Runway Detection in High Resolution Remote Sensing Data

Philip T.G. Jackson[1], Carl J. Nelson[1], Jens Schiefele[2], and Boguslaw Obara[1] *

[1] School of Engineering and Computer Sciences, Durham University, Durham, UK,
[2] Jeppesen GmbH, Neu-Isenburg, Germany

Abstract. Runways are vital descriptive features of airports and knowledge of their location is important to many aviation and military applications. With the recent wide availability of remote sensing data, there is demand for an automatic process of extracting runway geometry from satellite imagery. In particular, Very High Resolution (VHR) data makes it feasible to extract a runway's area precisely. In this paper we establish a novel method for accurate and precise extraction of geometric polygons for an arbitrary number of runways in VHR remote sensing imagery. Validated results are demonstrated for a dataset of twelve images of six different airports, at 61 cm resolution from the QuickBird II satellite.

1 Introduction

Modern day airports are complex structures which gradually extend over time: extensions, repairs and changes to existing features eventually alter the layout of most airports. For air traffic controllers, surface vehicle operators, security personnel and many other end users, access to up-to-date information on this layout is crucial to both safety and operational efficiency. Geographic Information Systems (GIS) can represent this information as a set of point, line and polygon objects overlaid on a VHR remote sensing image of an airport. These vector objects represent runways, taxilines, de-icing pads and other important features. Updating the GIS data then becomes a matter of comparing new remote sensing images with old, and with the existing GIS data, to isolate meaningful changes.

Human operators can perform this task by hand but doing so regularly for hundreds of airports is time consuming and can quickly become error-prone. This motivates the development of systems that can automate as much of the process as possible.

Runways are arguably the most prominent characteristic of an airport; this makes them a good starting candidate for automatic detection in VHR imagery. This paper introduces a novel algorithm that can accept a VHR remotely sensed image of an unknown airport as input, and fit a polygon accurately around each runway. For validation these polygons have been compared with those from existing GIS data for the airports, provided by Jeppesen.

* Correspondent author: boguslaw.obara@dur.ac.uk

2 Background

2.1 Related Work

Due to its importance in aviation and military operations, there have been several prior studies on automatic runway detection. Han *et al.* [5] segmented runways from low resolution images by using edge geometry to find long, homogeneous rectangles that were brighter than their surroundings. Tandra *et al.* [9] produced an edge detection and thresholding algorithm for use as part of a Foreign Object Detection (FOD) system. A real-time FOD system was developed by Kinaz *et al.* [7], which processes input from an infra-red camera mounted on the front of a plane; this system used convolution masks and straight line fitting to detect the runway's long edges and threshold markings.

Yang *et al.* [10] proposed a method that employed the Hough Transform (HT) in conjunction with Otsu thresholding and fractional gradient edge detection to find runways, but was unable to distinguish between runways and other straight objects.

These methods take diverse approaches but all share a few common assumptions: that the runways are long, straight objects or uniform intensity and that these criteria distinguish a runway from other features in the image.

Alternatively, Aytekin *et al.* [1] proposed a texture-based runway detection algorithm that used the Adaboost machine learning package to identify 32x32 pixel image tiles as runway or non-runway. The algorithm could learn from its training data which of 137 possible texture features were the most salient indicators, and construct its own classification scheme based on those.

2.2 Limitations of Traditional Methods

Very high resolution images of entire airports (and their surroundings) present challenges not fully addressed in the above papers. Unlike in the case of FOD systems, which monitor a single runway, an unknown number of runways are present in remote sensing images, along with many irrelevant features such as roads and residential areas. In lower resolution data, runways can appear as bright rectangles, surrounded by largely flat terrain; however, in the case of VHR remote sensing imagery, the background becomes much more complex. We also find that runways may be brighter or darker than their surroundings, or even change intensity along their length if different materials have been used for construction or parts of the runway have been renewed or extended. This makes intensity thresholding, as relied on in several previous studies (e.g. [5] [10] [3]), difficult to apply to these images.

The most obvious distinguishing feature of runways is that they are generally the longest, straightest objects present in an airport. The HT is the most obvious approach to extracting such an object, and has been applied to runway detection in the past [3]. The standard HT method [6] identifies the most prominent lines in an image by means of exhaustive search through a "Hough space", consisting of all possible straight lines subject to some quantisation in their parameters.

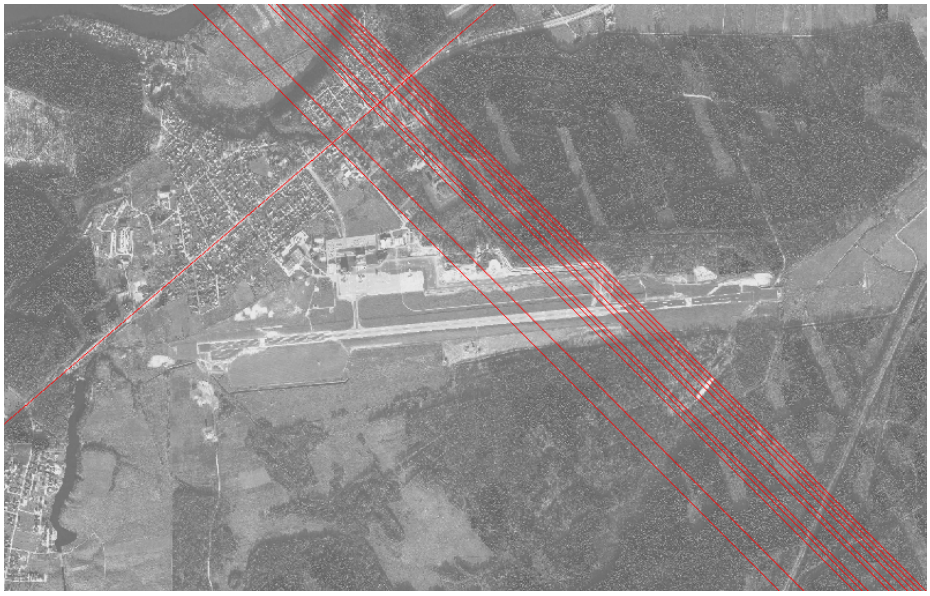


Fig. 1: Standard Hough Transform calculated from a Canny edge map of Kaunas International Airport. Lines are drawn preferentially through trees, since they produce many edge pixels in high resolution data.

These parameters, typically, are the line's closest approach to the origin, ρ, and the angle between the line's normal vector and the x-axis, θ. Since the standard HT detects straight lines rather than rectangles of non-zero width, the simplest application to runway detection is to produce a binary edge map, for example from the Canny [2] algorithm, and run the HT on that. In theory the HT will extract from it the long edges of the runways, as they should form the most prominent lines in the image.

This naive approach is severely limited, ultimately as the HT works by choosing the lines which pass through the most edge pixels. The dominant detected signals are often not runway edges, for several reasons:

- Many other objects in the image, such as taxiways, roads, tire marks down the centre of many runways and runway markings themselves, present competing edge pixels,
- The edges of runways are usually broken by adjoining taxiways, reducing number of edge pixels they produce in the edge map,
- Regions, which in lower resolution data may appear flat, such as trees and residential areas, produce many edge pixels in VHR data. Lines crossing these regions may intersect more edge pixels than lines along any runway edge, causing the HT to preferentially return these meaningless lines (see Figure 1).

A core limitation of the standard HT is that it is unable to automatically detect the number of lines; in general one must request at least as many lines as are present in the image. This is difficult in the case of previously unseen (or altered since last inspection) airports that may contain multiple runways.

3 Method

Our solution is a way of circumventing the problems caused by image complexity, so that the HT can be used to precisely locate the long edges of the runways. It begins by identifying the rough location of each runway in the image, in terms of its central axis line (in parametric ρ, θ form). To identify these central runway axes, we exploit the only property which is unique to and ubiquitous among all runways, namely, that they contain many runway markings, e.g. centre-line markings (Figure 3). These markings are white, elongated shapes on a dark background, easily detectable and with a well defined direction which is parallel to the runway. They can be extracted by searching the edge map for elongated loops of edges (as shown in Figure 2b). These objects are found by taking a Canny edge map [2] and removing from it all connected components that do not satisfy certain criteria (Figure, 2), in such a way that the only remaining components are elongated loops.

We begin by filtering components with too many or too few pixels. We found that fixed upper and lower bounds were sufficient for our dataset, in which there is negligible variation in both image resolution and the physical size of the markings. We then filter out components with too low an eccentricity, i.e. keeping long, thin elements, and finally remove components that are not closed loops. Roughly 20% of the remaining components correspond to runway markings.

Once the closed loops have been found, they can be used to populate an accumulator array analogous to the edge pixel-based accumulator array as used in the HT. However, they differ in that since runway markings have both a well defined direction and position, they can each lie along only one line, and hence cast a vote towards exactly one accumulator cell. The parameters of this line are given by,

$$\theta = \left(\phi - \frac{\pi}{2}\right) \mod \frac{\pi}{2} \quad \text{and}$$
$$\rho = x \cos\theta + y \sin\theta,$$

where ϕ is the angle the loop makes with the x-axis, and (x, y) are the coordinates of the loop's centroid. In this way, edge loops from the same runway will all increase the same accumulator cell, resulting in one clear peak per runway. The row and column indices of this peak in the accumulator array correspond to the runway's rho and theta parameters, respectively. To find the peaks, the transform space is thresholded at a fixed, empirical value of eight, i.e. at least eight markings must be found along the same line for a runway to be detected. This low threshold is possible because edge loops from runways will be be clustered at the same location in the transform space, while non-runway signals will be spread almost

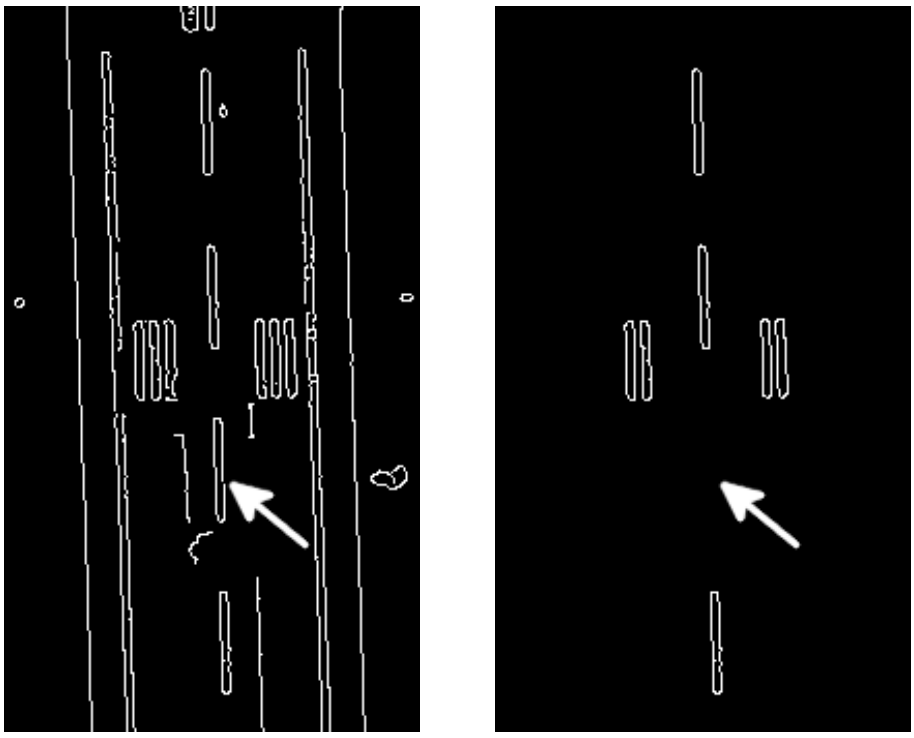


Fig. 2: A runway region from the Canny edge map, showing (a) original edges and (b) elongated loops left after connected component filtering. The arrow indicates a marking which was not detected, due to a small break in the edge loop.

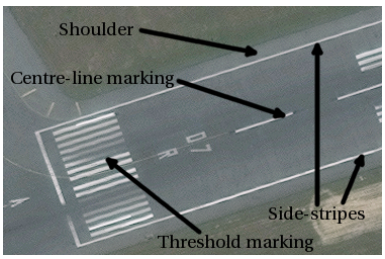


Fig. 3: At high resolutions, more detail becomes visible on a runway. For the purposes of our algorithm, we define the edges of the runway by its side-stripes, rather than its shoulder edges. Our algorithm also utilises threshold markings to delimit the ends of the runway, since they become discernible in VHR images.

randomly. Although, as shown in Figure 2, some markings are not detected, the runway peaks are still clearly distinguishable due to the low background noise.

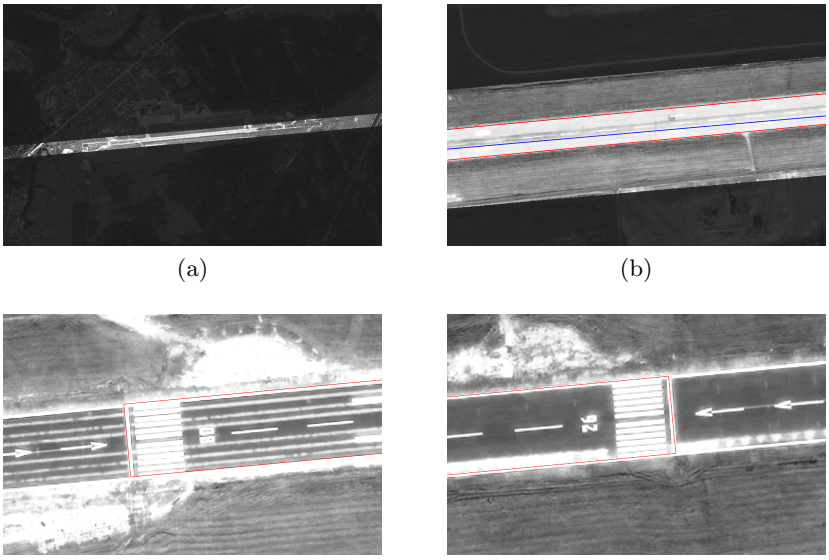


Fig. 4: Overview of the runway segmentation process. In (a), the region of interest - found by broadening the runway's axis line - is highlighted. In (b), a standard HT finds the long edges of the runway, displayed in red; the axis line is displayed in blue. In (c) and (d), the threshold markings have been found and the runway region capped to produce a rectangular polygon.

Once these lines are extracted, a region of interest can be produced for each runway by broadening the line into a strip. Since our dataset has constant resolution and little variation in the physical width of the runways, a constant strip width of 300 pixels was found to work on all images. The standard HT is then applied within this region to locate the long edges (see Figure 4). By restricting the HT to consider only edge pixels from within the region of interest, we remove the majority of the non-runway edge pixels and make the runway edges appear far more prominent in the Hough space. This also reduces running time for the operation, since fewer pixels must be processed.

Since we define the boundary of the runway as the white side-stripes rather than, for example, the outer edges of its shoulders (see Figure 3), we use the result of a local mean threshold as the input to the HT, rather than an edge map. The local mean threshold responds well to the white objects on a dark background, and responds less strongly to the edges of bright regions. In this way we ensure the HT detects specifically the white side-stripes, rather than any other long edges.

Due to the length of runway edges in VHR data (at least 3000 pixels in our dataset), an angular resolution of one sixteenth of a degree is used in the HT to allow the lines to fit closely along the full length of the runway.

Having extracted the runway's long edges, it is still necessary to find the threshold markings that terminate the runway (see Figure 3). These markings are distinctive in that they are periodic: normally 4-16 (depending on width) [4] white stripes of even spacing and thickness. Our algorithm identifies these markings by analysing independently all columns of pixels from the strip, each of which is a cross-section through the runway. Columns are classified as threshold or non-threshold; the classification requires three criteria to be met:

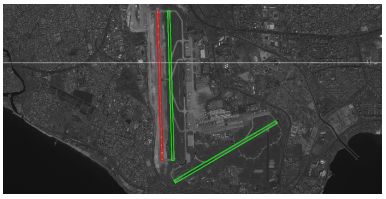
- Brightness: thresholds are marked in white paint, therefore a column that runs across one must have a greater mean intensity than that of the whole strip.
- Periodicity: threshold markings are stripes that repeat at a regular frequency (see Figure 3). As such, a clear frequency peak should be found in the column's Fourier transform.
- Number of stripes: threshold markings consist of 4-16 white stripes. In a single column, this appears as 4-16 intensity peaks. We count these by performing a local mean threshold within the column, and counting the number of connected components that appear. Only columns with the appropriate number of stripes for their width are classified as threshold; this allows for some noise and or clustering of signals.

Once the columns have been classified, the threshold markings will appear as large contiguous groups of threshold marked columns, the outer edges of which mark the boundaries of the runway.

4 Results

Our dataset consists of six bi-temporal pairs of images from the QuickBird II satellite, with a panchromatic resolution of 61 cm. These images vary in size from around 25-200 megapixels. Figure 5 presents the algorithm's output on three of six airports from the input dataset, provided by Jeppesen. In some cases the airport itself comprises only a small proportion of the total original image area.

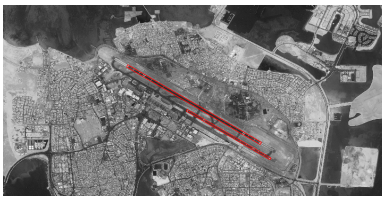
Between all twelve images, there are 26 runways, all but two of which are detected by the algorithm. In practice, the algorithm's accuracy is found to depend on the clarity of threshold markings - if they are too faint due to weathering then they will not be conclusively detected, and the runway's existence will not be reported. This issue is clearly shown in Figure 6, the sixth airport, where the runway marking are dulled by apparent weathering. These issues might be partially mitigated by using some pre-processing, such as histogram equalisation. All images have been processed appropriately for presentation.



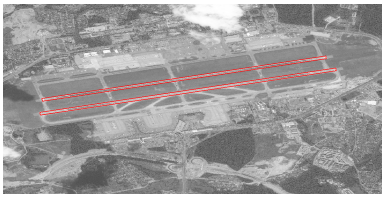
(a) Istanbul (????)

(b) Istanbul (????)

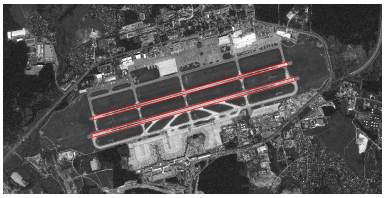
(c) Bahrain (May 2011)



(d) Bahrain (April 2014)

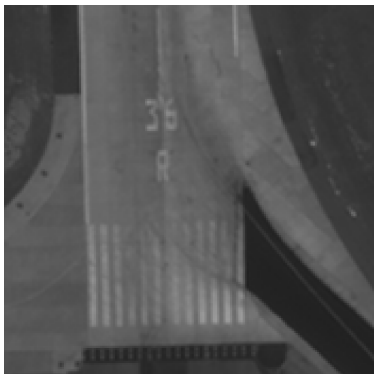


(e) Moscow (August 2011)



(f) Moscow (July 2014)

Fig. 5: The algorithm's output for six representative images from of our sample dataset, provided by Jeppesen. These images have been cropped to focus on the runway region, which can be a small proportion of the original images. The green boxes show the ground truth (from pre-existing GIS data) for the two runways which were not detected.



(a)

(b)

Fig. 6: Heavily eroded threshold markings caused by taxiing planes lead to the runways remaining undetected.

Table 1: Jaccard indices, comparing detected runway polygons with those marked by human experts.

Airport	Runways Detected	Jaccard Index
Kaunas (2008)	1 of 1	0.978
Kaunas (2009)	1 of 1	0.980
Bahrain (2011)	2 of 2	0.976
Bahrain (2014)	2 of 2	0.969
Hong Kong (2008)	2 of 2	0.974
Hong Kong (2014)	2 of 2	0.978
Moscow (2011)	2 of 2	0.969
Moscow (2014)	2 of 2	0.970
Helsinki (2008)	3 of 3	0.967
Helsinki (2014)	3 of 3	0.974
Istanbul (2006)	1 of 3	0.954
Istanbul (???)	3 of 3	0.968

5 Conclusion

An algorithm for the precise detection of runways from VHR remote sensing data is presented. We find regions of interest by observing features that are not discernible in lower resolution imagery, and produce vector polygons that precisely fit the runway area. Accurate results with a zero false-positive rate are demonstrated in our testing dataset. The algorithm is robust against variations in runway intensity and outline shape due to adjoining taxiways and other artefacts, and against the presence of large tire marks down the runway centre. It can detect a variable number of runways, and does not return false-positives by confusing runways with taxiways or other background features such as roads.

The vector objects which are produced will make it simple to compare runway areas in multi-temporal data, provided the image pairs can be co-registered, a challenge we are currently focussed on. In addition, a number of performance improvements can be made. For instance, although finding edge loops has proven the most effective way of detecting runway markings so far, it is a lengthy process, and could potentially be replaced by a local thresholding operation, a selection of which are listed in chapter eight of [8].

Acknowledgments

This research was supported by Jeppesen, a Boeing Company; grant number RF081BO.

References

1. Aytekin, O., Zongur, U., Halici, U.: Texture-based airport runway detection. Geoscience and Remote Sensing Letters, IEEE 10(3), 471–475 (May 2013)
2. Canny, J.: A computational approach to edge detection. IEEE Transactions on Pattern Analysis and Machine Intelligence 8(6), 679–698 (1986)
3. Di, N., Zhu, M., Wang, Y.: Real time method for airport runway detection in aerial images. In: Audio, Language and Image Processing, 2008. pp. 563–567 (July 2008)
4. Federal Aviation Administration: Aeronautical Information Manual, chap. 2.3 (2014)
5. Han, J., Guo, L., Bao, Y.: A method of automatic finding airport runways in aerial images. In: Signal Processing, 2002 6th International Conference on. vol. 1, pp. 731–734 (Aug 2002)
6. Hough, P.V.C.: Method and means for recognizing complex patterns (1962)
7. Kniaz, V.: A fast recognition algorithm for detection of foreign 3d objects on a runway. In: PCV14. pp. 151–156 (2014)
8. Sezgin, M., Sankur, B.: Survey over image thresholding techniques and quantitative performance evaluation. Journal of Electronic Imaging 13(1), 146–168 (2004), http://dx.doi.org/10.1117/1.1631315
9. Tandra, S., Rahman, Z.: Robust edge-detection algorithm for runway edge detection. In: Proc. SPIE. vol. 6813 (2008)
10. Yang, Z., Zhou, J., Lang, F.: Detection algorithm of airport runway in remote sensing images. TELKOMNIKA Indonesian Journal of Electrical Engineering 12(4), 2776–2783 (2014)