Improved Boundary Tracking by Off-Boundary Detection

Alex Chen[a]

[a]University of North Carolina, Chapel Hill, USA

ABSTRACT

This work discusses an improvement to the boundary tracking algorithm introduced by Chen et al 2011. This method samples points in an image locally and utilizes the CUSUM algorithm to reduce tracking problems due to noise or texture. However, when tracking problems do arise, the local nature of the algorithm does not give any mechanism in which to recover. This work introduces a second CUSUM algorithm to detect off-boundary movement, compensating for such movement by backtracking. Boundary tracking results comparing the two algorithms are presented, including both image data and a numerical comparison of the effectiveness of the algorithms.

Keywords: H yperspectral Imaging, Boundary Tracking, Image Segmentation, CUSUM Algorithm, Off-Boundary Detection

1. INTRODUCTION

Boundary tracking methods have been used for various applications from vehicular movement through an unfamiliar environment,[1,2] atomic force microscopy,[3] to image processing.[4,5] For the first two, the local nature of a "tracker" means that only a fraction of points needs to be sampled in order to obtain information about the domain. For the last, in which global information is often available, boundary tracking also has significant benefits due to its simplicity and computational speed.

Chen et al. 2011[5] introduced a boundary tracking method that was particularly useful for the analysis of hyperspectral data. Due to its large size, the processing of hyperspectral data can be very computationally expensive.[6,7] With new sensors that are able to capture both high spatial and high spectral resolution data, efficient methods for boundary and edge detection have become even more important. By only sampling points near the boundary of an object, the boundary tracking algorithm is able to detect object boundaries with a high degree of precision. For the fractal-like coastlines especially prevalent in hyperspectral data, high precision tracking allows the detection of coastal features in great detail.

A significant drawback to boundary tracking in comparison with more traditional image segmentation algorithms,[8–11] however, is its relatively weak performance in the presence of noise or textures. As a local algorithm, it is particularly sensitive to patches of noise, which can lead the tracker astray. Once off of the boundary, only a chance movement back toward the boundary allows the boundary to be tracked again.

One improvement is the use of a change-point detection algorithm such as Page's CUSUM algorithm[12,13] to "average" noise. Weighted intensities along the path of the tracker are summed, and a boundary detection is assigned only when a certain threshold of this statistic is reached. This adaptation significantly improves tracking in noisy and textured data. However, the problem of off-boundary movement still remains.

In this work, we introduce a second CUSUM statistic to track the characteristic behavior of off-boundary movement. Qualitatively, movement along a boundary should consist of alternating values characteristic of both sides of the boundary. On the other hand, movement away from the boundary consists of values primarily from only one region.

To this end, we employ a two-sided CUSUM test of the likelihood ratio of the density functions between the two regions. If estimates for the density functions are unknown, they can be estimated from samples. The null hypothesis corresponds to a mean value of 0, indicating movement along the boundary. The alternative hypothesis is that either an increase or decrease in the mean occurs.

Further author information: E-mail: achen@samsi.info

Once off-boundary movement is detected via the CUSUM test, a backtracking procedure is used to move to the last detected boundary point. Fortunately, the CUSUM test also provides a good estimate for the detection delay in off-boundary movement. Experimentally, it was found that slight perturbations in the position and trajectory vectors result in new paths that may ultimately track the boundary through the problem areas.

One of the most significant applications for boundary tracking is high dimensional or high resolution data such as hyperspectral images. By sampling only along a path, boundary tracking gives a fast method to detect the boundaries of an object. Numerical experiments show that the use of the two-sided CUSUM test for off-boundary movement and backtracking results in fewer tracking failures. This mitigates the most significant weakness in boundary tracking.

2. ALGORITHM DESCRIPTION

The boundary tracking scheme from Chen et al.[5] introduced a tracker initialized at a point $\vec{x}_0$ near the boundary. The tracker attempts to follow the boundary between two regions Ω_1 and Ω_2, whose union represents the image domain Ω. Movement was according to the update equation

$$\vec{x}_k = \vec{x}_{k-1} + (V\cos\theta_{k-1}, V\sin\theta_{k-1}), \tag{1}$$

where V (e.g. $V = 0.5$) is a step size chosen depending on the level of desired tracking detail and θ_k is the angle of travel chosen according to the update equation:

$$\theta_k = \theta_{k-1} + \omega d(\vec{x}_k), \tag{2}$$

where ω is an angular increment, chosen based on the expected straightness of edges. By default, $\omega = 0.5$ radians.

In this formulation, a decision function $d(\cdot)$ is required to determine whether the tracker location is in one region or another. In the simplest form, the decision function is based on a threshold of image intensities in the grayscale case.

The boundary tracking algorithm works well for a variety of examples and is especially useful for large data sets such as hyperspectral data. Fig. 1 shows a tracking example for hyperspectral data with the original boundary tracking algorithm. The decision algorithm used is determined by the minimum spectral angle distance to each class of reference signatures. The spectral angle distance between two vectors $\vec{u}, \vec{v}$ is given by the formula

$$d_{SA}(\vec{u}, \vec{v}) = \cos^{-1}\left(\frac{\vec{u}\cdot\vec{v}}{\|\vec{u}\|\|\vec{v}\|}\right) \tag{3}$$

The spectral angle distance is known to work especially well for hyperspectral data,[14,15] in part due to its removal of illumination effects. By defining two classes of objects $\{\vec{u}_i\}_{i=1}^{L}, \{\vec{v}_j\}_{j=1}^{K}$, based on spectral signatures, two regions can be defined: $\Omega_1 = \{\vec{x} : \min_i d_{SA}(\vec{x}, \vec{u}_i) < \min_j d_{SA}(\vec{x}, \vec{v}_j)\}$ and $\Omega_2 = \Omega\backslash\Omega_1$. The reference signatures can be chosen by hand or by an automatic selection scheme.[16,17]

It was found, however, that the tracker could make significant errors in tracking due to noise. The boundary tracking algorithm therefore included a CUSUM algorithm that modified the decision function. It supposed that the density functions f, g of image intensities s_k (or another function defined on the imaging domain) of the regions Ω_1, Ω_2 were known. Then changes from Ω_1 to Ω_2 could be tracked by registering a change only when the statistic

$$U_k = \max\left(U_{k-1} + Z_k, 0\right),\ U_0 = 0 \tag{4}$$

for $Z_k = \log[g(s_k)/f(s_k)]$ exceeded a threshold $\bar{U}$. This ensured that changes were recorded only when enough evidence suggested that a region change had been made. Changes from Ω_2 to Ω_1 were tracked similarly by tracking the statistic L_k, given by $L_k = \max(L_{k-1} - Z_k, 0)$ and registering a change when $\bar{L}$ was exceeded. For boundary tracking applications in moderate noise, relative low values $\bar{U} = \bar{L} = 0.8$ are sufficient.

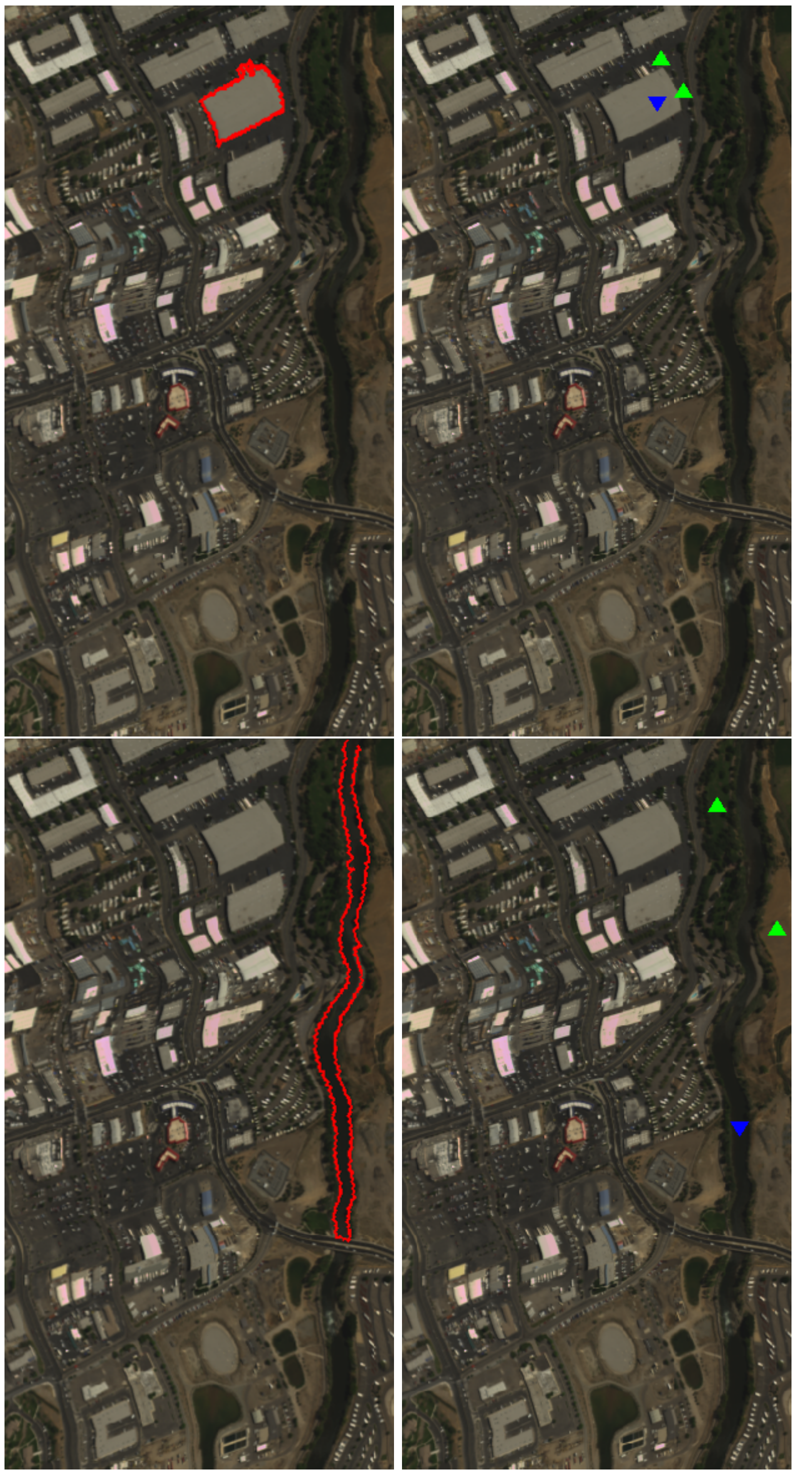

Figure 1. Hyperspectral boundary tracking[5] of various objects in the hyperspectral data set from Reno, NV.[18] Top Left: The tracking of a building. Top Right: The reference signatures used in each class for the building tracking. Bottom Left: The tracking of a river. Bottom Right: The reference signatures used in each class for the river tracking.

However, tracking errors could still occur for excessively noisy or textured images. In case the tracker left the proximity of the boundary, it could only arrive back at the boundary by chance. Indeed, since the algorithm assumed that the boundary would be followed closely, no part of the algorithm considered any global mistakes. The sinusoidal nature of the tracking path ensured that the tracker was often able to self-correct, but the probability of back-tracking was much lower the farther away the tracker was led.

For this case, we introduce a second CUSUM algorithm that detects and corrects for off-boundary movement. In fact, the same statistics U_k and L_k can be used for this step. In the previous case, changes from Ω_1 to Ω_2 were detected when U_k exceeded a certain threshold. This indicated that enough observations likely sampled from Ω_2 had been obtained, when measured against the likelihood of the observations being sampled from Ω_1.

In contrast, off-boundary movement assumes that the tracker is still moving in Ω_1, so that the criterion should be reversed: $L_k \geq \bar{L}_2$ for some threshold $\bar{L}_2$ indicating off-boundary movement. Similarly, off-boundary movement originating in the region Ω_2 can be tracked via the criterion $U_k \geq \bar{U}_2$.

In practice, off-boundary movement is considerably more rare than correct tracking. Furthermore, the sinusoidal movement of the tracker is already somewhat self-correcting, so that only a more serious loss of the boundary is treated. Thus, $\bar{U}_2, \bar{L}_2$ are typically set much higher, at 30. Once this threshold is reached, the algorithm backtracks by according to the "kicking" introduced in:[5] the tracker is placed in the direction of the last known boundary point at twice the distance.

Fig. 2 shows a comparison of the time series produced by successful and unsuccessful trackings along a boundary. In a successful tracking, the time series samples points from both sides of the boundary threshold (at 0.5 in the example). For the unsuccessful tracking, however, the tracker remains on one side of the boundary, so it samples points only above or below the 0.5 threshold. In the example given, the tracker is driven to the lighter side (above 0.5) from about $t = 300$ to $t = 400$ while from $t = 400$ to $t = 1300$, the tracker samples points from below 0.5. This can be seen in the corresponding tracking path.

3. NUMERICAL EXAMPLES

Fig. 3 shows a comparison of the boundary tracking method of Chen et al. and the proposed method on a noisy "U" image. The same image has been used in each case, with the tracker initialized at different starting points. The tracking results were classified into three categories. First, in a completely unsuccessful tracking, the boundary was not followed at all (defined as finding less than ten pixels along the boundary). A partially successful tracking followed the boundary for more than ten pixels but did not find the entire boundary. Finally, a completely successful tracking found the entire boundary.

Table 1 shows that the proposed modification results in a much greater percentage of partially successful trackings. However, the number of completely successful trackings was not improved significantly. This may be due to the fact that the adjustments in the path due to off-boundary movement sometimes result in the tracker moving even farther from the boundary. While the CUSUM algorithm correctly identifies cases in which the tracker moves off of the boundary, a stronger correction algorithm would improve boundary tracking even more.

A comparison of boundary tracking success			
Method	Completely unsuccessful	Partially successful	Completely successful
Chen et al. 2011	34	23	43
Proposed Modification	13	43	44

Table 1. A comparison of the number of successful trackings out of 100 trials for the noisy "U" image for the boundary tracking method of Chen at al.[5] and the proposed modification.

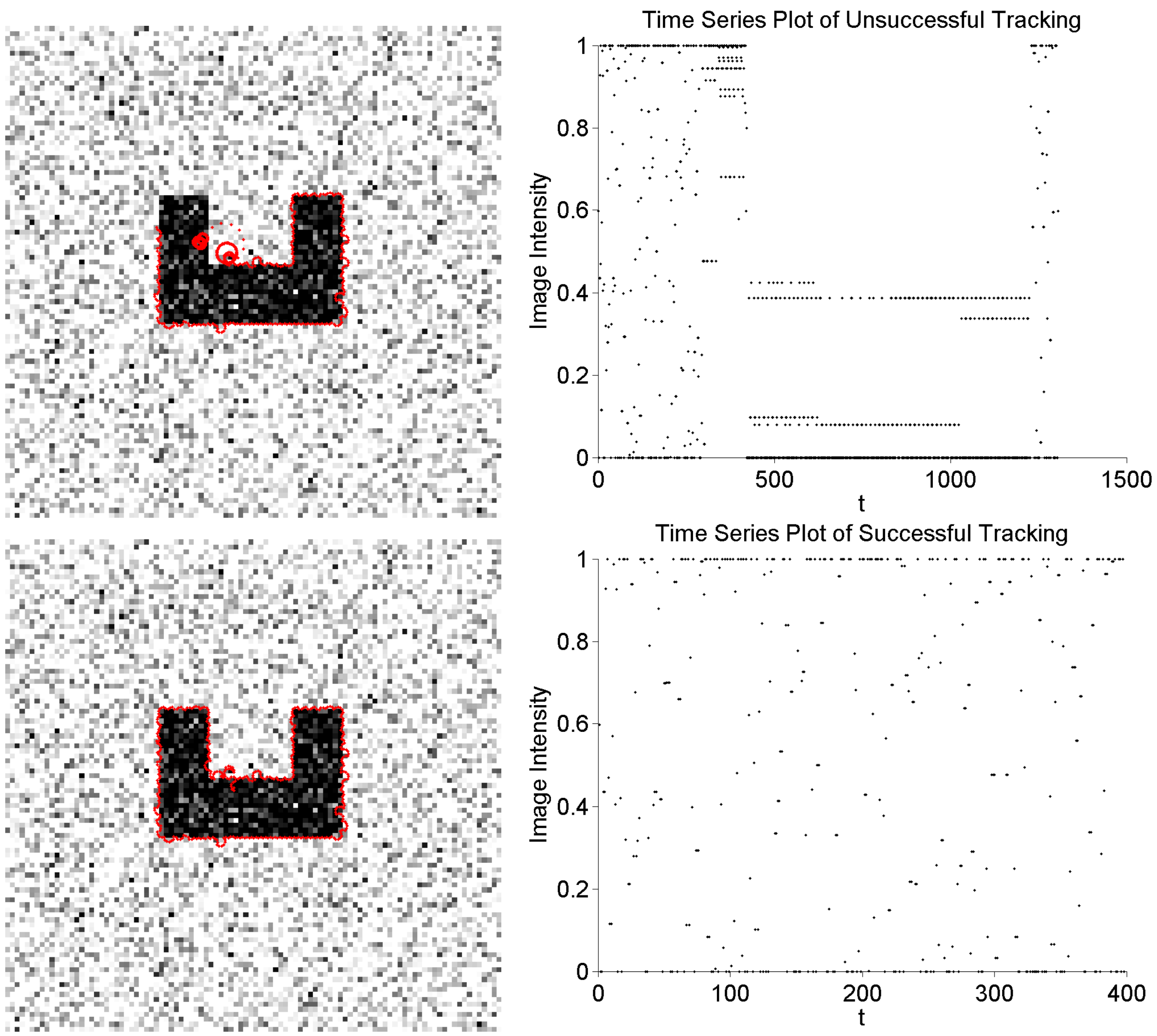

Figure 2. A comparison of the time series of successful and unsuccessful boundary tracking paths. Top Left: An unsuccessful tracking path. Top Right: The corresponding time series to the unsuccessful path. Bottom Left: A successful tracking path. Bottom Right: The corresponding time series to the successful path.

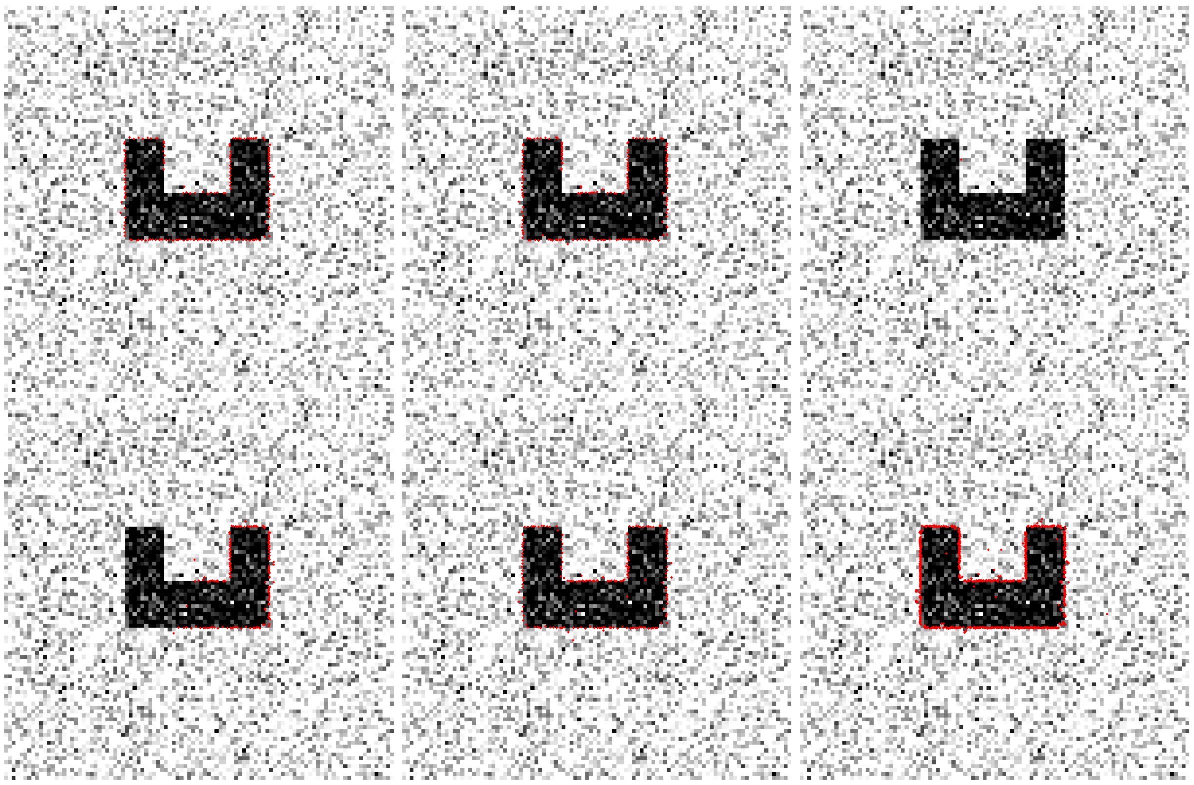

Figure 3. A comparison of the original boundary tracking algorithm and the modified boundary tracking algorithm for various initial starting points. The top images are trackings by the original tracking algorithm and the bottom images use the proposed modification. Each set–left, middle, right, respectively use the same initial starting points. The top right example is a "completely unsuccessful tracking." The top left, top middle, and bottom left examples are "partially successful trackings," while the bottom middle and bottom right examples are "completely successful trackings."

REFERENCES

[1] Marthaler, D. and Bertozzi, A. L., [*Recent Developments in Cooperative Control and Optimization*], Kluwer Academic Publishers (2003). Tracking environmental level sets with autonomous vehicles.

[2] Jin, Z. and Bertozzi, A., "Environmental boundary tracking and estimation using multiple autonomous vehicles," in [*2007 46th IEEE Conference on Decision and Control*], 4918–4923 (December 2007).

[3] Andersson, S., "Curve tracking for rapid imaging in AFM," *IEEE Transactions on Nanobioscience* **6** (December 2007).

[4] Chen, A., Wittman, T., Tartakovsky, A., and Bertozzi, A., "Image segmentation through efficient boundary sampling," in [*Proceedings of the 2009 Workshop on Sampling Theory and Applications*], (May 2009).

[5] Chen, A., Wittman, T., Tartakovsky, A., and Bertozzi, A., "Efficient boundary tracking through sampling," *AMRX* **2011**, 182–214 (September 2011).

[6] Bachmann, C. M., Ainsworth, T. L., and Fusina, R. A., "Exploiting manifold geometry in hyperspectral imagery," *IEEE Transactions on Geoscience and Remote Sensing* **43**(3), 441–454 (2005).

[7] Bachmann, C. M., Ainsworth, T. L., and Fusina, R. A., "Improved manifold coordinate representations of large scale hyperspectral imagery," *IEEE Transactions on Geoscience and Remote Sensing* **44**(10), 2786–2803 (2006).

[8] Kass, M., Witkin, A., and Terzopoulos, D., "Snakes: Active contour models," *International Journal of Computer Vision* **1**(4), 321–331 (1988).

[9] Caselles, V., Kimmel, R., and Sapiro, G., "Geodesic active contours," *International Journal of Computer Vision* **22**(1), 61–79 (1997).

[10] Chan, T. and Vese, L., "Active contours without edges," *IEEE Transactions on Image Processing* **10**, 266–277 (February 2001).

[11] Esedoglu, S. and Tsai, Y. R., "Threshold dynamics for the piecewise constant Mumford-Shah functional," *J. Comput. Phys.* **211**(1), 367–384 (2006).

[12] Page, E. S., "Continuous inspection schemes," *Biometrika* **41**, 100–115 (June 1954).

[13] Basseville, M. and Nikiforov, I., [*Detection of Abrupt Changes - Theory and Applications*], Information and System Sciences Series, Prentice Hall, Englewood Cliffs, NJ, US (1993).

[14] Shippert, P., "Introduction to hyperspectral image analysis," *Online J. of Space Commun.* (2003).

[15] Yunas, R., Goetz, A., and Boardman, J., "Discrimination among semi-arid landscape endmembers using the spectral angle mapper (SAM) algorithm," in [*Summaries of the Third Annual JPL Airborne Geoscience Workshop*], 147–149 (1992).

[16] Chen, A., "Active contours with edges: Combining hyperspectral and grayscale segmentation," in [*Proceedings of SPIE Remote Sensing*],

[17] Winter, M., "N-FINDR: an algorithm for fast autonomous spectral end-member determination in hyperspectral data," in [*SPIE Conference on Imaging Spectrometry V*], **3753** (1999).

[18] "Urban and mixed environment sample: Reno, NV, USA." SpecTIR Reflectance + IGM.