of multicriteria decision analysis techniques, and that is the decision rules approach, and the one of specific interest is dominance-based rough set approach (DRSA). This method takes samples of decisions made by human experts, and analyses them to determine the minimum set of decision rules expressed in the form of "if…, then…" statements. These statements are then used to evaluate the alternatives in the multi-criteria decision problem, and aggregated with an appropriate aggregation technique such as the Fuzzy Net Flow Score. There is the capacity to deal with inconsistent preference information from the human decision makers by using the rough sets, and fuzzy sets can be implemented to address uncertainty in the input data. This method is the most transparent and understandable of all of those investigated so far, and is being treated as the most promising technique for use in this research.

## 5. Trajectory Planning and Tracking: Commanding the platform to land in the right place

The development of a UAV platform capable of precision flight, addressing safety and reliability as main concerns, is the logical progression for future UAVs in civilian airspace. Achieving this realization will not be limited to designing advanced control laws and/or flight control systems, since these UAVs will be mainly used to support reconnaissance and surveillance roles. For these applications, computer vision can offer its potential, providing a natural sensing modality for feature detection, tracking and visual guidance of UAVs.

An important part of the fixed-wing aircraft forced landing problem is how to navigate to land on a chosen site in unknown terrain, while taking into account the operational flight envelope of the UAV and dynamic environmental factors such as crosswinds and gusts, small flying objects and other obstacles in the UAV glide path. Static obstacles such as buildings, telegraph/light poles and trees on the perimeter of the chosen landing site will also be considered as they may interfere with the approach glide path of the UAV.

### 5.1 Vision-based Navigation in the Literature

In order to command the aircraft to the desired landing site, visual information plays a crucial role in the control of the platform. Using the visual information to control the displacement of an end effector is referred to in the literature as *visual servoing* (Hutchinson, Hager et al. 1996). It is envisaged that the location of the candidate landing sites in the image should be used to command the aircraft while is descending.

Previously (Mejias, Roberts et al. 2006) has demonstrated an approach to command the displacement of a hovering vehicle using an Air Vehicle Simulator, AVS (Usher, Winstanley et al. 2005). This task required the development of suitable path planning and control approaches to visually manoeuvre the aircraft during an emergency landing. In this approach the vehicle had to navigate through a scaled operating environment equipped with power lines and artificial obstacles on the ground and find a safe landing area.

### 5.2 Preliminary Results in Dynamic Path Planning using a Fixed-Wing UAV

Currently, work is underway to develop robust path/trajectory planning and tracking algorithms, and initial simulations using the MATLAB Simulink programming environment have provided valuable feedback on the designs trialled. In these simulations, an AeroSim model of an Aerosonde UAV was modified and expanded to include blocks for flight controls, path planning, GPS waypoint navigation, wind generation, wind correction and an interface to FlightGear. By running MATLAB and FlightGear concurrently, the user is able to visualize the UAV flying in a manner as dictated by the Simulink model.

At present, the primary focus of this simulation is to evaluate the dynamic path planning capability for a UAV performing a forced landing in changing wind conditions. This simulation is intended to serve as a tool in the design and testing of a visual servoing and path planning system for automating a fixed-wing UAV forced landing. It will be further enhanced to model complex, uncooperative environments with hazards such as buildings, trees, light poles and undulating terrain, as well as machine vision for use in the feedback control loop.

### 5.2.1. Wind Compensation

In the current forced landing simulation, the initial wind velocities are given by uniformly distributed random numbers that are updated every sixty seconds. These numbers generate the initial $W_{North}$, $W_{East}$ and $W_{Down}$ components, which are then multiplied by a continuous square wave giving the profile shown in Figure 13. The values of $W_N$, $W_E$ and $W_D$ were chosen based on the wind rose generated for Brisbane, Australia, and combined to give a maximum wind velocity of 60 kts, which can arise from any direction. A wind rose is a diagram that summarises the occurrence of winds at a location, showing their strength, direction and frequency. The wind rose used in the simulation represented wind measurements taken at 9 a.m. from 1950 to 2000, and are published by the Australian Government Bureau of Meteorology.

Note that gusts have not been modelled in the simulation, instead, the input wind is assumed to blow with a constant magnitude and direction for sixty seconds, before changing magnitude and direction for the next sixty seconds. Whilst this does not necessarily represent the wind conditions found in an actual descent, it does present a challenging wind shift scenario for the simulations to date. Future simulations will include wind gusts.

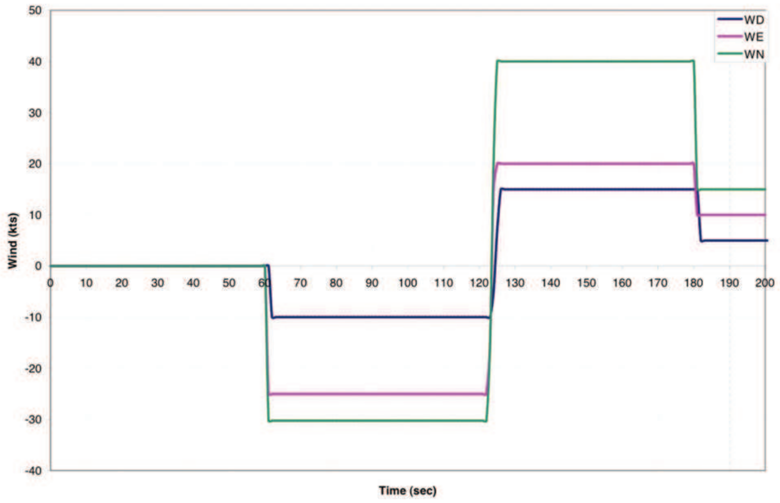


Figure 13. Wind components ($W_N$: Green, $W_E$: Pink, $W_D$: Blue). These components are used to compute the resultant wind vector incident on the UAV

Correction for wind is performed using the principles of vector algebra to compute the wind correction angle, which is compared with the current aircraft heading and passed as input to the UAV flight planning subsystem. From Figure 14, suppose that waypoint B is 600m (0.32 nmi) north-east (045˚ true) of waypoint A and the UAV glides from A to B, maintaining a heading of 045˚ true and a constant True Airspeed (TAS) of 37kts. A wind velocity of 340˚/9.7kts coming from the south-east will cause the UAV to drift to the left.

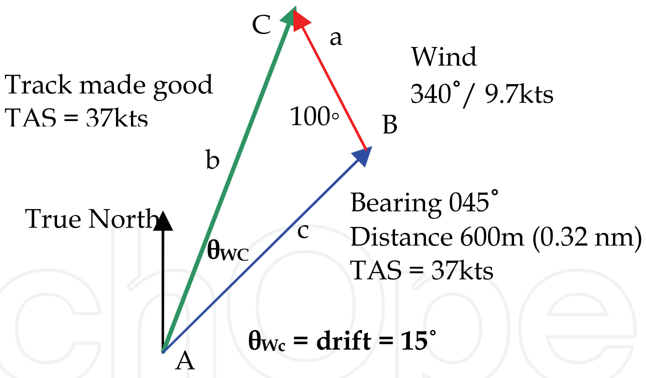


Figure 14. Wind Triangle Calculations

This implies that the wind correction angle supplied to the flight planning subsystem must be 15˚ in the opposite direction, such that the "track made good" will converge on the "required track" to target.

### 5.2.2 Path Planning

In this simulation, the path planning algorithm generated a series of waypoints, which formed a flight path along which the UAV was guided to land at the chosen landing site. The waypoints were extracted from the forced landing circuit pattern as outlined in (CASA 2001). Table 1 gives the coordinates of the idealised waypoints for a right-hand circuit pattern, and Figure 15 shows their relationship to the landing site. Note that a similar pattern for a left-hand circuit pattern can also be generated.

| **Waypoint** | **Longitud (rads)** | **Latitude (rads)** | **Alt (ft)** |
|---|---|---|---|
| High Key | 0.4782 | 2.6725 | **2500** |
| Low Key | 0.4783 | 2.6722 | **1700** |
| End Base | 0.4786 | 2.6721 | **1200** |
| Decision Height | 0.4786 | 2.6723 | **670** |
| Overshoot1 | 0.4787 | 2.6724 | **400** |
| **Aimpoint** | **0.4784** | **2.6725** | **13** |

Table 1. Waypoints – Left-hand Approach Circuit Pattern

Based on the initial position of the UAV, the path planning algorithm then generated a modified table of waypoints which included the aim point, and all or a combination of the other waypoints listed in Table 1. The UAV then flew to these new waypoints using the great-circle navigation method defined in (Kayton and Fried 1997), using a set of Proportional-Integral-Derivative (PID) controllers to control the airspeed and bank angle Figure 15 depicts three possible flight paths generated using the planning algorithm described.Fixed-Wing Simulation Results

To test the performance of the path planning algorithm, a Monte Carlo simulation consisting of 500 automated landings was conducted. The simulations were run with randomised initial aircraft positions, attitudes and wind velocities. In this simulation we observed that the majority of landings had a radial miss distance between 0 and 400m from the aimpoint, which is located one-third along the length of the landing site from the direction of final approach. The value of the miss distances can be attributed to several factors; the relative spacing between the waypoints, how the path planning algorithm selects the waypoints for the UAV to navigate to and the fact that the UAV is constrained to fly with a positive 3 degree pitch attitude. However, from these tests it was observed that 151 landings lay within the site boundaries, corresponding to approximately 32% of the total population. While this figure was not exemplary, it did present a baseline for subsequent refinements to the navigation and path planning algorithms to improve upon.

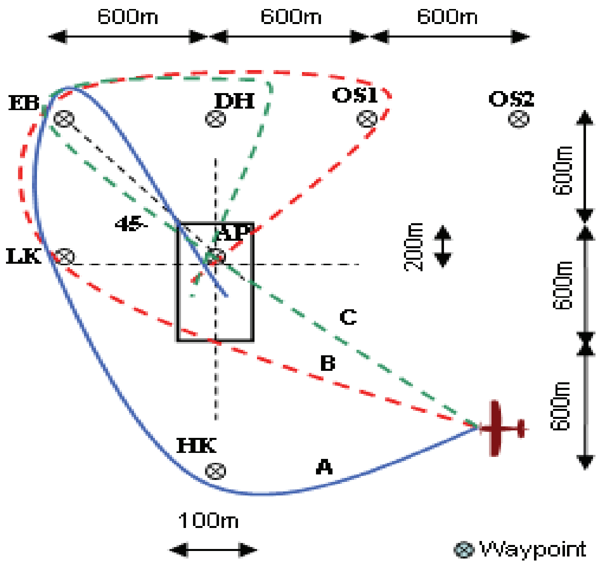


Figure 15. Forced Landing Circuit Patterns. HK=high key, LK=low key, EB=end base, DH=decision height, OS1=overshoot 1, AP=aimpoint.

Figure 16 shows a plan and isometric view of the aircraft trajectory during one simulated landing manoeuvre. The green arrows depict the direction of the changing wind affecting the aircraft during flight. The path described by the red line is the trajectory computed by the path planning algorithm, while the blue line is the actual path that the aircraftdescribes. The designated landing area is illustrated by a thick green line.

### 5.3 Remodelled Path Planning and Tracking Algorithm

To improve upon the algorithm described above, a remodelled path planning, tracking and control strategy has been implemented that does not restrict the robotic aircraft to following trajectories developed for human pilots. In addition, a model of a Boomerang 60 UAV was chosen for evaluation, as this represented the platform to be used in future flight tests and has manoeuvring capabilities similar to the Aerosonde. One other major difference between the remodelled algorithms and that based on the CASA forced landing circuit is that the latter will only attempt to guide the UAV from an initial point of engine failure to a point where the aircraft is aligned with the landing site and trimmed for final descent. Usually, this final approach point is 400-500 ft above ground level and it has been assumed that the Global Positioning System (GPS) signals will be available throughout the descent to this altitude. Below this altitude, GPS signals could be affected by multipath errors, dropouts, jamming and other adverse conditions, hence other sensors, such as machine vision, would need to be employed to enhance the accuracy and integrity of the existing navigation and guidance system. The descent from final approach to touchdown will be the topic of future research.

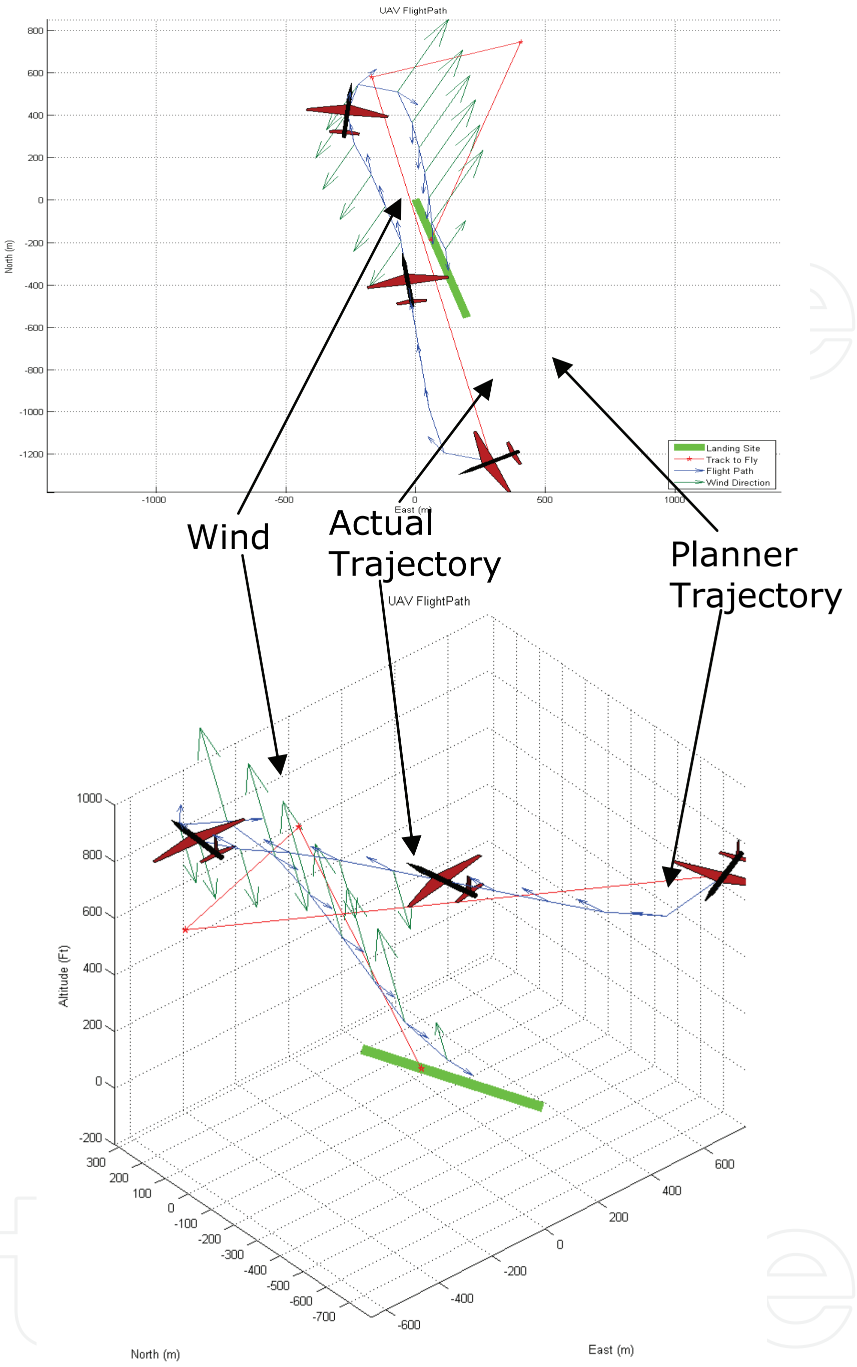


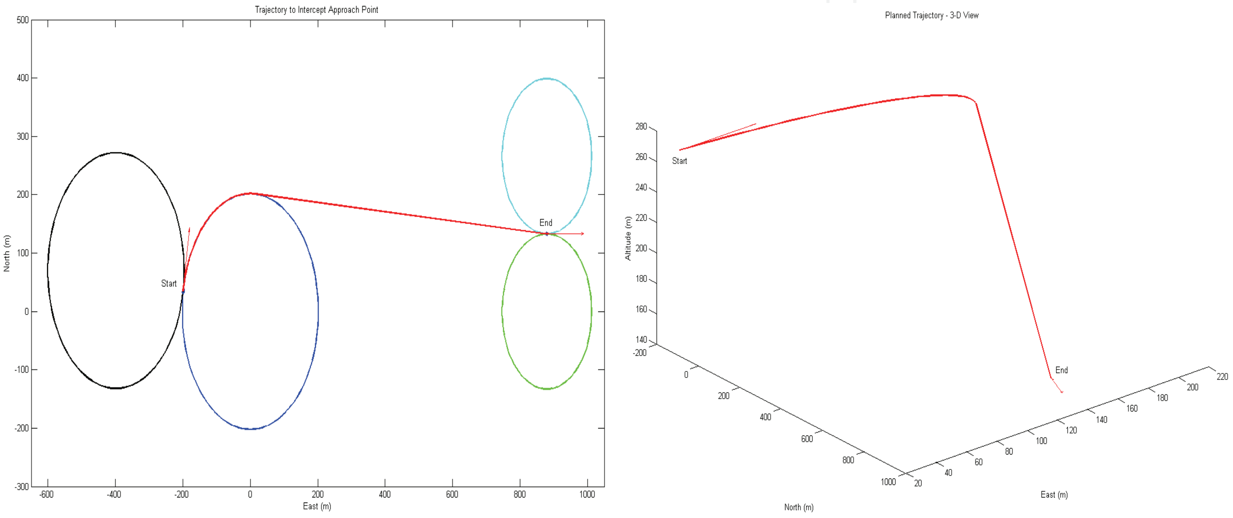
Figure 16. Detailed view of the forced landing simulations under changing wind. Green arrows indicate the direction of the wind

### 5.3.1 Improved Path Planning

The new path planning strategy is based on the concept of Dubins circles [Dubins, 1957] and the work presented in [Ambrosino, 2006]. Given a desired start and end position, the shortest path to the goal can be constructed geometrically in the *xy*-plane by the union of an arc of circumference, a segment and again an arc of circumference. There is no constrain on the radii of the arcs except that they should not exceed the minimum turn radius of the aircraft.

To translate the path into 3-D, the portions of path described by the arcs are then transformed into that traced segments of helix, and the two segments are then joined together by a straight line. This line has an angle of elevation $\gamma_P$ that does not exceed the maximum dive angle, $\gamma_{sc}$ of the aircraft. Should the aircraft be initially higher than the described path allows, the planning algorithm will extend the path such that the aircraft can circle to lose altitude (in reality flying along *n* spirals of a helix) before joining the path at the start of the first helix.

A representative example of the planned trajectory in 2-D is shown in Figure 17a. The desired initial and final positions are indicated by red arrows, and these positions are linked by the shortest (optimal) Dubins path highlighted in red. Note that the radii of the initial and final arcs of circumference are different, but are both greater than the minimum turn radius of the aircraft. The final 3-D flightpath is depicted in red in Figure 17b.



**a)** **b)**

Figure 17. (a) The planned descent trajectory in 2-D. The optimal path is shown in red, joining the desired start and end positions (red arrows). (b) The planned descent trajectory in 3-D

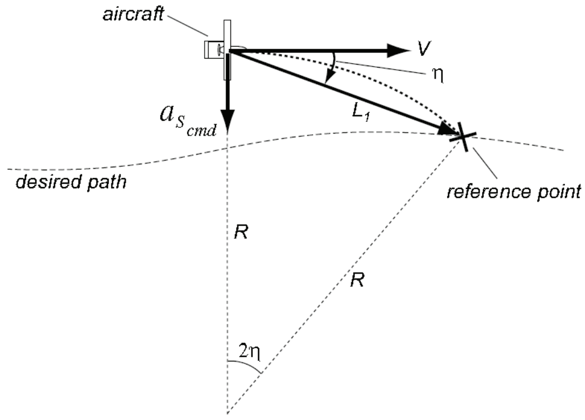


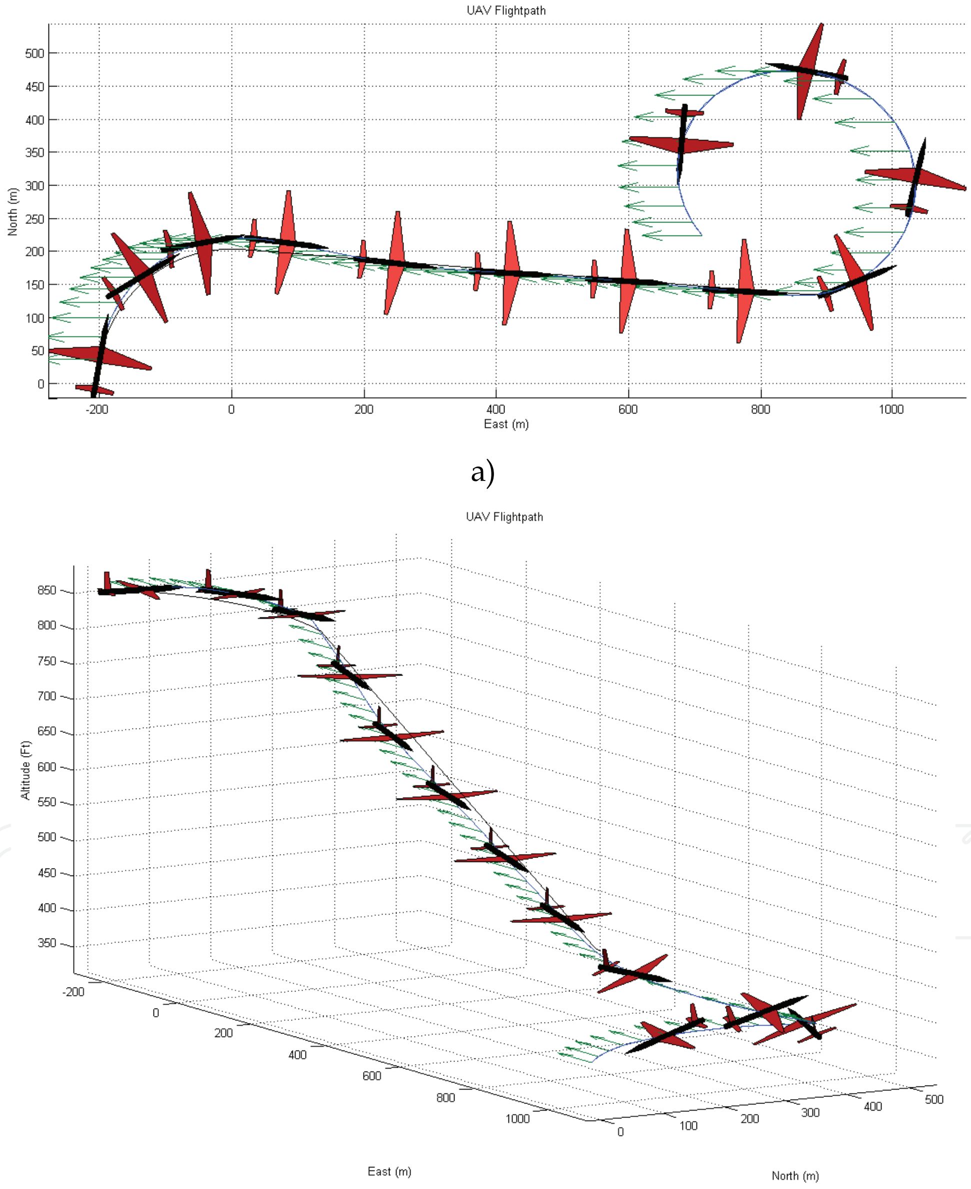
Figure 18. Diagram for guidance logic (Park, 2007)

### 5.3.2 Trajectory Tracking

The new trajectory tracking strategy is based primarily on the work of [Niculescu, 2001] and [Park, 2007]. Essentially, the guidance logic selects a reference point on the desired trajectory, and then generates a lateral acceleration command using the reference point. The lateral acceleration command is determined by

$$a_{s_{cmd}} = 2\frac{V^2}{L_1}\sin\eta \qquad 4$$

Where $V$ is the airspeed of the aircraft, $L_1$ the distance to the reference point, forward of the vehicle, and $\eta$ the angle between the $V$ and $L_1$ vectors. The sign of $\eta$ controls the direction of acceleration, which is equal to the centripetal acceleration required to follow an instantaneous circular segment with radius $R$. The relationships between V, $L_{1,}$ $\eta$ and $R$ are depicted in Figure 18.



a)

b)

Figure 19. (a) 2-D view of the flightpath using the new path planning and trajectory tracking algorithms. Wind is indicated by the green arrows. (b) 3-D view of the same flightpath

To track the path in the longitudinal plane, the algorithm compares the desired path angle, $\gamma_d$ with the current aircraft path, $\gamma_{sc}$ and feeds the error into a set of PID gain schedules. In calculating $\gamma_d$, the algorithm uses the speed polar curve specific to the aircraft to determine the desired airspeed and path angle, $\gamma_w$ to achieve in ambient wind conditions, then compares this value with the original path angle, $\gamma_{p.}$ The speed polar curve is used to determine the glide angle that would produce the minimum loss of altitude in wind. The error between $\gamma_w$ and $\gamma_p$ is then multiplied by a gain such that the aircrafty will still converge on the path to fly. Figures 19a and 19b show an example of the performances of the new trajectory planning and tracking algorithms. Wind is blowing from the east at 5 m/s (10 kts) and is indicated by the green arrows. The solid black line shows the path-to-track and the actual flightpath is depicted by the solid, blue line. The lateral cross-track error at the final approach point (which can be considered as the aimpoint when comparing with the initial algorithms) is approximately 1.1 m, while the longitudinal cross-track error (altitude error) is just over 4 ft. These results demonstrate that the new algorithms can be feasibly implemented on a real UAV for flight testing.

## 6. Conclusions

A number of research programs have been presented in this chapter and an overview of the approaches used in a Forced Landing system for UAVs. This overview aims to present the methodology for the development of a system capable of flight trials for a UAV forced landing.

Research in this area by the group over the past three years has seen the technical risk of an autonomous UAV forced landing system decrease, and the group is now confident that with the existing results and new research objectives, future flight tests will demonstrate this level of capability that is missing in UAVs today. The capability for a UAV to be able to land in an unknown environment with no human input is something that must be solved if UAVs are to fly above populated areas in civillian airspace.

Many complex decision making problems still remain to be investigated. These problems involve multiple conflicting objectives, and it is most often true that no dominant alternative will exist that is better than all other alternatives. Generally it is impossible to maximize several objectives simultaneously, and hence the problem becomes one of value tradeoffs. The tradeoff issue often requires subjective judgement of the decision maker, and there may be no right or wrong answers to these value questions.

It is believed that the approach presented will allow the progression of this novel UAV forced landing system from the development and simulation stages through to a prototype system that can demonstrate this important capability for UAVs to the research community.

## 7. Acknowledgement

The authors would like to thank Richard Glassock for piloting the test aircraft, as well as Dmytriy Stepchenkov and Lachlan Mutch for providing engineering support in the field.

## 8. References

Ambrosino, G., Ariola, M., Ciniglio, U., Corraro, F., Pironti, A., & Virgilio, M. (Eds.). (2006) *45th IEEE Conference on Decision & Control*, . San Diego, U.S.A

Azencott, R., Durbin, F., & Paumard, J. (1996). Robust recognition of buildings in compressed large aerial scenes. Paper presented at the *Image Processing*, 1996. Proceedings., International Conference on.

Brans, J. P., & Mareschal, B. (2005). Promethee Methods (Vol. 78): Springer New York

Canny, J. F. A computational approach to edge detection. *IEEE Trans Pattern Analysis and Machine Intelligence, 8*(6), 679-698.

CASA. (2001). *VFR Flight Guide*: Civil Aviation Safety Authority Australia (CASA) - Aviation Safety Promotion Division.

Chen, K., & Blong, R. (2002). Extracting building features from high resolution aerial imagery for natural hazards risk assessment. Paper presented at the G*eoscience and Remote Sensing Symposium*, 2002. IGARSS '02. 2002 IEEE International.

Cox, T. H., Nagy, C. J., Skoog, M. A., & Somers, I. A. (2004). *Civil UAV Capability Assessment*.

Dyer, J. S. (2005). MAUT. Multiattribute Utility Theory (Vol. 78): Springer New York.

Dubins, L. E. (1957). On Curves of Minimal Length with a Constraint on Average Curvature with prescribed Initial and Terminal Positions and Tangents. *American Journal of Mathematics*, 79, 471-477.

Fitzgerald, D. L. (2007). *Landing Site Selection for UAV Forced Landings Using Machine Vision.* Phd Thesis, Australian Research Centre for Aerospace Automation (ARCAA), Queensland University of Technology.

Fitzgerald, D. L., Walker, R. A., & Campbell, D. (2005). A Vision Based Forced Landing Site Selection System for an Autonomous UAV. Paper presented at the 2005 *IEEE International Conference on Intelligent Sensors*, Sensor Networks and Information Processing (ISSNIP).

Howard, A., & Seraji, H. (2001). Intelligent Terrain Analysis and Information Fusion for Safe Spacecraft Landing. *NASA - Jet Propulsion Laboratory, Technical Document*.

Howard, A., & Seraji, H. (2004). Multi-sensor terrain classification for safe spacecraft landing. *IEEE Transactions on Aerospace and Electronic Systems, 40(4)*, 1122-1131.

Hutchinson, S., Hager, G. D., & Corke, P. I. (1996). A tutorial on visual servo control. *Robotics and Automation, IEEE Transactions on , Volume: 12, Issue: 5 , Oct.*, 651 -670.

Kayton, M., & Fried, W. R. (1997). *Avionics Navigation Systems* (2 ed.): John Wiley & Sons, Inc., New York.

Mejias, L., Roberts, J., Corke, K. U., & Campoy, P. (2006). Two Seconds to TouchDown. Vision-Based Controlled Forced Landing. Paper presented at the *IEEE/RSJ International Conference on Intelligent Robots and Systems*, Beijing, China.

Niculescu, M. (2001). Lateral Track Control Law for Aerosonde UAV. Paper presented at the *39th AIAA Aerospace Sciences Meeting and Exhibi*.

Park, S. J. D., & How, J. P. (2007). Performance and Lyapunov Stability of a Nonlinear Path-Following Guidance Method. *Journal of Guidance, Control, and Dynamics*,, 30, 1718-1728..

Usher, K., Winstanley, G., Corke, P., Stauffacher, D., & Carnie, R. (2005). Air vehicle simulator: an application for a cable array robot. Paper presented at the *IEEE International Conference on Robotics and Automation*, Barcelona, Spain