Runway Detection for Autonomous Aircraft Landing Using Digital Signal Processing Techniques

Peter A. Sayegh*, Haasini R. Nandyala†, Daniel C. Wang‡, Sharvani Vadlamani§
Department of Electrical Engineering
Columbia University, New York, NY 10027
Email: *pas2232@columbia.edu, †hrn2111@columbia.edu, ‡dcw2146@columbia.edu, §sv2734@columbia.edu

***Abstract*—This report presents a digital signal processing (DSP) approach to runway detection for autonomous aircraft landing systems. The project leverages fundamental DSP techniques including edge detection, Hough transform, and frequency domain analysis to identify and track runway structures from aircraft camera imagery. The proposed system processes visual data through a multi-stage pipeline designed to extract geometric and textural features characteristic of runway environments. This project demonstrates practical applications of classical signal processing theory to modern autonomous vehicle navigation challenges.**

I. Introduction

Autonomous aircraft landing systems require robust vision-based runway detection to complement GPS and Instrument Landing Systems, particularly in GPS-denied environments. Monocular cameras provide lightweight, cost-effective sensing but introduce challenges including absence of depth information and scale ambiguity, requiring geometric reasoning to infer runway structure from two-dimensional projections. This work frames runway detection as a signal processing problem: given a single aerial image during approach, identify runway edges despite perspective distortion, illumination variations, and background clutter. We employ classical computer vision techniques, such as edge detection, Hough Transform, and RANSAC filtering, rather than deep learning, motivated by limited aviation training data, requirements for algorithmic interpretability in safety-critical systems, and the structured nature of the problem. Runways exhibit strong geometric priors: approximate linearity, parallel edges, and dominance as the primary scene feature. These properties are naturally exploited through analytical methods that transform spatial-domain edge measurements into parameter-space line representations, followed by consensus-based model selection. The proposed pipeline demonstrates that fundamental digital signal processing techniques, appropriately constrained by domain knowledge, achieve effective detection on structured problems without extensive training data or opaque neural architectures.

II. Methodology

A. Overview of the Pipeline

Figure 1 illustrates the modules of our runway detection pipeline.

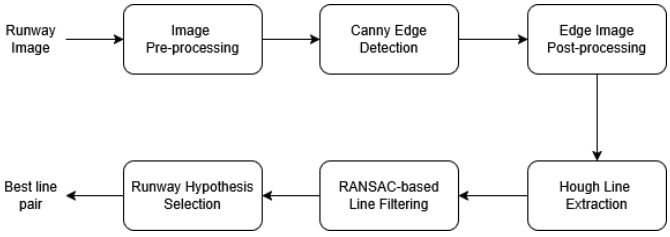

Fig. 1. Runway Detection Pipeline

B. Image Processing and Edge Detection

Before the image can be sent to RANSAC to detect the runway, it first needs to be processed to remove all unnecessary noise. This process includes pre-processing to smooth out artifacts, before running the Canny edge detector to create an edge image. This edge image contains a lot of noise which would cause the hough transform to detect many extraneous lines, so much of the noise is removed through post-processing of the edge image.

1) Pre-processing: Gaussian smoothing is performed on the FFT-processed images to reduce isolated high-frequency artifacts and fragmentation of long edges into short segments. It also improves the gradient stability for Canny edge detection

Gaussian smoothing convolves the image with the following Gaussian kernel:

$$G(x, y; \sigma) = \frac{1}{2\pi\sigma^2} \exp\left(-\frac{x^2 + y^2}{2\sigma^2}\right), \tag{1}$$

The smoothing scale is kept small to avoid blurring the runway edges.

2) Edge Detection: The edges are extracted using the Canny edge detector, which consists of the following steps

(a) Finds the intensity of horizontal and vertical gradients of the image to obtain gradient magnitude and direction
(b) Applies gradient magnitude thresholding or lower bound cut-off suppression to get rid of spurious response to edge detection
(c) Thins edges by keeping pixels that are local maxima along the gradient direction
(d) Chooses percentile-based upper and lower thresholds based on the strongest gradients
(e) Applies double threshold to determine potential edges by classifying edge pixels as strong edges (above high

threshold), weak edges (below thresholds), and non-edges (below low threshold).

(f) Tracks edges with hysteresis; finalizes the detection of edges by suppressing all the weak edges not connected to strong ones

Canny edge detection was chosen because it gives thin, well-localized edges while suppressing noisy gradient responses.

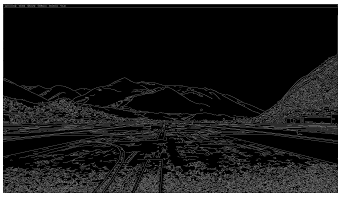

Fig. 2. Canny Edge Image

3) Post Processing: The resulting Canny edge image contains large amounts of noise from textures which would lead to many extraneous lines created during the Hough transform. Several steps are implemented attempting to remove these noisy edges.

In the first step, the edge image undergoes morphological cleanup, which consists of three steps.

(a) Edges are thinned out to single-pixel thickness, which improves geometric consistency for later line-fitting. It also helps simplify connect-component analysis which will be conducted later.

(b) Small isolated pixels and short branches are removed to reduce noise.

(c) The junctions of long curved/squiggly lines are broken up so they can be removed later. The goal is to keep only simple, straight lines which would be candidates for the runway edge.

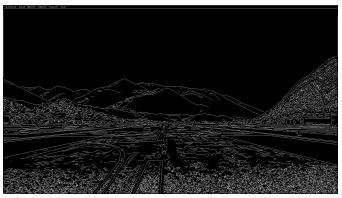

Fig. 3. Edge Image after Morphological Cleanup

The second step is a connected-component filter, which removes all continuous lines shorter than a certain threshold, as well as closed areas under a certain area in an attempt to remove non-runway edges.

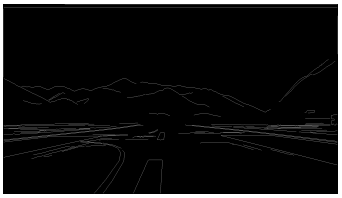

Fig. 4. Edge Image after Connected-component Filter

This step does a good job in removing most of the smaller edges which capture unwanted texture. But many long edges which aren't part of the runway still exist.

The next step only keeps the N longest components, with the idea that the runway edges will be some of the longest present. For this step, N was set to forty, a conservative number, as more aggressive settings would start to remove runway edges which were segmented when transformed to Canny edges.

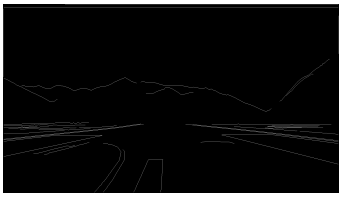

Fig. 5. Edge Image after Longest Components Filter

This step performed well on images with many edges that survived the connected-component filter, but the background, which typically contained long edges from mountains or clouds, would still remain.

If we were certain that the runway would always only exist within the bottom half of the images in the dataset, the top half could be safely removed. This technique wouldn't work on our dataset, however, as many images had the runway in the center or even only in the top half of the image. The resolution to this issue was to create an adjustable region of interest (ROI) function. This function would compute the vertical edge distribution and find the cutoff row where 10% of edges are above the line. All edges above this row are removed. The function would refrain from removing any edges if the original image already contained very few edges. The function was kept conservative to make sure that the runway edges would never be removed through the function.

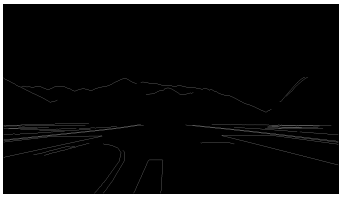

Fig. 6. Edge Image after ROI detection

C. Hough Transform for Line Detection

The Hough Transform [1] maps edge pixels from image space to parameter space, enabling robust line detection despite noise and fragmentation.

Parameterization in (ρ, θ) space: Lines are represented in polar form as $\rho = x\cos\theta + y\sin\theta$, where ρ denotes the perpendicular distance from the origin to the line and θ represents the angle of this perpendicular. This parameterization avoids singularities encountered with slope-intercept form and naturally handles vertical lines. The implementation discretizes θ from 0° to 179° in 1° increments and ρ from

$-D$ to $+D$ in unit steps, where $D = \lceil \sqrt{\text{width}^2 + \text{height}^2} \rceil$ represents the image diagonal. Image coordinates follow the MATLAB convention, where the origin is at the top-left corner, x increases to the right and y increases downward. The Hough formulation is directly applied to this MATLAB coordinate system.

Accumulator construction: For each edge pixel (x_i, y_i), the algorithm computes ρ for all discrete θ values and increments the corresponding accumulator cell $A[\rho, \theta]$. Computational efficiency is achieved through precomputed trigonometric values and vectorized ρ calculation across all angles simultaneously. Each edge pixel votes for all lines passing through it, causing true linear structures to produce peaks in the accumulator where multiple votes coincide.

Peak detection and thresholding: Peaks are identified using adaptive percentile-based thresholding combined with non-maximum suppression. The threshold is set at the 90th percentile of accumulator values to capture both strong line characteristic of perspective-distorted runways. Peaks are then selected by repeatedly identifying the global maximum in the accumulator and suppressing a large neighborhood around it to prevent duplicate detections of the same physical line. Suppression is performed using an anisotropic neighborhood in (ρ, θ) space, ensuring each retained peak corresponds to a distinct line hypothesis rather than a discretization artifact. Detected peaks are sorted by vote strength, with higher accumulator values indicating stronger global support in the edge image.

D. RANSAC-Based Line Filtering

Random Sample Consensus (RANSAC) provides robust line pair selection in the presence of spurious detections from image clutter and background features.

Motivation: eliminating spurious lines from clutter. The Hough Transform generates numerous line candidates, many arising from non-runway features such as terrain boundaries, cloud edges, shadows, or image artifacts. A direct pre-filtering approach removes near-duplicate lines within 50 pixels to reduce redundancy. RANSAC then iteratively samples line pairs and evaluates their geometric consistency with the edge data, effectively distinguishing true runway boundaries from false positives through consensus-based model selection.

Inlier definition and distance metric. A point (x_i, y_i) is classified as an inlier if its perpendicular distance to either line satisfies $d_i = |x_i \cos\theta_k + y_i \sin\theta_k - \rho_k| < \tau$, where $k \in \{1, 2\}$ indexes the two runway edge candidates and $\tau = 50$ pixels is the distance threshold. This bilateral inlier criterion accounts for the dual-edge structure of runways, counting edge pixels supporting either boundary. The perpendicular distance metric directly derives from the polar line representation, providing efficient vectorized computation across all edge pixels.

Line pair evaluation and scoring: The angular separation of every valid line pair which passed the pre-filters, and those below 3° or exceed 35° are rejected, as runway edges almost always fall within this range. For valid pairs, inlier counts are computed independently for each line. The final score combines total inlier support with a balance term that favors pairs where both lines receive comparable support. The line pair which maximizes this combined score is selected as the runway hypothesis.

E. Runway Geometry Constraints

Physical runway geometry imposes structural constraints that serve as band pass filters in both spatial and angular domains, enhancing detection specificity.

Approximate parallelism of runway edges: True runway boundaries exhibit near-parallel geometry despite perspective foreshortening. The implementation enforces $|\theta_1 - \theta_2| < \epsilon_{\text{parallel}}$, where $\epsilon_{\text{parallel}} = 30$ accommodates moderate perspective distortion from oblique viewing angles while rejecting arbitrary line pairs. This constraint exploits the fundamental property that parallel lines in world coordinates maintain bounded angular separation under projective transformation, effectively filtering non-runway line combinations.

Vanishing point consistency: Under perspective, runways typically converge to a common vanishing point located above the runway footprint. if the candidate line pairs have a vanishing point which lie outside a plausible image region, or don't exist due to parallelism, they are rejected. This region is defined as $v_y < 0.8H, v_y > -H, v_x > -0.75W, v_x < 1.5W$, where (v_x, v_y) is the coordinate of the vanishing point, and H, W is the height and width of the image.

Expected angular range: Runway edges in aerial approach imagery exhibit characteristic orientations determined by camera geometry and runway heading. The angular band pass filter retains lines with $\theta \in [0 - 80, 100 - 180]$, explicitly rejecting near-horizontal features ($\theta \in [80 - 100]$) corresponding to horizon lines, cloud layers, or terrain boundaries. This constraint leverages prior knowledge that runway edges appear as approximately vertical or moderately oblique linear structures in downward-looking camera views, implementing domain-specific frequency selectivity in orientation space.

Spatial consistency across the image: Properly framed approach imagery centers the runway structure within the field of view. A spatial attention mechanism restricts line detection to the horizontal band $x \in [0.25W, 0.75W]$, where W denotes image width, eliminating peripheral clutter while focusing computational resources on the high-probability runway region. Additionally, a 10-pixel border mask removes artificial edges arising from image boundaries, preventing false detections from frame artifacts. These spatial constraints implement region-of-interest filtering analogous to windowing functions in frequency-domain analysis, suppressing out-of-band spatial components that violate the expected runway position prior.

III. Experimental Setup

A. Dataset

The dataset used in this project consists of 5,587 monocular runway-approach images captured in Microsoft Flight Simulator 2020 and released via Kaggle. The imagery spans a wide range of operational conditions, including clear and

overcast daylight, foggy/low-visibility approaches, and nighttime scenes. These variations introduce common failure modes for classical vision pipelines: reduced runway-to-background contrast, partial occlusion of runway boundaries, competing strong edges from terrain or clouds, and strong perspective distortion during final approach.

1) Ground truth masks and edge-based reference: A key advantage of this dataset is the availability of per-image binary ground-truth (GT) runway masks provided with the Kaggle dataset. These masks define the visible runway surface at pixel resolution and serve as dense spatial supervision. For line-based evaluation, runway boundaries are derived by extracting the mask contour using Canny edge detection, yielding the two dominant runway edges under perspective projection. Using mask-derived edges preserves runway geometry under scale changes, truncation, and partial visibility, enabling robust geometric comparison with detected line structures.

2) Dataset labeling for controlled evaluation: To enable reproducible and controlled experimentation, images were curated and labeled using a lightweight metadata schema derived from automated signal-processing analyses rather than manual visual inspection. Each image was assigned categorical attributes describing acquisition conditions based on measurable image statistics and geometric cues: *time_of_day*, *weather*, *lighting*, *blur*, *runway_distance*, and *view_angle_bin*. These labels were inferred using combinations of intensity histograms, contrast measures, frequency-domain energy distribution, edge density, and runway scale estimated from the ground-truth mask. The labels are not used by the detection algorithm itself; instead, they support stratified sampling and controlled evaluation across environmental conditions.

B. Evaluation and Testing Methodology

Evaluation is structured to reflect the two-stage nature of the detection pipeline. The first stage measures whether the Hough Transform produces line hypotheses aligned with the physical runway boundaries. The second stage evaluates whether RANSAC successfully selects an accurate pair of runway edges from these candidates. Both stages operate on geometric comparisons between detected lines and GT-derived runway edges using distance- and orientation-based metrics.

1) Hough Transform evaluation: runway edge coverage: The Hough-stage evaluation assesses whether the line extraction process identifies runway-aligned structures, independent of subsequent model selection. For each image, all filtered Hough line candidates are compared against the GT runway edges.

Each Hough line is uniformly sampled within a vertically constrained region of interest (ROI) derived from the GT mask extent. For each sampled point, the distance to the nearest GT runway edge pixel is computed using a distance transform of the binary mask. A Hough line is considered to match a runway edge if a sufficient fraction of its sampled points lie within a fixed distance threshold (10 pixels).

At the image level, results are summarized by counting how many of the two GT runway edges are supported by at least one Hough line:

- **0-edge match**: no runway edge is detected,
- **1-edge match**: exactly one runway edge is detected,
- **2-edge match**: both runway edges are detected.

This metric captures the presence of usable runway structure prior to RANSAC, without enforcing pairing or global consistency.

2) RANSAC evaluation: final runway edge accuracy: The RANSAC-stage evaluation measures the geometric accuracy of the final runway hypothesis, which consists of exactly two detected line estimates. Each detected line is compared directly to the corresponding GT runway edge.

Lines are sampled within the same GT-derived vertical ROI and evaluated using distance transforms. Spatial accuracy is quantified using:

- **Hit@10**: fraction of sampled points within 10 pixels of the GT edge,
- **MeanDist**: mean distance (in pixels) to the GT edge.

Orientation consistency is evaluated by estimating a dominant GT runway orientation using principal component analysis (PCA) on the GT mask boundary. Angular error is computed modulo π, and an angular pass metric (**AngPass@15**) reports whether detected runway edges fall within 15 degrees of the GT orientation.

Unlike the Hough-stage evaluation, which measures the availability of runway-aligned evidence, the RANSAC evaluation explicitly measures whether the algorithm correctly identifies both runway edges with accurate geometry, enabling separation of detection and model-selection failures.

3) Controlled subset evaluation: Evaluations are performed on controlled, stratified subsets to analyze performance under increasing visual difficulty. The primary evaluation set consists of 150 images categorized as *easy*. These images are close to the runway and feature clear weather, minimal angle, and favorable lighting, resulting in well-defined and unobstructed lane markings. This subset serves as a baseline for assessing Hough-stage line detection and RANSAC-stage hypothesis selection under near-ideal conditions.

Three additional subsets introduce increased difficulty by modifying a single environmental factor at a time. The fog subset reduces global contrast while largely preserving scene structure. The night subset introduces low illumination and glare, degrading edge clarity and increasing noise. The medium-distance subset increases the camera–lane separation, reducing lane pixel resolution and making line detection more sensitive to noise and parameter settings.

This controlled design enables attribution of performance degradation to specific visual factors—contrast loss, noise, and structural ambiguity—providing a clear definition of image difficulty and supporting targeted analysis of parameter effectiveness.

IV. Results

A. Qualitative Results

Fig. 7. Successful runway detection under clear, near-range conditions

Fig. 8. Detection failure at increased runway distance

Fig. 9. Night-time failure due to incorrect RANSAC line selection

Figures 7-9 illustrates representative qualitative outcomes of the proposed runway detection pipeline across varying conditions. These examples are selected to highlight typical success cases as well as dominant failure modes observed in the quantitative evaluation.

Successful detection (clear, near-range). In favorable conditions with strong contrast and well-defined runway boundaries, the Hough Transform produces multiple runway-aligned line hypotheses, and RANSAC successfully selects the correct pair corresponding to the physical runway edges. In these cases, the detected lines closely align with the ground-truth mask boundaries both spatially and in orientation, resulting in high Hit@10 and low mean distance error.

Partial failure under night-time conditions. In night-time imagery, runway lighting and artificial light sources introduce competing high-contrast edges. While the Hough Transform often still detects runway-aligned lines, RANSAC may incorrectly select a pair dominated by non-runway structures (e.g., lighting artifacts or background edges). This failure mode illustrates that sufficient Hough evidence does not always guarantee correct model selection when geometric cues are ambiguous.

Failure under increased runway distance. For medium-distance approaches, the runway occupies a smaller image footprint and exhibits stronger perspective compression. In these cases, both Hough line coverage and RANSAC performance degrade substantially. Detected lines frequently fail to align with either runway edge, reflecting the sensitivity of purely geometric, line-based methods to scale and perspective distortion when the runway spans few pixels.

These qualitative observations align closely with the quantitative trends reported in Section IV.B, reinforcing that most failures arise from either insufficient line evidence at the Hough stage or incorrect line-pair selection during RANSAC under degraded visibility or extreme perspective.

B. Quantitative Analysis

TABLE I
QUANTITATIVE RUNWAY EDGE DETECTION RESULTS ACROSS CONTROLLED DATASET SUBSETS. HOUGH-STAGE RESULTS REPORT THE PERCENTAGE OF IMAGES IN WHICH FILTERED HOUGH LINES MATCH 0, 1, OR BOTH GROUND-TRUTH RUNWAY EDGES WITHIN A 10-PIXEL THRESHOLD. RANSAC-STAGE RESULTS REPORT SPATIAL ACCURACY OF THE FINAL SELECTED RUNWAY EDGE PAIR.

Dataset	**Hough Edge Matches (%)**			**RANSAC Accuracy**	
	0 edges	1 edge	2 edges	Hit@10	MeanDist (px)
150_easy (clear, near)	33.33	15.87	50.79	0.3166	71.68
150_medium (clear, mid)	80.60	5.97	13.43	0.0232	351.60
77_fog (near)	30.16	20.63	49.21	0.3329	64.20
150_night (near)	44.55	21.78	33.66	0.2714	79.51

Table I summarizes quantitative performance across four controlled dataset subsets, separating intermediate Hough line coverage from final RANSAC runway edge accuracy. This separation allows failures in geometric model selection to be distinguished from failures in line evidence extraction.

Clear, near-range approaches (150_easy). Under favorable conditions, the Hough Transform frequently identifies runway-aligned structure, with both runway edges detected in over 50% of images. At least one runway edge is detected in two-thirds of images. Given sufficient line evidence, RANSAC achieves moderate spatial accuracy, indicating that geometric constraints are effective when the runway occupies a large portion of the image.

Effect of distance (150_medium). Increasing runway distance significantly degrades performance. Hough-stage detection collapses, with over 80% of images failing to match either runway edge. With increased runway distance, the edge appears shorter in the image leading to shorter peaks during the Hough transformation. The current algorithm aggressively filters out lines with lower relative peaks (90th percentile of accumulator values), so these lines are removed before RANSAC. This lack of line evidence propagates to RANSAC, resulting in near-zero Hit@10 and large mean distances. These results highlight the sensitivity of geometry-based constraints to scale and perspective distortion.

Effect of visibility (77_fog). Fog primarily reduces contrast but preserves runway geometry. Hough-stage performance remains comparable to clear near-range conditions, and RANSAC accuracy is not substantially degraded. Fog/haze acts as a low-pass filter by removing high-frequency detail and contrast. If the visibility isn't too impacted, the "filter" removes only unnecessary textures such as the asphalt, which actually helps during edge-detection.

Effect of illumination (150_night). Night-time imagery introduces both sparse edges and competing artificial lighting. The reduced lighting causes lower contrast leading to lower intensity edges Hough-stage coverage decreases relative to clear conditions, and RANSAC accuracy degrades accordingly. Nevertheless, performance remains significantly better than in

the medium-distance case, reinforcing the dominant role of runway scale in the image.

Overall, the results demonstrate that Hough-stage runway coverage is a necessary prerequisite for successful RANSAC selection, and that distance-induced geometric changes represent the primary limitation of the proposed line-based pipeline.

V. Discussion

A. Hough and RANSAC Parameter Selection

Due to time constraints, our primary objective was making sure the runway detection pipeline could perform reliably on the easiest images in the dataset (daytime, clear weather, close and well-defined runway). Parameters were therefore tuned using this subset, with the goal of establishing a stable and interpretable baseline that could be extended to more challenging scenarios in the future.

The Hough transform parameters were selected to aggressively suppress noise and clutter. In particular, the high cutoff (90th percentile of accumulator values) favors long, continuous runway edges with strong global support, while filtering out weaker line responses caused by texture, shadows, or background structures. This design gives RANSAC a significantly smaller candidate line set, increasing the probability of choosing the correct line pair. However, in images where runway edges are shorter, partially occluded, or less contrasted, valid runway edges are filtered out before RANSAC is applied, giving no chance in a successful detection. Lowering the threshold would improve robustness to such cases by allowing a broader set of candidate lines, but increases the chance of RANSAC choosing an incorrect pair of lines.

The RANSAC parameters were similarly tuned for these easy images. Constraints on angular separation, vanishing point location, and runway width reflect the strong perspective cues present when the runway is relatively close to the camera. For runways further away, where perspective convergence is weaker, the edges appear more nearly parallel and fall outside of the current parameters. Improving generalization to such cases would require relaxing angular tolerances and widening acceptable runway width bounds. While these adjustments would increase robustness across a broader range of runway geometries, they would also reduce discriminative power in simple scenes by expanding the hypothesis space and increasing sensitivity to clutter-induced line pairs.

Overall, parameter selection in this pipeline reflects an explicit tradeoff between precision and robustness. The current configuration prioritizes high-confidence detection for a narrow set of geometries which fit the easy images of our dataset at the expense of generality. Extending the pipeline to handle more diverse conditions would require either sacrificing performance on specific geometries or introducing adaptive parameter strategies that adjust thresholds and constraints based on scene characteristics.

VI. Conclusion

This project demonstrates that classical digital signal processing techniques can effectively tackle low level computer vision problems such as runway detection. By integrating edge detection with Hough-based line extraction and RANSAC filtering, we demonstrate that training-free techniques can effectively recover runway geometry under favorable conditions. The results clarify both the advantages and limitations of purely geometric pipelines, particularly their sensitivity to scale changes and perspective effects at longer ranges. Taken together, this work underscores the continued relevance of principled signal processing methods in safety-critical tasks, while pointing to the need for future enhancements that improve robustness across more challenging operating conditions.

VII. Future Work

Several extensions would enhance the system's robustness and operational scope. **Temporal filtering** via Kalman filtering would enable multi-frame tracking of runway state vectors $\mathbf{x}_k = [x, y, \theta, w, v_x, v_y, \omega]^T$, smoothing detections across video sequences and rejecting false positives through temporal consistency. **Texture analysis** using Gabor filter banks would complement edge-based detection by characterizing oriented surface patterns, improving performance under low-contrast conditions where edges alone are unreliable.

Hybrid classical-learning architectures represent a promising direction, where convolutional neural networks provide robust region proposals or edge maps that feed into the Hough-RANSAC pipeline. This approach would combine data-driven adaptability with interpretable geometric reasoning, preserving the theoretical foundations of signal processing while exploiting deep learning's capacity for appearance modeling. Finally, **stochastic model refinement** through Bayesian model selection or information-theoretic criteria would extend RANSAC beyond maximum likelihood, balancing model complexity against data fit for improved generalization.

Supplementary material

The project's GitHub repository can be accessed on this link: github.com/peter-sayegh/runway-detection-dsp/tree/main The dataset used for this project can be assessed on this link: kaggle.com/datasets/relufrank/fs2020-runway-dataset

References

[1] D. A. Forsyth and J. Ponce, *Computer Vision: A Modern Approach*. Pearson, 2nd ed., 2012.

Appendix

Prior to spatial-domain preprocessing, we analyzed input images in the frequency domain to isolate structural components relevant to runway geometry. Runway edges correspond to strong, low-curvature linear features, which manifest as concentrated energy along specific orientations in the Fourier domain.

A. Fourier Transform Formulation

Given a grayscale image $I(x, y)$, its 2D discrete Fourier transform (DFT) is defined as

$$F(u, v) = \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} I(x, y) \exp\left(-j2\pi \left(\frac{ux}{M} + \frac{vy}{N}\right)\right). \quad (2)$$

We visualize the log-magnitude spectrum $\log(|F(u, v)|)$ to examine dominant spatial frequencies and directional structure.

B. High-Pass Filtering

To suppress low-frequency illumination variations and emphasize edge-like structures, we apply a high-pass filter in the frequency domain:

$$\tilde{F}(u, v) = H(u, v) \cdot F(u, v), \quad (3)$$

where $H(u, v)$ attenuates low-frequency components near the DC region while preserving higher spatial frequencies. This helps us have a better baseline for running the edge detection algorithm because low frequency background structures are suppressed, while the edges of the runway remain.

The filtered image $\tilde{I}(x, y)$ is recovered via the inverse Fourier transform:

$$\tilde{I}(x, y) = \mathcal{F}^{-1}\{\tilde{F}(u, v)\}. \quad (4)$$

C. Impact on Downstream Edge Detection

FFT-based filtering improves robustness to gradual intensity gradients and atmospheric effects, but can also amplify high-frequency noise. It's effective in suppressing periodic or structured noise such as parts of the image with repetitive textures (asphalt grain, grass striping, etc.). Global weather conditions with repetitive high-frequency components such as heavy rain or snowfall would also theoretically benefit.

We empirically evaluated the trade-off between improved edge contrast and increased spurious responses in subsequent Canny edge detection. Our results on average decreased after FFT-filtering so ultimately we decided to remove it from the final pipeline.

This analysis motivates selective use of frequency-domain pre-processing. If given more time, FFT-filtering would be best used selectively given the context of the image. For example if an image was detected with a weather condition such as rain, FFT-filtering would be applied.

D. Removed Post-processing features

Several methods of post-processing were implemented to remove noise which ended up too aggressive or ineffective and were removed.

Linearity filter: This filter attempted to remove lines that did not meet a specified linearity ratio, defined using the relationship between area and perimeter (area = perimeter/2 with perimeter ¿¿ 2*area). However, when the threshold was too strict, runway edges in blurrier images were removed, while looser thresholds failed to filter anything meaningful. Because no stable operating point could be found, this filter was ultimately discarded.

PCA filter: Another approach to assess linearity applied PCA to each connected component, exploiting the fact that straighter lines exhibit high variance along one principal axis and minimal orthogonal variance. This method suffered from the same sensitivity issues as the linearity filter and was therefore removed.

Orientation filter: This filter estimated edge orientation using image gradients and measured local orientation coherence within small windows, removing edges with insufficient gradient magnitude or directional agreement. Although theoretically more robust and less sensitive to outliers than PCA due to its local nature, it consistently removed nearly all edges even with relaxed parameters and was consequently discarded.

E. Contributions

Peter conceived the project concept and developed the initial proposal, establishing the technical framework and system architecture. Peter implemented the core line detection algorithm using the Hough Transform, developed the RANSAC-based filtering module, and designed the geometric validation constraints based on runway structure.

Haasini was responsible for dataset acquisition and curation, categorizing and labeling images according to difficulty tiers. She established additional classification criteria to distinguish between easy and challenging test cases, and developed the validation algorithm using binary map comparison to quantify detection performance.

Sharvani implemented the frequency-domain preprocessing pipeline, developing the 2D FFT algorithm and designing the high-pass filter to reduce noise prior to edge detection.

Daniel implemented the image processing and Canny edge detector. He added global peak detection and the line drawing algorithm in the Hough transformation, as well as the scoring system and vanishing point feature in RANSAC. Daniel streamlined the separated detection steps and combined them together to implement the full pipeline for runway detection, as well as performed parameter tuning.