2014

AR Browser for Points of Interest in Disaster Response in UAV Imagery

Danielle Crowley  
*Texas A&M University, decrowley10@gmail.com*

Tim McLaughlin  
*Texas A&M University, timm@viz.tamu.edu*

Robin Murphy  
*Texas A&M University, murphy@cse.tamu.edu*

Brittany Duncan  
*University of Nebraska - Lincoln, bduncan@unl.edu*

Ann McNamara  
*Texas A&M University, ann@viz.tamu.edu*

Follow this and additional works at: [http://digitalcommons.unl.edu/cseconfwork](http://digitalcommons.unl.edu/cseconfwork)

Crowley, Danielle; McLaughlin, Tim; Murphy, Robin; Duncan, Brittany; and McNamara, Ann, "AR Browser for Points of Interest in Disaster Response in UAV Imagery" (2014). *CSE Conference and Workshop Papers*. 247.  
[http://digitalcommons.unl.edu/cseconfwork/247](http://digitalcommons.unl.edu/cseconfwork/247)
AR Browser for Points of Interest in Disaster Response in UAV Imagery

Danielle Crowley
Texas A&M University
Langford Center
3137 TAMU
College Station, TX 77843-3137
decrowley10@gmail.com

Robin Murphy
Texas A&M University
Department of Computer Science & Engineering
333 H.R. Bright Building
3112 TAMU
College Station, TX 77843-3137
murphy@cse.tamu.edu

Ann McNamara
Texas A&M University
Department of Visualization
C306 Langford Center
3137 TAMU
College Station, TX 77843-3137
ann@viz.tamu.edu

Tim McLaughlin
Texas A&M University
Department of Visualization
C108 Langford Center
3137 TAMU
College Station, TX 77843-3137
timm@viz.tamu.edu

Brittany Duncan
Texas A&M University
Department of Computer Science and Engineering
333 H.R. Bright Building
3112 TAMU
College Station, TX 77843-3137
bduncan@cse.tamu.edu

Abstract
This work in progress describes AerialAR, a global positioning system (GPS) augmented reality (AR) application for mobile devices that automatically labels points of interest (POI) in unmanned aerial vehicle (UAV) imagery. This has important implications for assisting emergency responders. Existing AR applications for UAVs provide the pilot with navigational situational awareness such as terrain features; AerialAR locates and labels mission-relevant points such as schools that may need to be evacuated or hospitals to transport victims to. Locating POI in UAV imagery poses more challenges than those addressed by typical AR browsers on smartphones. The UAV operates at different altitudes as opposed to handheld devices and the UAV camera can tilt over a wide range of angles rather than simply facing forward. AerialAR overcomes these issues by developing a set of equations that translate UAV telemetry and field of view (fov) into a projection onto a Google Map. The map can then be queried for categories of POI. The current version calculates the POI distance and angles with an average error of 0.04% as compared to the Haversine and Rhumb line equations for the distance between the UAV location projected on the ground and the POI on the Google Map. Future work will complete AerialAR by processing UAV video in real-time on mobile devices.
1 Introduction

Small, unmanned aerial vehicles (UAVs) are being adopted into disaster response to allow responders to actively view incidents. Recent work showed that responders had difficulty determining important or hiding buildings [1] For example, a responder viewing a chemical train disaster to identify leaking materials but also needs to ascertain if derailment is near an densely populated building, such as a school, that should be immediately evacuated.

Currently, disaster responders must look back and forth between the display streaming video from the UAV and a separate display showing a Google map to reconcile the information from the imagery and the map. This wastes valuable time and is potentially open to human error when identifying Points of Interest (POIs)

This paper describes a new iOS application, AerialAR, which automatically identifies and labels POIs in UAV imagery given the following input: UAV telemetry, camera field of view, and a POI search term from the responder. It assumes that there is wireless connectivity, either to the internet or to a local responder network and Google server. AerialAR contributes both a new application of AR but also new mathematical formulae for accurately determining the relationship between the 3D UAV’s GPS, altitude, camera tilt, and field of view information with the 2D Google map. The ultimate version of AerialAR is shown in Figure 1. The current version works only on still images.

2 Related Work

GPS-based AR, also called AR browsers, is better suited for UAVs than marker-based AR or natural feature tracking AR as these rely on modifications to the environment or computer vision. Augmented reality for UAVs has focused on displaying information critical for the navigation, enemy targeting or flight safety of the platform. AerialAR is not intended to help the pilot fly the UAV but rather help the responder(s) viewing the UAV video to comprehend the impact of the disaster and immediate environment surrounding the disaster.

2.1 Augmented Reality

AR combines computer-generated information and real-time footage obtained by a camera. AR overlays virtual information on the physical world. AR can be categorized into three categories: marker-based AR, natural-feature tracking, and GPS-based. In marker-based AR, the system recognizes a fiducial or is able to recognize images preregistered with a specific AR device in order to obtain the camera’s position. Natural-feature tracking AR relies on computer vision and is very difficult to achieve in outdoor lighting [1] and due to the difficulty in reliably distinguishing buildings and purpose. GPS-based AR allows computer-generated content to be accurately displayed despite lack of a fiducial marker or poor visibility.
AerialAR employs GPS-based AR, also called an AR browser. AR browsers place labels in the vicinity of where a building is located on the screen as the user looks through the camera at the buildings [3]. AerialAR is different from existing AR browsers as it is tailored for use with UAVs. This means it needs to accommodate imagery at different altitudes and different camera angles.

2.2 Augmented Reality for UAVs
Existing Augmented Reality systems for UAVs can be divided into two categories: heads-up displays and general navigational situation awareness enhancement. Heads-up displays superimpose vector graphics upon the pilot’s view or screen of the real world and typically provide basic navigation and flight information, as well as graphics registered with targets in the environment [7].

Situation awareness AR systems are concerned with navigational situation awareness, either traversing complex terrains [2], providing depth information for time to contact [5], or locations of other UAVs [4]. The three identifying systems were split between marker-based AR [5], natural feature tracking [5], and GPS-based [4,2].

3 Overview of AerialAR
AerialAR has four inputs:

- UAV telemetry (latitude, longitude, altitude, heading, and camera pitch) as a XML file. The telemetry can be obtained through an XML file that is returned to the UAS.
- The horizontal and vertical field of view of the camera, known a priori.
- a camera image
- a POI search terms from the 126 search terms are integrated into the Google Maps API.

It uses the inputs to project the camera view onto a 2D Google Map. The Google Map API is queried for the points of interest, which are returned as latitude and longitude coordinates. AerialAR then creates labels which are scaled so that closer POIs have larger fonts and displays the labels on the image. It also places a rectangular box as a marker on the POI as shown in Figure 2.

The software is divided into four major functions:

The init function takes user input regarding UAV position. The URL request function uses the UAV latitude, longitude, and the POI type from the init function and creates a HTTP request in order to gather POI data from Google Maps. Google Maps API accepts 126 general establishment types such as schools, hospitals, and police departments. The URL response function parses the returned JSON file from the http request and creates POI objects with latitude, longitude, and name variables.

The filter function calculates the distance and orientation of the POIs in relation to the UAV position. As the position of the center of view and UAV xy coordinate of the camera are found by using distance equations, points are created. Points consist of an x and y value. After these points are found, vectors for the UAV to the POI, UAV xy coordinate to the center of

---

2 JSON (JavaScript Object Notation) is a syntax for storing and exchanging text information and is the format Google Maps API uses to send data to the user.
The vector class function then calculates the magnitudes and the angles between the vectors. By finding these angles and comparing them to the field of view, the function can create a new list of POIs that are within the UAV’s field of view. The list of POIs contains the image location information, which is found by using the ratio of the angle between the vectors to the UAV’s field of view.

A picture view function is called when the button labeled “View picture” is pressed and uses the POI’s image location to position a text label on the UAV image which is input by the user.

AerialAR is initially targeted for iOS applications and specifically iPads. While AerialAR can be used on the iPhone, it is better suited for the iPad due to the larger screen. The iOS development platform was chosen because responders frequently carry iPads and iPhones because this allows them to be continuously mobile and limits the amount of equipment they have to carry.

4 Key Equations

The equations for finding the distance and bearing between two GPS points (the position of the UAV and the POIs in the world frame) replace the Haversine and Rhumb line equations. The vector equations operate in three-dimensions whereas the Haversine and Rhumb line equations only work in two-dimensions. Vectors represent the distances between points. The vectors calculated are outlined in Figures 4 and 5. It is assumed that the POIs are stationary.

4.1 UAV position

The position of the UAV in the world frame as shown in Figures 3 and 4 is \((ux, uy, uz)\) where \(ux\) is the latitude, \(uy\) is the longitude, and \(uz\) is the altitude, \(a\). The camera is positioned in the body of the UAV and the horizontal rotation of the camera on the \(z\)-axis is the heading, \(h\). The UAV \(xy\) coordinate is the point directly on the ground below the UAV \((ux, uy, 0)\). The pitch of the camera, \(p\) in Figure 5, is the vertical rotation of the camera on the \(y\)-axis. The horizontal and vertical field of view angles are \(fh\) and \(fv\) respectively.

4.2 UAV \(xy\) coordinate to center of view vector

Finding the center of view of the UAV camera is important in order to calculate the POI location in respect to the UAV and the image pixel location of the POI in the imagery. The center of view is the middle of the horizontal field of view and the middle of the vertical field of view as seen in Figure 3. The center of view is the point in the exact middle of the UAV image. The distance of the center of view, \(c\), was found by Equation (1) in Figure 6.

The result of this calculation is a distance in feet without a direction. A counter clockwise rotation matrix is applied to the distance in order to find the \(x\) and \(y\) displacement values, taking into account the UAV heading. In order to simplify finding if a POI is without the horizontal field of view, rotate the center of view coordinate by the matrix and the heading but also rotate it to the left most corner of the field of view. This allows for a comparison of the whole horizontal field. The results are returned in feet and the equations for this transformation are Equations (2) and (3).
4.3 UAV xy coordinate to POI vector
The UAV xy coordinate to POI vector is the distance that the POI is away from the UAV xy coordinate. In order to find the distance between the POI and the UAV xy coordinate, find the difference in their latitudes and longitudes. This distance is returned in degrees and to calculate the magnitude of the vector, the degrees must be converted to feet. The conversion for longitude to feet, where \(i_x\) is feet in the x direction, \(u_x\) is the camera latitude, and \(\delta\) lon is the found longitude difference between the POI and UAV xy coordinate, is Equation (4).

\[
i_x = 69.17222 \times \cos(u_x) \times 528(4)
\]

\[
i_y = 364636.8 \times \Delta\text{lat}(5)
\]

\[
m_i = \sqrt{i_x^2 + i_y^2}(6)
\]

\[
\theta_h = \cos^{-1}\left(\frac{c_x + i_x + c_y + i_y}{m_c + m_i}\right)(7)
\]

\[
\theta_v = \theta_v - (p - (f_v/2))(8)
\]

4.4 Angles between center of view POI
The horizontal angle between the center of view and POI determines how far the POI is offset in the horizontal field of view from the UAV. The equation to find the angle is Equation (7).

\[
f_o > \theta_t(9)
\]

\[
(h + (f_h/2)) > \theta_h > (h - (f_h/2))(10)
\]

\[
w_x = \left(\theta_t/f_h\right) \times r_w(11)
\]

\[
w_y = \left(\theta_v/f_v\right) \times r_h(12)
\]

4.5 Angles between UAV and POI
It will eliminate extra variables and reduce any arcing effect by temporarily rotating the camera to face directly towards the POI. Two vectors can be created in order to find the angle from the UAV to POI and the first vector is the UAV xy coordinate to the UAV, \((0, 0)\) to \((0, a)\). The second vector is from the UAV xy coordinate to the POI: from \((0, 0)\) to \((m, a)\). The angle between these two vectors, \(\theta_v\), is solved with the previous equation. The angle must be adjusted to find the angle vertical from the bottom of the field of view instead of the angle from the UAV xy coordinate. This adjustment is shown in Equation (8).

4.6 Finding POI pixel location in image
The pixel location of a POI is the \((x, y)\) pixel coordinates of where the POI is in the UAV imagery. In order for a POI to be labeled, the POI must be within the field of view of the camera. The conditions that must be passed are Equations (9) and (10).

\[
\alpha > \theta_t\text{ and } \beta > \theta_h(10)
\]

If the POI passes the conditions then the pixel values \((w_x, w_y)\), where \(r_w\) is the screen’s width resolution and \(r_h\) is the screen’s height resolution, can be found by Equations (11) and (12).

\[
w_x = \left(\theta_t/r_w\right) \times r_w(11)
\]

\[
w_y = \left(\theta_v/r_v\right) \times r_h(12)
\]
from the scene. The targets on the buildings are red and white in order draw attention to the POI position on screen and appear prominently.

The project uncovered two potential problems for a UAV AR browser: POIs may be along the horizon, as seen in Figure 8, making it difficult to visually determine if the label was on the correct building and the latitude and longitude coordinates returned by Google Maps vary in position relative to the POI.

Conclusions
The equations for finding POIs within the field of view of a UAV are accurate within 0.04%. Although AerialAR is being developed for iOS, the equations and general software organization are applicable to an AR browser for UAV imagery on any mobile device.

Future work on this application will mark and label POIs on real-time footage. While the determination of the POI is accurate, user acceptance of the labeling scheme has not been established. Future work will evaluate the position of labels and markers and the labeling font and coloring.

Acknowledgments
This work was funded through NSF Grant CNS-1263027. The authors thank Kyle Barnes and Ivan Espinosa for their assistance.

References


