Landing a UAV on a Runway Using Image Registration

Andrew Miller and Mubarak Shah and Don Harper
University of Central Florida
4000 Central Florida Blvd, Orlando FL, 32816
{amiller,shah,harper}@cs.ucf.edu

***Abstract*— In this paper we present a system that uses only vision to land a UAV on a runway. We describe a method for estimating the relative location of the runway as an image by performing image registration against a stack of images in which the location of the runway is known. An approximation of the camera projection model for a forward-facing view of a runway is derived, allowing the course deviation of the UAV to be estimated from a registered image. The course deviation is used as input to a linear feedback control loop to maintain the correct flight path. Our method is implemented as a real-time multithreaded application, which is used to control an aircraft in Microsoft Flight Simulator. We also show results of applying the vision component of the system to video recorded from an actual UAV.**

I. INTRODUCTION

Unmanned Aerial Vehicles (UAVs) traditionally rely on a GPS receiver for navigation, and solid state inertial sensors for control. Color video cameras are a tempting alternative sensor choice because they are light, inexpensive, and easy to use. Cameras are most often used on UAVs for high-level functions, such as target recognition or surveillance. If the low-level functions could also be performed by the camera, then the number of needed sensors could be reduced, especially in a small or micro UAV where component weight is tightly constrained.

For a successful landing approach, the UAV should descend at a constant angle (conventionally 3 degrees below the horizon), point in the direction of the runway centerline, and touch down with the wings level at the beginning of the runway. Our proposed method is to steer the UAV towards the runway and maintain these constraints by using visual information about the horizon and the relative position of the runway as observation inputs to a linear feedback controller.

The unique contributions of this paper are a simplification of the camera projection model that allows linear estimates of the UAV's position to be obtained by measuring image geometry, and a method for estimating the position of the UAV relative to the runway by performing image registration against a stack of reference images with a known runway location.

This is a suitable problem for vision because the geometry of the horizon and runway in the image map directly to world coordinates. Human pilots navigating by Visual Flight Rules have been landing aircraft on runway since the beginning of aviation using only visual cues. Our method for finding the runway is based on performing image registration against a set of prerecorded images from different points along the glide path in which the location of the runway is known.

Our method differs from other approaches that recognize a runway based on visual features of the runway itself. Our method uses information about the terrain surrounding the runway at different scales and distances, so it can steer the UAV towards the runway even before the runway itself becomes visible.

Our approach also does not make use of physical flight dynamics models or intrinsic camera calibration. Instead we obtain geometric features that are approximate linear indicators of the physical quantities we want to measure, i.e. angular deviation from the ideal glide path. The linear controllers have gains that are tuned to simultaneously compensate for the parameters of the camera and the flight characteristics of the UAV.

II. PREVIOUS WORK

There have been several projects involving the autonomous landing of a UAV. Most involve a downward facing camera, which is appropriate for landing a helicopter [1], [2] or a balloon [3]. The orientation of the camera is important in these examples because the ground plane is nearly parallel to the image plane, so there is a minimal effect of perspective distortion. Other projects involving the use forward looking cameras on a UAV use a reduced amount of information, such as the controlling the UAV based on the orientation of the horizon [4], flying towards features that can be detected using with linear classifiers [5], or avoiding obstacles by steering away from areas of high optical flow [6]. The cameras on UAVs are traditionally used for higher-level tasks such as tracking target objects in real-time [7], or determining a safe place to land in an unfamiliar location [8].

A vision system for landing a UAV is essentially a specific application of the well-known pose estimation problem, where the goal is to determine the location and orientation of the camera given test images [9], although these methods typically require a larger number of points to be found with known GPS locations. There has also been research into the specific problem of a camera view of perspective-distorted planes, such as detecting the orientation [10] or the vanishing line [11] using texture information.

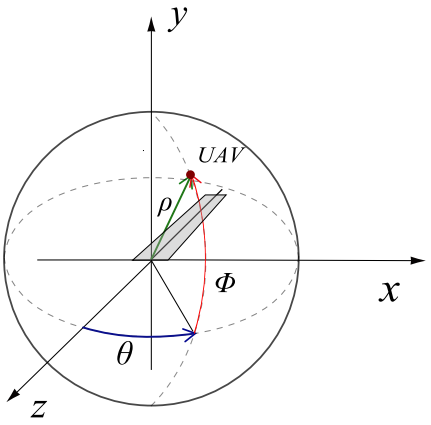


Fig. 1. Coordinate system describing the runway and the UAV. The runway begins at the origin and extends in the negative Z direction. The UAV position is described with spherical coordinates.

III. PROJECTIVE GEOMETRY OF A RUNWAY

In this section we show how linear estimates for the position and orientation can be obtained using geometric features of a video frame. The simplifying assumption that we make is that the UAV only deviates from the desired glide path by small amounts. This allows us to use small-angle approximations ($\sin x \approx x$, $\cos x \approx 1$) to reduce the camera projection matrix.

The runway is described as a line on the ground plane on Z axis. The beginning of the runway, where the UAV should land, is at the origin. We describe the position of the UAV with spherical coordinates (ρ, ϕ, θ). This coordinate system is illustrated in Figure 1. The orientation of the UAV can be described with *yaw* (or *heading*), *pitch*, and *roll* angles: α, β, and γ.

We assume the UAV is equipped with a forward-pointing pinhole camera that has square pixels, a principle point in the center of the image, and focal length f. Since small changes in orientation of a camera cause only rigid transformations of the image, i.e. distances and angles in the image are unchanged, we assume the wings of the UAV are level, i.e. $\beta = 0$ and $\gamma = 0$.

The projection of any point on to image coordinates is given by the following equation:

$$\begin{bmatrix} wx & wy & w \end{bmatrix}^T = \mathbf{CR} \begin{bmatrix} X & Y & Z & 1 \end{bmatrix}^T \tag{1}$$

where $\mathbf{C}$ is the internal calibration matrix,

$$\mathbf{C} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & \frac{1}{f} \end{bmatrix} \tag{2}$$

and $\mathbf{R}$ is the euclidean rotation and translation of the world relative to the camera.

$$\mathbf{R} = \begin{bmatrix} \cos\alpha & 0 & \sin\alpha \\ 0 & 1 & 0 \\ -\sin\alpha & 0 & \cos\alpha \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & -\rho\cos\phi\sin\theta \\ 0 & 1 & 0 & -\rho\sin\phi \\ 0 & 0 & 1 & -\rho\cos\phi\cos\theta \end{bmatrix} \tag{3}$$

After applying the small-angle approximations to all of the sin and cos terms, we arrive at a simplified form of the projection matrix:

$$\mathbf{CR} \approx \begin{bmatrix} 1 & 0 & \alpha & -\rho\theta - \rho\alpha \\ 0 & 1 & 0 & -\rho\phi \\ \frac{-\alpha}{f} & 0 & \frac{1}{f} & \frac{-\rho}{f} \end{bmatrix} \tag{4}$$

We are now interested in applying this projection to the origin, i.e. the beginning of the runway, and to the point at infinity on the runway line:

$$\mathbf{CR} \begin{bmatrix} 0 & 0 & 1 & 0 \end{bmatrix}^T \propto f \begin{bmatrix} \alpha & 0 & \frac{1}{f} \end{bmatrix}^T \tag{5}$$

$$\mathbf{CR} \begin{bmatrix} 0 & 0 & 0 & 1 \end{bmatrix}^T \propto f \begin{bmatrix} \theta + \alpha & \phi & \frac{1}{f} \end{bmatrix}^T \tag{6}$$

The y-coordinate of the runway's vanishing point is zero, meaning it always falls on the horizon. The x-coordinate of the vanishing point is proportional to the heading, α. The y-coordinate of the point at the beginning of the runway is proportional to the elevation angle of the UAV. These quantities are illustrated in Figure 6. The intuitive relationship between the measurements and the relative glide path deviation of the UAV is shown in Figure 3.

IV. OVERVIEW OF PROPOSED SYSTEM

Our method is based on using computer vision to determine the orientation of the UAV and its relative position to the ideal path leading to the runway. The UAV can then be steered to maintain the ideal path using a cascading control approach.

Our method is summarized by the following steps:

- Estimate the current position of the runway in the video
- Determine the relative position of the UAV to the ideal glideslope, i.e. if the UAV is too high or too far left
- Determine the ideal attitude of the UAV to steer it back on the glideslope
- Estimate the current attitude by detecting the orientation of the horizon
- Apply a force to the control surfaces of the UAV to attain the ideal attitude

V. RUNWAY ESTIMATION

A. Preparation of Reference Frames

Our method for locating the runway involves registration against a prior image in which the location and orientation of the runway is already known. The questions to answer are: 1) what kind of image should be used as a reference, and 2) how should the runway be annotated.

Since the reference images will be used for image registration, they should be as similar to the incoming video frames as possible. Thus the best source for reference images is a video from a previous landing of the UAV flown manually or by another autopilot system. It would be more convenient to use satellite or aerial imagery, from Google Earth, for example, except that registration would be much more difficult (especially in real-time) because the reference

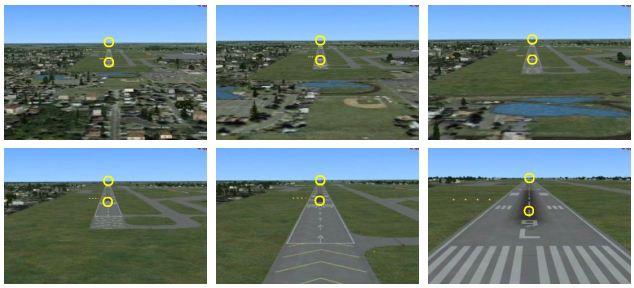
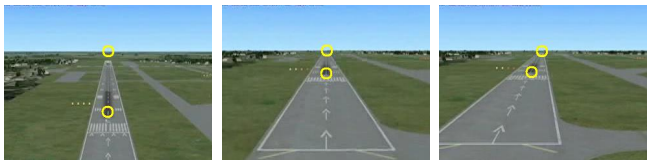


Fig. 2. Reference frames taken from a video as the UAV gets closer to the runway. The reference frames are sampled from the video at increasing frequency as the altitude decreases, since the ground appears to move faster. The vanishing point and a point at the beginning of the runway (the spot where the UAV should touch down) are annotated in each frame.



a b c

Fig. 3. In the middle frame the UAV is on the correct glide path; the runway is perpendicular to the horizon, and the beginning of the runway is approximately 3 degrees below the horizon. In the left frame, the UAV is too high, and the beginning of the runway is further from the horizon. In the right frame, the UAV is too far to the right, and the runway makes an acute angle with the horizon.

images would differ greatly from the incoming video in terms of scale, noise, perspective skew of feature points, etc.

The view from the UAV is very different when the UAV is far away from the runway and high in the air than when it is low to the ground and near the runway, so it is best to store a rough sampling of keyframes from the reference video with some overlap. Since the ground also appears to move much faster when the UAV is lower, the keyframes for lower altitudes should be closer together than keyframes for higher altitudes. In our experiments, we used a quadratic sampling strategy, i.e. the 2nd closest frame, 4th closest frame, 9th closest frame, and so on, for a total of 100 reference images. Some of these images are showin in Fig 2.

We describe the position and orientation of the runway with two points: a point at the beginning of the runway (the spot where the UAV should touch down), and the vanishing point on the horizon made by the centerline of the runway. There are several ways to obtain these points. The most straightforward method is to manually annotate each reference image by clicking the points, which is tedious and somewhat imprecise. Another way is to annotate a single reference frame and use image registration to estimate the location in all of the other frames. As stated above, it is difficult to register all of the frames in the video to a single frame because the view varies substantially at different altitudes. However, offline training is much less restrictive than the real-time algorithm, so robust but time-consuming registration techniques can be used [12].

B. Finding the Homography

From a high altitude, and ignoring effects of the spherical lens distortion or the curvature of the Earth, the visible features below the horizon lie on a ground plane. Any point on the ground in one view can be related to the corresponding point in a different view by a $3x3$ projection matrix, H. Given a reference frame with the two annotated runway points, we can use this relationship to locate these points in a new frame. The projection matrix can computed in several ways, the most convenient of which is a combination of SIFT feature point correspondence [13] and RANSAC [14] homography fitting. An intuitive summary of this method is provided below.

SIFT produces a set of distinctive feature points from an image, typically between 100 and 1000 in our experiments. SIFT also produces a 128-dimensional feature vector for each of these points, which can be used to match corresponding points between a reference frame and a test frame. The feature vector is essentially a histogram of gradient vectors from the patch centered around the feature point, and is rotation invariant and robust to small affine distortion. Usually this method finds between 20 and 200 *strong* correspondences. Any four point correspondences (except in a degenerate case when three of the points are colinear) determine a homography matrix, and a larger number of correspondences can be used through least-squares-regression. If all of the correspondences were correct, and the feature point locations were very precise, then any four points could be chosen, and the resulting homography could transform the coordinates of every point in one image to the exact coordinates of the corresponding point in the other image. In practice, though, several of the correspondences are incorrect, and the coordinates of some of the correctly corresponded points are imprecise. The RANSAC approach is used to select at random a small group of matched feature points, and compute the homography matrix as a hypothesis. The hypothesized homography is then tested by applying the homography matrix to the rest of the feature points, and see how many of the projected feature points end up near the corresponding point in the other image. If a high percentage of these points are inliers, then the hypothesis is accepted; otherwise, another small group of points is selected.

The more similar two images are, i.e. the smaller the change in viewpoint between them, the more SIFT points will be matched. If we performed the matching process for a given test frame against all 100 reference frames, we could obtain a graph similar to the one shown in Fig 5. The reference frame with the highest number of matching points is the most similar to the test frame, and will produce the most accurate registration. The index of the best matching frame can also be used as a rough gauge of altitude and distance to the runway.

As the UAV approaches the runway, the best-index should periodically increase, one frame at a time. Thus it is not necessary to match against all of the reference frames if the best-index is known for the previous frame. Instead, a counter

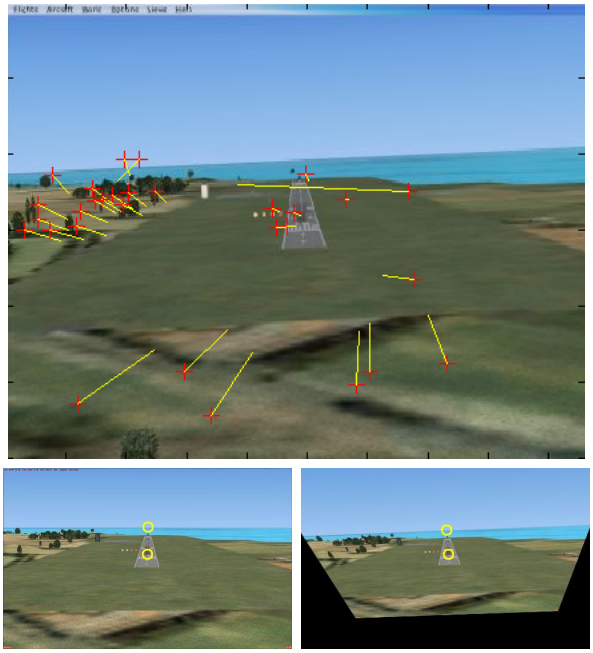


Fig. 4. The result of performing registration between a test frame *(top)* and a reference frame *(left)* is a matrix that can map every point in the reference image to a corresponding coordinate in the landing image, as shown by the warped image *(right)*. The matched feature points are shown as line segments from the coordinate in the test frame to the coordinate in the reference image.

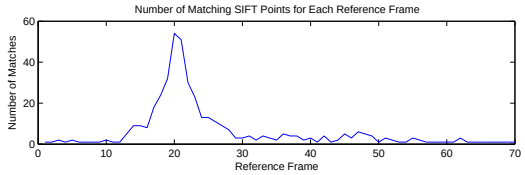


Fig. 5. The number of matched SIFT feature points between each reference frame and a test frame is an indication of the similarity between the viewpoints in the two image. Thus the reference frame with the highest number of points will produce the most accurate estimate of the runway orientation. The index of the best matching frame can also be used as a rough gauge of altitude and distance to the runway.

is used to represent the previous best-index, and each test frame is only compared to the previous-best reference frame and to its successor. If the successor has a higher number of matches than the previous, then the counter is incremented.

C. Measuring the Image Geometry

We want to measure three geometric properties, the runway offset, the runway angle, and the runway distance. A visual interpretation of these values is shown in Fig 6.

In the current reference frame, the annotated runway beginning point and vanishing point are given by T and V:

$$T = \begin{bmatrix} x_T \\ y_T \\ 1 \end{bmatrix}, \qquad V = \begin{bmatrix} x_V \\ y_V \\ 1 \end{bmatrix} \tag{7}$$

The first step is to project the coordinates from the reference frame into the current test frame, by applying the homography H. The second step is to transform the frame

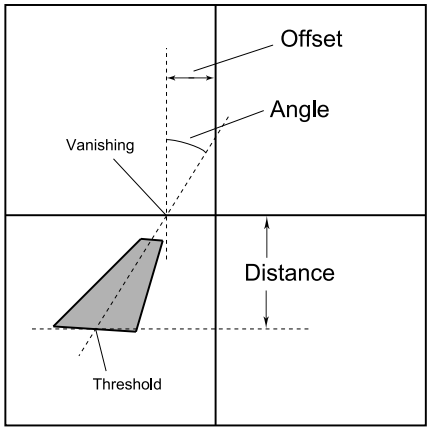


Fig. 6. Geometric interpretation of the three measured quantities (Offset, Distance, and Angle) based on the beginning and vanishing points of the runway image. The coordinate space has already been rotated and translated so that the horizon lies on the X axis.

into a canonical representation that is invariant to the bank and pitch angle of the UAV by applying the matrix M_H to any homogeneous coordinate in the frame, where θ_H is the *horizon angle* and D_H is the *horizon distance*, which is the perpendicular distance from the horizon line to the center of the image, measured in pixels. The horizon line is detected using a quick iterative method described in [4].

$$M_H = \begin{bmatrix} \cos(\theta_H) & \sin(\theta_H) & 0 \\ \sin(\theta_H) & -\cos(\theta_H) & -D_H \\ 0 & 0 & 1 \end{bmatrix} \tag{8}$$

$$T' = M_H \cdot H \cdot T, \qquad V' = M_H \cdot H \cdot V \tag{9}$$

The transformed vanishing point, V', will now lie on the X axis. The offset is the horizontal distance in pixels from the center of the image to V'.

D. Controlling the UAV

There are five observations that we want to maintain for a stable landing approach. We accomplish this by using five cascading closed-loop PI (Proportional-Integral) controllers in the arrangement shown by Fig 7. The control system uses only two control surfaces on the UAV - the ailerons and elevator; the other controls - the throttle and rudder - are kept constant.

A PI controller has three signals: (1) feedback, or an observation of a physical quantity we want to control, e.g. the heading of the UAV; (2) a setpoint, or the target value we want the observation to equal; and (3) an output that lets the controller influence the physical system, e.g. the ailerons of the UAV. The intuition behind the PI controller can be summarized by the following two rules: (Proportional) If the UAV is too far left, then it should turn right. (Integral) If it is still too far left after some time has passed, then it should turn right harder.

In a cascaded arrangement of controllers, the output of one PI loop is used as the setpoint for the following controller. For example, if the UAV is too far to the left of the runway,

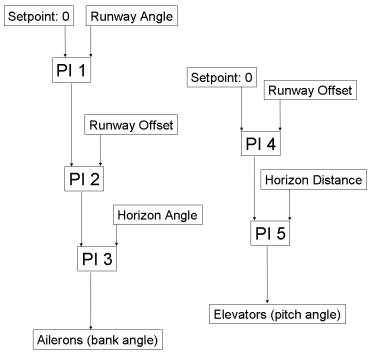


Fig. 7. The UAV is controlled by two independent cascaded PI control loops. The lower loops controlling the horizon are updated at 20Hz, while the higher loops controlling the runway orientation are updated at 1Hz

the first PI controller will set the desired heading to the right. The next control loop will set the desired horizon angle to bank the UAV to the right, and the final loop will apply forces to the ailerons to attain the desired bank angle.

We are not calculating the actual altitude and horizontal position of the UAV, but instead we obtain measurements that are linearly or near-linearly related to these physical quantities. Put another way, the uncertainty about the actual location and orientation of the UAV, arising from the unknown sensor parameters, is addressed by tuning the gains and setpoints of the control loops. In our simulation, the gains were tuned manually, although a more sophisticated method could be used if a flight model is available.

We added a heuristic to our implementation that allows the UAV to reduce the throttle to idle and tilt back (a flare maneuver) when it is very close to the runway, i.e. when the last reference frame produces the highest number of matching feature points. At this time, the measurements from the runway estimator are ignored, and the UAV simply levels its wings by keeping the horizon even.

VI. RESULTS

Our system was implemented as a multithreaded C++ program running on a 2.33GHz Macbook Pro laptop. The control loop runs at 50Hz, the video capture and horizon detection runs at 20 Hz, and the runway estimation runs at 1Hz. Although the runway course corrections can only occur intermittently, the horizon control keeps the attitude of the UAV stable at a higher frequency. Additionally, the loops controlling the ailerons and elevator are updated even faster than their inputs, which makes the control surface movements smoother rather than jerky.

We tested our method by having it land an airplane in Microsoft Flight Simulator X, using the built-in scenery and airports. The reference frames were generated by recording the video of a landing controlled by the built-in autopilot, which also serves as a performance benchmark.

In order to be as realistic as possible, the computer with the vision system is separated by hardware interfaces from the computer running the simulation program. The monitor

Fig. 8. The result of performing registration between a test frame *(right)* and a reference frame *(left)* is a matrix that can map every point in the reference image to a corresponding coordinate in the landing image, as shown by the warped image *(top)*.

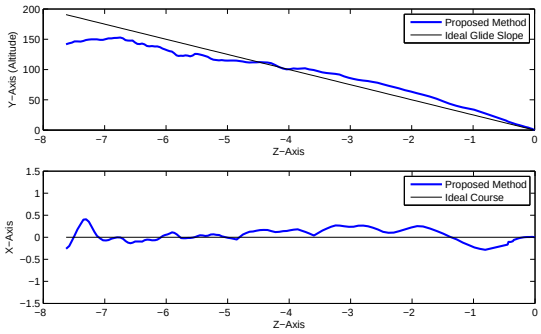


Fig. 9. Polar plots of the UAV position relative to the runway over time, controlled by our proposed method

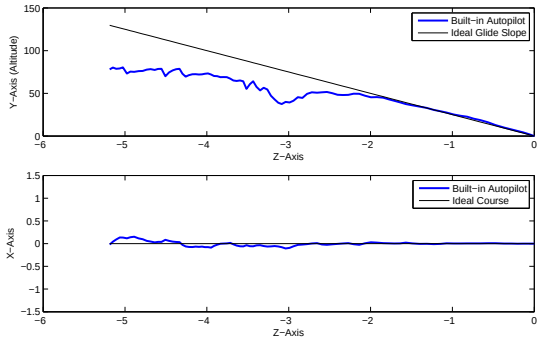


Fig. 10. Polar plots of the UAV position relative to the runway over time, controlled by the built-in autopilot

TABLE I

ERROR IN Y-AXIS (GLIDESLOPE)

	Mean Abs Error	Std Deviation
Built-in Autopilot	22.0	16.2
Proposed Method (Run 1)	10.5	11.9
(Run 2)	11.9	12.7
(Run 3)	12.9	15.1
(Run 4)	13.8	16.2
(Run 5)	12.6	14.1

TABLE II

ERROR IN X-AXIS (PARALLEL TO THE GROUND)

	Mean Abs Error	Std Deviation
Built-in Autopilot	74.5	36.6
Proposed Method (Run 1)	96.2	48.8
(Run 2)	100.9	46.5
(Run 3)	101.8	60.0
(Run 4)	111.7	59.3
(Run 5)	103.8	52.2

output of the simulation computer is sent through a firewire framegrabber that provides video for the vision computer. The vision computer then sends commands to the simulation computer over ethernet, where the commands are interpreted as a virtual joystick device.

The estimated course deviation and glide slope for a typical experiment are shown as a signals in Figure 9. For clarity, we display the measured angles as distances by assuming that horizontal distance is linear with respect to time. Since the camera was not calibrated, and the altitude and speed of the UAV was not measured, the signals do not have actual units are estimates assumed to be linearly related to the actual altitude and lateral deviation. For comparison, the signals are also shown for the landing controlled by the built-in autopilot in Figure 10. The mean absolute error and standard deviation are also shown in Tables I and II.

Our method reliably lands the plane safely on the runway, although it sometimes rolls off the runway into the nearby grass. The greatest cause of error is that as the UAV gets closer to the runway, a small deviation in the X axis causes an increasingly larger angular deviation. Since the linear PI controller operates on the angular deviation estimate, the UAV overcorrects and oscillates as it gets closer. Another note is that the curvature of the earth causes the runway to appear closer to the horizon when the UAV is farther away. This is why the built-in autopilot is estimated to be too low when it is further away. Our system controls the UAV to keep the apparent distance between the horizon and the runway constant, so it is actually too high when it appears to have a constant glide slope. Both of these problems could be solved by including into the algorithm estimates for the camera calibration focal length, the initial altitude, and airspeed.

At this time, we were unable to fly an actual UAV under autonomous control or over a real runway. Instead, as a way of demonstrating the applicability of our method to real data, we show that the approach can be used on a video that was taken from while flying a UAV over a road. An arbitrary point in the road is chosen to represent the beginning of the runway. An example of estimating the position of the runway is shown in Figure 8.

VII. CONCLUSION

We have proposed a system for landing a UAV on a runway using only a camera as a sensor. Our method is unique in that it uses the projective geometry of a forward-looking camera to simplify the pose-estimation problem in this situation. Some advantages of our approach are that it does not rely on a specific flight model for the UAV or require camera calibration, nor is it necessary for the runway to have a specific size, shape, or set of markings. Since the terrain rather than the runway is used as visual information, it is possible to control the UAV before the runway is visible.

Extensions of this system could include cross-country navigation and waypoint-following that lead to a landing approach. A higher-level vision method may be able to recognize distinctive landmarks such as rivers or highways, which would be more robust to varying viewpoints and lighting conditions. This may also permit registration against a satellite map, removing the need for a pre-recorded flight.

REFERENCES

[1] S. Saripalli, J. Montgomery, and G. Sukhatme, "Vision-based autonomous landing of an unmanned aerial vehicle," ICRA, 2002.

[2] C. Sharp, O. Shakernia, and S. Sastry, "A vision system for landing an unmanned aerial vehicle," ICRA, 2001.

[3] N. Trawny, A. I. Mourikis, S. I. Roumeliotis, A. E. Johnson, and J. F. Montgomery, "Vision-aided inertial navigation for pin-point landing using observations of mapped landmarks: Research articles," *J. Field Robot.*, vol. 24, no. 5, pp. 357–378, 2007.

[4] S. Ettinger, "Vision-guided flight stability and control for micro air vehicles," Masters Thesis, University of Florida, 2001.

[5] C. De Wagter, A. Proctor, and E. Johnson, "Vision-only aircraft flight control," Digital Avionics Systems Conference, 2003.

[6] P. Y. O. William E. Green and G. Barrows, "Flying insect inspired vision for autonomous aerial robot maneuvers in near-earth environments," ICRA, 2004.

[7] F. Rafi, S. M. Khan, K. Shafiq, and M. Shah, "Autonomous target following by unmanned aerial vehicles," SPIE Defence and Security Symposium 2006, Orlando FL.

[8] A. Johnson, J. Montgomery, and L. Matthies, "Vision guided landing of an autonomous helicopter in hazardous terrain," ICRA, 2005.

[9] A. Hakeem, R. Vezzani, A. Yilmaz, M. Shah, and R. Cucchiara, "Using spatiotemporal geometric constraints for estimating trajectories of hand-held cameras," Submitted to ISPRS Journal of Photogrammetry and Remote Sensing, 2007.

[10] J. Krumm and S. Shafer, "Shape from periodic texture using the spectrogram," in *Proceedings of the 1992 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR '92)*, June 1992, pp. 284 – 289.

[11] Y. Sheikh, N. Haering, and M. Shah, "Shape from dynamic texture for planes," in *CVPR '06: Proceedings of the 2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*. Washington, DC, USA: IEEE Computer Society, 2006, pp. 2285–2292.

[12] Y. Sheikh and M. Shah, "Aligning dissimilar images directly," Asian Conference on Computer Vision, 2004.

[13] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," IJCV, 2004.

[14] M. A. Fischler and R. C. Bolles., "Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography," Comm. of the ACM, Vol 24, pp 381-395, 1981.