The 2010 IEEE/RSJ International Conference on
Intelligent Robots and Systems
October 18-22, 2010, Taipei, Taiwan

A Vision-Based Boundary Following Framework for Aerial Vehicles

Anqi Xu and Gregory Dudek

Abstract— **We present an integration of classical computer vision techniques to achieve real-time autonomous steering of an unmanned aircraft along the boundary of different regions. Using an unified conceptual framework, we illustrate solutions for tracking coastlines and for following roads surrounded by forests. In particular, we exploit color and texture properties to differentiate between region types in the aforementioned domains. The performance of our system is evaluated using different experimental approaches, which includes a fully automated in-field flight over a 1 km coastline trajectory.**

I. INTRODUCTION

In this paper we describe the design and evaluation of a system for autonomous vision-based control of an unmanned aerial vehicle (UAV). While various control frameworks exist for UAVs, they typically depend on global positioning system (GPS) data for guidance, and often require constant supervision from a human operator. In this work, we examine a control approach for automated flight based on tracking visual cues on the ground. Our work is motivated by the desire to track and follow the boundaries of environmental features such as coastlines, areas with vegetation, and large animal herds. In the short term, we aim to perform aerial reconnaissance and detect salient geological features that can be used to guide marine robotic vehicles.

This work focuses on boundary tracking tasks in which the regions of interest are visually homogeneous and exhibit features that differentiate them from their surroundings. Some potential airborne reconnaissance and surveillance applications include air-fighting forest fires, confining oil spills, and cataloging the development of geological structures.

The primary solution presented in this work focuses on tracking and following coastlines. This is part of a larger project in which aerial vehicles serve as scouts for underwater robotic systems. Moreover, because the coastline tracker can detect coral reefs, this standalone implementation can potentially be used to assist in the work of biologists studying these endangered marine ecosystems.

In addition to the previous task, one of the most beneficial applications of our framework is to potentially assist firemen in suppressing forest fires. We are motivated to control airtankers and helicopters to douse water along the perimeters of the fire to contain its spread. As a proof of concept, we present a solution to detect and track highways and roads surrounded by forests using texture classification.

This work presents the synthesis of several components of the control system to achieve fully autonomous flight for the vehicle shown in Fig. 1. We however omit discussions on some important aspects of the problem, including vehicle dynamics, flight stabilization, and low-level image processing techniques. While each of these are instrumental to the overall performance of our system, they are based on established methods outside the scope of this paper.

The authors are with the School of Computer Science, McGill University, 3480 University Street, Montréal, QC, Canada H3A 2A7 {anqixu,dudek}@cim.mcgill.ca

The authors gratefully appreciate the financial support of the National Science and Engineering Research Council (NSERC) of Canada.

Fig. 1. The Procerus® Unicorn is a fixed-wing unmanned aerial vehicle with an on-board autopilot microprocessor and gimbal-mounted camera.

Our primary goals are to design a robust and real-time system. To address the speed requirement, we rely predominantly on well-studied existing computer vision techniques known to be fast and reliable. To ensure robustness, we introduce fallback schemes and rejection criteria to either recover from or filter poor intermediate results.

II. RELATED WORK

Despite much work on UAV automation, there has been only limited work to date on the use of appearance-based models for visual guidance. Our work has similarities to vision-based target trackers for miniature quadrotor aircraft [2], for conventional helicopters [14], and for more exotic aerial vehicles [1]. Bourgault *et al.* [4] also investigated automated target tracking and search tasks for fixed-wing and other flying robots, although their emphasis was primarily on the probabilistic modeling of the abstract problem.

Our work draws inspiration from the well-established automated road following literature for terrestrial vehicles, which includes the seminal work by Pomerleau [12] using Artificial Neural Networks and an appearance-based algorithm. Similarly, Ma *et al.* [10] tracked the curve dynamics in noisy images of roads using an Extended Kalman Filter.

Giguère and Dudek [8] developed an unsupervised clustering technique for terrain classification. This work exploits both spatial and temporal continuities to increase the system's accuracy and robustness, which we drew inspiration from in the design of our boundary tracker.

Classifiers used in our system are similar to previous works on color segmentation [17][9]. In a classical survey on texture classifiers, Randen and Husøy [13] compared the

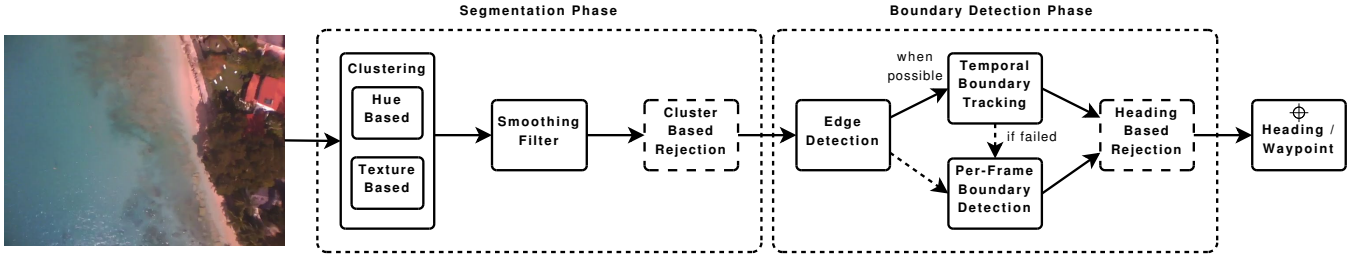


Fig. 2. Block diagram of the proposed boundary tracking framework.

performance of Gabor filter banks, wavelet analysis, and Discrete Cosine Transforms. Although the experimental results illustrated the high discriminating powers of these methods, the presented data also implies that these algorithms are too computationally demanding for our application. Martin *et al.* [11] combined weaker texture classifiers with color cues to build a swift and accurate boundary detection framework.

III. BOUNDARY TRACKING FRAMEWORK

This work processes aerial images to extract boundary information of a region of interest in two phases: first, a segmentation algorithm labels each pixel in the scene as either belonging to the target region or not, and then a boundary line is fit through the connected group of edge elements in the binary labeled image. We can compute a new waypoint or heading for the UAV from one of the extremities of the resulting boundary line segment. The main components of this framework are illustrated in Fig. 2.

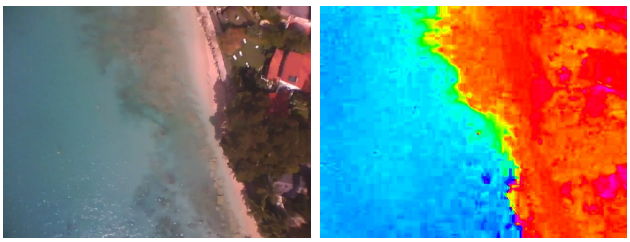
A. Segmentation Phase

We employ conventional cluster analysis to segment images and highlight regions of interest. This operation depends on an image representation well matched to the domain of interest. We use color cues to distinguish water from land and texture traits to differentiate between forests and roads.

1) Hue-Based Clustering for Tracking Coastlines: A natural representation to discriminate water from land within aerial photos of coastline is the dominant color for these two regions. In particular, the hue of a pixel is computed from its Red-Green-Blue (RGB) representation as:

$$C = \max(R,G,B) - \min(R,G,B)$$

$$H = \begin{cases} 0^\circ, & \text{if } C = 0 \\ \frac{G-B}{C} \cdot 60^\circ, & \text{if } \max(R,G,B) = R \\ \frac{B-R}{C} \cdot 60^\circ + 120^\circ, & \text{if } \max(R,G,B) = G \\ \frac{R-G}{C} \cdot 60^\circ + 240^\circ, & \text{if } \max(R,G,B) = B \end{cases} \quad (1)$$



(a) (b)

Fig. 3. Aerial photo of a tropical beach (a) and its hue representation (b).

Although Eq. (1) is equivalent to one of the canonical definitions of hue [15], we explicitly mapped neutral colors ($C = 0$) to red hue ($H = 0^\circ$). This ensures that buildings, rocks and corals are classified together with trees, grass and sand, as seen in Fig. 3. This allows us to easily identify aquatic regions from the rest of the scene.

To determine the label of a pixel $L_{x,y}$, we compare its color $H_{x,y}$ to the hue of different classes H_{water} and H_{land}:

$$\Delta H_{water} = \left|(H_{x,y} - H_{water} + 180^\circ) \, mod \; 360^\circ - 180^\circ\right|$$
$$\Delta H_{land} = \left|(H_{x,y} - H_{land} + 180^\circ) \, mod \; 360^\circ - 180^\circ\right|$$

$$L_{x,y} = \begin{cases} \text{'water'}, & \Delta H_{water} \leq \Delta H_{land} \\ \text{'land'}, & \text{otherwise} \end{cases}$$

We compute H_{water} and H_{land} by manually choosing initial hues and then applying the K-means algorithm [6] on training footage collected in the field.

2) Texture-Based Clustering for Tracking Forest Paths: We present a fast and accurate method for differentiating coarse tree texture from smooth road texture using a representation based in the Speeded-Up Robust Features (SURF) descriptor [3]. This descriptor summarizes directional gradient information within the neighborhood of a target pixel as a single numerical vector. Although the original SURF algorithm only computes these vectors at locations pertaining to salient features, we apply this technique to describe local regions (at a fixed scale) of all uniformly sampled image coordinates. In contrast to the original design, we also omit the vector normalization step, thus preserving the distinction between strong and weak gradients corresponding to coarse and smooth textures.

Each SURF vector can be computed efficiently using a constant number of integer operations regardless of the scale. Our system only computes these vectors for sub-sampled locations within the image, since we do not typically require pixel-level accuracy. In addition, we chose the scale the SURF descriptor empirically using our collected footage.

We apply the K-means algorithm to map SURF vectors to a binary label indicating whether the corresponding location contains trees or roads. Although in general we cannot predict the final locations of the two clusters in vector space, we assume that our aerial footage contains primarily these two terrain types. Under this assumption, Fig. 4 shows that the K-means algorithm is capable of differentiating trees from roads, where the tree label is attributed to the centroid with the larger magnitude corresponding to the coarser texture.

3) Post-Clustering Refinement: Both our manual cluster analysis and the K-means algorithm assign labels to each pixel independently from its neighbors, and thus the clustered image often contains small patches of incorrectly labeled pixels, as illustrated by Fig. 4(b). We apply a smoothness constraint to minimize the sizes of these erroneous regions, which we implement using a median filter [16].

The success of the segmentation phase depends on certain domain-specific assumptions being met. For the coastline tracker, each image should ideally contain approximately the same amount of water and land. But if the plane is suddenly pushed away from the coastline (e.g. due to strong lateral winds), then the resulting frames might contain insufficient coastline information. We define the rejection criterion $R_{cluster}$ by imposing minimum and maximum bounds σ and Σ on the relative size of the target region:

$$R_{cluster} = \begin{cases} 0, & \sigma \leq \frac{p}{wh} \leq \Sigma \\ 1, & \text{otherwise} \end{cases}$$

where p is the number of pixels labeled as the target region and w and h are the width and height of the image. Frames for which $R_{cluster} = 1$ are dropped immediately.

(a) (b)

Fig. 4. Aerial photo of a road surrounded by trees (a) and its labeled representation using the SURF descriptor and the K-means algorithm (b).

B. Contour Detection Phase

The second phase of our framework involves fitting a straight line through the binary image's boundary, which can be then transformed into a desired heading for the UAV.

For simplicity, our framework currently assumes that only a single hypothesis exists; that is to say, we assume only one continuous border between water and land or between forest and road. Despite using this idealized assumption, we will demonstrate in Section IV that our framework produces accurate results for realistic setups, even in the road tracker setting where forest-road boundaries come in pairs.

1) Edge Detection: The edge pixels (or edgels) of the clustered image can be computed using any classical edge detection method. We use the Sobel operator [5] to generate connected edgel sets from our binary images.

2) Temporal Boundary Tracking: Given the availability of a recent line fit, we take advantage of temporal continuity by assuming that the boundary in the current frame corresponds to the connected group of edgels closest to this previous line. Since the plane is travelling at a high altitude, the contents in consecutive frames do not change drastically and thus allow us to make this temporal association. Our linear regression implementation rotates the coordinate axes about the principal component and then applies linear least squares.

3) Per-Frame Boundary Detection: The accuracy of our temporal boundary tracker depends on having a previous line fit from a relatively recent frame. Notwithstanding the very first iteration where no previous fit is available, our cluster-based rejection criterion $R_{cluster}$ can also potentially reject a large sequence of frames. In these cases where temporal continuity cannot be guaranteed, we use the RANSAC algorithm [7] to fit a straight line through the largest number of edgels within a reasonably close distance.

4) Post-Fit Refinement: After obtaining the line fit, we compute its two intersections with respect to the image borders. Since we choose not to make assumptions about the structure of the boundary beyond the current frame, we use the intersection closest to the front of the plane to determine a new heading direction for the UAV.

Despite the presence of our failsafe mechanisms, there are cases where the computed heading may be erroneous and, in particular, may direct the plane in the opposite direction of its bearing. To prevent the UAV from back-tracking through previously covered terrain, the rejection criterion $R_{heading}$ imposes an upper bound Φ on the distance between the current heading ϕ_{t-1} and the newly computed direction ϕ_t:

$$R_{heading} = \begin{cases} 1, & |(\phi_t - \phi_{t-1} + 180^\circ) \, mod \; 360^\circ - 180^\circ| > \Phi \\ 0, & \text{otherwise} \end{cases}$$

IV. EMPIRICAL VALIDATION

We conducted three sets of experiments to assess the performance of the proposed boundary tracking framework. The first two trials evaluate our coastline-following implementation, where we deployed our UAV to fly along the shores of a tropical island. For the third trial, we conducted a preliminary assessment of the road tracking implementation using publicly available aerial photos.

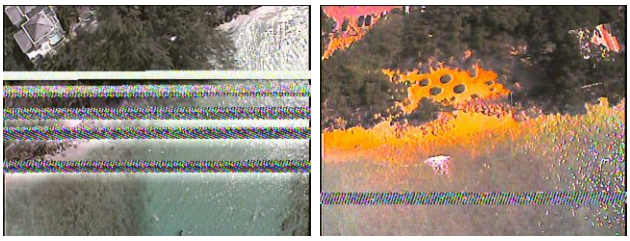
A. Hardware and Software Setup

Our unmanned aerial vehicle is a rigid body fixed-wing plane commercially available from Procerus® Technologies. The UAV's 1 meter wingspan is built using expanded polypropylene (EPP) foam, which allows it to bounce upon landing. An electric motor powered by a pair of 3-cell lithium polymer batteries can drive the plane at average ground speeds of 14*m/s* and for durations up to 30 minutes.

The plane's moment-to-moment heading and flight characteristics are regulated using a micro-processor unit. This autopilot is connected to numerous sensors, including a 3-axis accelerometer, a pressure sensor, a magnetometer, and a GPS unit. Communication between the autopilot and the ground control software is achieved via radio frequency. This UAV can operate in many modes, ranging from purely manual control to fully autonomous waypoint-based navigation.

An on-board camera transmits live analog video stream at 30 *fps* via a separate radio frequency. This camera is attached to a gimbal which can be controlled either by a human operator or via software interfacing with the ground control application. Because the transmission is sensitive to environmental disturbances, some of the received frames contain significant image distortions, as illustrated by Fig. 5.

Although we successfully ran our tracker on this noisy stream, we also manually attached a separate video recorder on the underbelly of the UAV to capture noise-free video for use as training data.



(a) (b)

Fig. 5. Some of the frames from the transmitted video stream contain noise such as scan line artifacts (a) and color distortions (b).

We implemented our tracking framework in C++ and within a Linux environment on-board a 1.66*GHz* dual core notebook computer. The analog video stream transmitted from the UAV is acquired by a USB video capture device and is processed in real-time at 10 *Hz*.

We continuously regulate the gimbal orientation so that it is always perpendicular to the ground plane. This removes the need to transform frames using projective geometry.

B. Coastline Tracker Evaluation Criterion

We collected video footage from our manually mounted recorder spanning three flight sequences along the same shoreline. Nearly 6000 frames were extracted from the H.264-encoded video files at a resolution of 640×480 pixels at 30 *fps*; these were used for off-line performance evaluation. Although the video quality is much cleaner compared to the on-board gimbal-mounted camera, these frames are not always parallel to the ground plane, and thus some might contain too much land or too much water. Thankfully, our system can filter out these bad frames using the $R_{cluster}$ rejection criterion.

We compute the absolute angular distance between the headings generated by our tracker ϕ_t and by the i-th volunteer ϕ_t^i. We summarize our results as average and worst-case values for errors $\Delta\phi_t$ per frame t averaged over all human datasets, and we compare these errors with the average pairwise discrepancy $\Delta\phi_t^*$ among humans:

$$\Delta\phi_t = \frac{1}{5}\sum_{i=1}^{5} \left| \left(\phi_t - \phi_t^i + 180^\circ\right) mod\ 360^\circ - 180^\circ \right|$$

$$\Delta\phi_t^* = \frac{1}{10}\sum_{i=1}^{5}\sum_{j=i+1}^{5} \left| \left(\phi_t^i - \phi_t^j + 180^\circ\right) mod\ 360^\circ - 180^\circ \right|$$

C. Coastline Tracker Experimental Results

Fig. 6 shows the average discrepancy among humans for selecting UAV headings. Since our footage consists of three flights over the same coastline, we observe that the volunteers disagreed consistently during the middle portion of each flight, which correspond to the large coral reefs seen in Fig. 11. This illustrates the difficulty of the problem that we are trying to solve – if humans strongly disagree with each other on the location of the shoreline in these frames, then it is ambiguous to even define a baseline to compare against our system's performance. Despite this, we assume that the average discrepancy among our human data provides a reasonable reference of optimal accuracy.

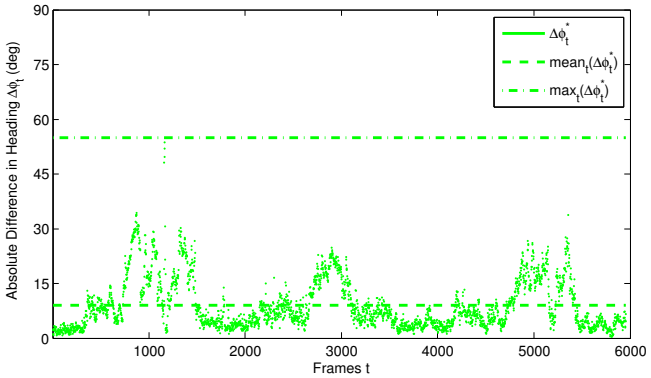


Fig. 6. The average discrepancy in terms of heading among human datasets suggest that in certain scenes (e.g. containing coral reefs) even humans strongly disagree with each other on the location of the coastline.

We analyzed the effects of the lower bound σ on the required amount of water in each frame, without restricting other parameters (i.e. $\Sigma = 1$ and $\Phi = 180^\circ$). Fig. 7 shows that although we can increase accuracy by tuning σ, the number of rejected frames becomes too large before we can observe significant improvement in the worst-case error.

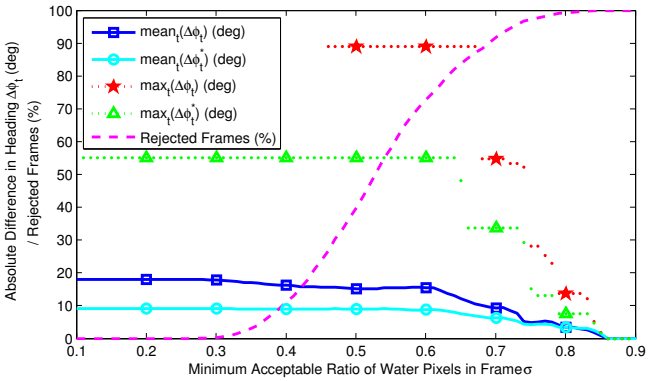


Fig. 7. By rejecting frames with too little water in them, the performance of our system improves slowly compared to the number of frames rejected.

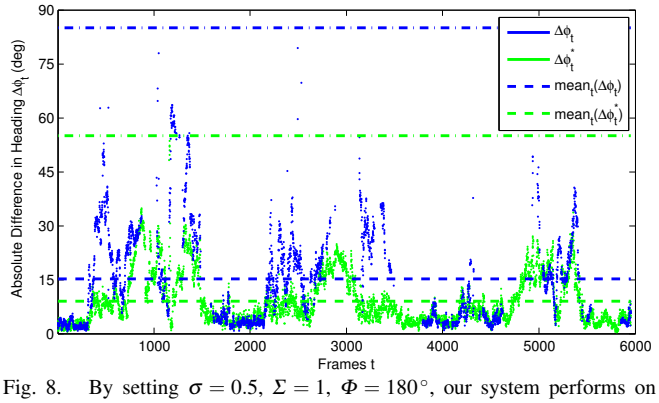


Fig. 8. By setting $\sigma = 0.5$, $\Sigma = 1$, $\Phi = 180^\circ$, our system performs on average less than two times as poor as the average inter-human discrepancy. Unfortunately, this configuration dropped too many consecutive frames, including regions with low inter-human discrepancy.

Fig. 8 illustrates our system's average performance with $\sigma = 0.5$. Because our implementation processes video at

10 Hz, it can tolerate errors spanning several frames so long as the total duration is limited. By dismissing these isolated errors, the average performance is quite comparable to the human discrepancy rate. Unfortunately, the system also rejected large number of consecutive frames (e.g. $t \in [3500, 3800]$ and $t \in [5550, 5850]$), meaning that the UAV would not be regulated for up to 9 seconds intervals. We thus conclude that increase σ alone is too dangerous because it might drop too many consecutive frames.

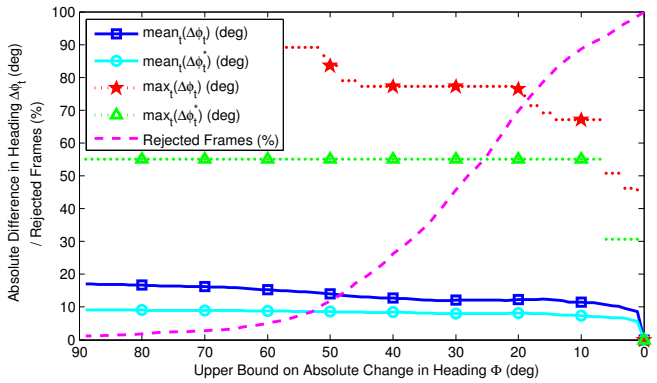


Fig. 9. By rejecting frames where the desired heading is significantly different from the current heading, we can significantly improve both the average and worst-case performance while preserving most of the frames.

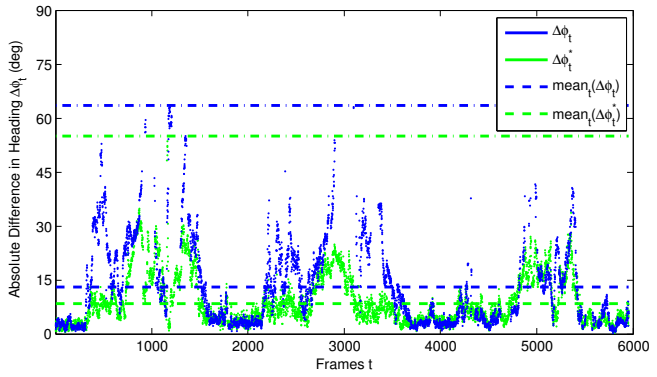


Fig. 10. By setting $\sigma = 0$, $\Sigma = 1$, $\Phi = 45°$, our system overall performs comparably to the average inter-human discrepancies. In addition, this configuration dropped consecutive frames of at most 4 seconds, and during times where humans strongly disagreed with each other.

We also investigated the change in performance by restricting the maximum deviation Φ between the current UAV heading and the waypoint generated by our system, without using the cluster-based rejection criteria (i.e. $\sigma = 0$ and $\Sigma = 1$). This experiment indicates that we do not need to reject as many frames as previously to obtain decent results, as illustrated in Fig. 9. By imposing a maximum absolute tolerance of $\Phi = 45°$, Fig. 10 shows that the system correctly ignores scenes with large associated inter-human discrepancies. More importantly, the longest consecutive frame drop was only about 4 seconds long ($t \in [2900, 3100]$), and furthermore almost none of the frames containing non-ambiguous boundaries were rejected.

In conclusion, we observe that the heading-based rejection threshold Φ is more effective at filtering out ambiguous scenes than the cluster-based threshold σ. The final results

TABLE I

AVERAGE AND WORST-CASE ABSOLUTE DISTANCE IN GENERATED HEADING BETWEEN OUR ALGORITHM AND TWO HUMANS

	Average Value	Worst-Case Value
Error with Respect to Subject 1 $\Delta\phi_t^1$	4.37°	11.67°
Error with Respect to Subject 2 $\Delta\phi_t^2$	4.50°	10.12°
Discrepancy Between Subjects $\Delta\phi_t^*$	0.81°	3.14°

in Fig. 10 illustrate that our system performs quite accurately on average (with a mean error of less than 15°). In addition, among frames where the coastline is potentially ambiguous (i.e. where the inter-human discrepancy is above its mean $\Delta\phi_t^* > mean_t(\Delta\phi_t^*)$), our system performs only 39% more worse than humans.

D. Coastline Tracker Field Trial

For our trial, we imposed lenient parameters for all three rejection thresholds ($\sigma = 0.35$, $\Sigma = 0.8$, $\Phi = 135°$) to assess the tracker's performance in the worst-case scenarios. After manually aligning the UAV with the coastline, our algorithm processed the transmitted stream from the on-board gimbal-mounted camera and steered the plane comfortably in real-time (at 10 Hz) along a 1 km stretch of the shore made up of various terrain types. The vehicle traveled at a nominal altitude of 150m with an average ground speed of 14m/s with 7m/s lateral wind speed. The trajectory is shown in Fig. 11.

This trial demonstrates that our framework is capable of tracking the shoreline under real flight conditions. Despite the significant amount of imaging and transmission noise present (e.g. Fig. 5), our overall performance was still robust and could likely be further improved with additional tuning of the parameters.

E. Forest Path Tracker Experiment

To gauge the performance of the texture-based classifier for differentiating roads from its tree surrounding, we ran our algorithm through 15 manually selected aerial photos from a publicly available image database. We asked two volunteers to define pairs of waypoints in each image corresponding to headings along the center of the roads. Since our classifier identifies the side of the road rather than its center, our results should be interpreted qualitatively due to this discrepancy.

Fig. 12. Despite some noticeable mis-classified regions, our framework is able to provide a linear estimate of the location for the side of the road.

Fig. 11. Our tracking algorithm successfully drove an UAV along a 1km tropical coastline. The bright curve denotes the trajectory of the vehicle. Please refer to our accompanied video submission for a playback of this flight (at 2x speed).

The results of our system with all rejection criteria disabled are shown in Table I. Despite the fundamental discrepancy between the headings produced by our algorithm and those specified by humans, our framework achieved both decent average and worst-case performance. Although additional experiments are required to be conclusive, we are very pleased with these preliminary results (e.g. Fig. 12).

V. CONCLUSIONS AND FUTURE WORK

In this paper we presented a fully autonomous vision-based control framework for steering a small aircraft along the boundary of various terrains or regions of interest. Our design approach involved using well-established vision algorithms in addition to fallback mechanisms and rejection criteria to ensure that the system performs accurately.

We used hue to distinguish water from land in our coastline tracking experiments, and employed a novel texture-based representation to differentiate between forest and ground cover in our road tracking assessment. In addition to our off-line experiments, we successfully flew our UAV along a 1 km tropical coastline solely using our algorithm.

We are currently investigating generalizations to our classifier for both the coastline tracking and road tracking applications. In addition, we are interested in introducing limited human interaction prior to execution to improve the overall performance of our tracking. For example, if the operator generates either a general heading or a sample trajectory prior to launch, we can use this information to fine-tune our rejection criteria and minimize premature rejections. Finally, we are in the preliminary stages of extending our aerial tracker to perform coverage, surveillance, and collaboration with underwater robotic vehicles.

REFERENCES

[1] J.R. Azinheira, Patrick Rives, J.R.H. Carvalho, Geraldo F. Silveira, Ely C. de Paiva, and Sameul S. Bueno. Visual servo control for the hovering of an outdoor robotic airship. volume 3, pages 2787–2792, 2002.

[2] Abraham Bachrach, Alborz Geramifard, Daniel Gurdan, Ruijie He, Sam Prentice, Jan Stumpf, and Nicholas Roy. Co-ordinated tracking and planning using air and ground vehicles. In *The 11th International Symposium on Experimental Robotics*, pages 137–146, Greece, 2008.

[3] Herbert Bay, Andreas Ess, Tinne Tuytelaars, and Luc Van Gool. Speeded-up robust features (surf). *Computer Vision and Image Understanding*, 110(3):346–359, 2008.

[4] Frédéric Bourgault, Tomonari Furukawa, and Hugh F. Durrant-Whyte. Optimal search for a lost target in a bayesian world. In Shin'ichi Yuta, Hajime Asama, Sebastian Thrun, Erwin Prassler, and Takashi Tsubouchi, editors, *FSR*, volume 24 of *Springer Tracts in Advanced Robotics*, pages 209–222. Springer, 2003.

[5] Richard O. Duda and Peter E. Hart. *Pattern Classification and Scene Analysis*. John Wiley & Sons Inc, 1973.

[6] Richard O. Duda, Peter E. Hart, and David G. Stork. *Pattern Classification (2nd Edition)*. Wiley-Interscience, 2 edition, 2000.

[7] Martin A. Fischler and Robert C. Bolles. Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography. *Comm. ACM*, 24(6):381–395, 1981.

[8] Philippe Giguere and Gregory Dudek. Clustering sensor data for terrain identification using a windowless algorithm. In *Proceedings of Robotics: Science and Systems IV*, Zurich, Switzerland, June 2008.

[9] Patrick Lambert and Thierry Carron. Symbolic fusion of luminance-hue-chroma features for region segmentation. *PR*, 32(11):1857–1872, November 1999.

[10] Yi Ma, Jana Koscká, and Shankar S. Sastry. Vision guided navigation for a nonholonomic mobile robot. *IEEE Transactions on Robotics and Automation*, 15(3):521–536, 1999.

[11] David R. Martin, Charles C. Fowlkes, and Jitendra Malik. Learning to detect natural image boundaries using local brightness, color, and texture cues. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26(5):530–549, 2004.

[12] Dean A. Pomerleau. —efficient training of artificial neural networks for autonomous navigation. *Neural Computing*, 3(1):88–97, 1991.

[13] Trygve Randen and John Håkon Husøy. Filtering for texture classification: A comparative study. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 21:291–310, 1999.

[14] Srikanth Saripalli, James F. Montgomery, and Gaurav S. Sukhatme. Visually guided landing of an unmanned aerial vehicle. *IEEE Transactions on Robotics and Automation*, 19(3):371–380, 2003.

[15] A. R. Smith. Color gamut transform pairs. In *Proceedings of the 5th annual conference on Computer graphics and interactive techniques (SIGGRAPH'78)*, pages 12–19, New York, NY, USA, 1978. ACM.

[16] Emanuele Trucco and Alessandro Verri. *Introductory Techniques for 3-D Computer Vision*. Prentice Hall PTR, Upper Saddle River, NJ, USA, 1998.

[17] Chi Zhang and P. Wang. A new method of color image segmentation based on intensity and hue clustering. volume 3, page 3617, Los Alamitos, CA, USA, 2000.