

# NAVIGATION IN DENSE HUMAN CROWDS USING SMARTPHONE TRAJECTORIES AND OPTICAL AERIAL IMAGERY

O. Meynberg<sup>a</sup>, F. Hillen<sup>b</sup>, B. Höfle<sup>c</sup>

<sup>a</sup> Remote Sensing Technology Institute, German Aerospace Center (DLR), Oberpfaffenhofen, 82234 Wessling, Germany, oliver.meynberg@dlr.de

<sup>b</sup> Institute for Geoinformatics and Remote Sensing (IGF), University of Osnabrueck, Barbarastr. 22b, 49076 Osnabrueck, Germany, fhillen@igf.uos.de

<sup>c</sup> Institute of Geography, University of Heidelberg, Berliner Str. 48, 69120 Heidelberg, Germany, bernhard.hoefle@geog.uni-heidelberg.de

**KEY WORDS:** aerial images, pattern recognition, Bag-of-Words model, texture classification, smartphone trajectories, information fusion, lowest-cost navigation

## ABSTRACT:

In this paper, we propose a navigation system for smartphones which enables visitors of very large events to avoid crowded areas or narrow streets and to navigate out of a dense crowd quickly. Therefore, two types of sensor data are integrated. First, optical images acquired and transmitted by an airborne camera system are used to compute an estimation of a crowd density map. For this purpose, a patch-based approach with a Bag-of-Visual-Words Framework for texture classification in combination with an interest point detector and a smoothing function is used. Second, the GPS location information and the current movement speed of the visitor are gathered via a smartphone app and is afterwards analyzed to enhance the density estimation. The combined density information is afterwards used for a lowest-cost navigation. Two possible use cases are described namely an emergency application as well as a basic navigation application. A prototype implementation is conducted as a proof of concept.

## 1 INTRODUCTION

During mass events with tens of thousands of visitors event organizers and security authorities have very limited information about current location and number of visitors. Terrestrial sensors, like CCTV cameras, are often only available at the most important spots and have a limited field of view. Airborne monitoring systems can provide additional high-resolution imagery in real-time (Haarbrink and Koers, 2006). Due to their mobility and large field of view even the largest festivals can be covered in a couple of minutes. With their increasing ground resolution, objects like cars and persons (Schmidt and Hinz, 2011) can be detected.

In this paper, the detection results of these methods are leveraged and integrated into an implementation which can be used by the official authorities as well as every visitor. It could be of great use at open-air festivals where large groups gather spontaneously and not always at well monitored locations. Events in city centers often take place not only at a single plaza or market but also in the neighboring streets which are often not wide enough for large groups to pass without problems. Also, the motion of crowds in the space directly in front of a concert stage is hardly predictable during a live performance. Although the audience's number might be below the area's maximum capacity, the number of persons per square meter (crowd density) can quickly reach a critical level and lead to dangerous situations. If the crowd density gets above a certain threshold, the situation can become life threatening and the authorities must intervene. But even if the scenario is not that extreme individuals experience these situations differently. The current physical condition, hydration level, alcohol abuse as well as the weather condition can be a reason for the person to leave the crowded area on the fastest way possible.

For this situations, we propose a concept of fusing 9-cm optical, aerial images with spatiotemporal location and movement data from smartphones on the ground. The idea is to provide the visitor with an up-to-date crowd density map layer and a lowest-cost

routing functionality.

This paper is structured as follows. Section 2 describes the preparation and processing of the respective sensor data as well as their fusion. The potential benefit of our approach for the user is outlined in section 3 by describing possible applications. In Section 4, the implementation of a demonstrable prototype including first navigation results are presented. Section 5 concludes this paper with an outlook on the potential of our approach.

## 2 METHODOLOGY

In this section, the automatic patch-based crowd density estimation as well as the extraction of GPS trajectories from mobile smartphone sensor data are described. Furthermore, it is shown how these two different kinds of sensor data are combined to a cost layer for the lowest-cost navigation.

### 2.1 Crowd Density from Aerial Images

In the following, the process to automatically compute a density map from aerial images is described. Eventually, this density map serves as input for the cost navigation in GRASS GIS (see section 2.2). Because the integration of both kinds of sensor data takes place in the world coordinate system, the aerial images are georeferenced and orthorectified (Kurz et al., 2012) before the crowd detection is initialized. Generally, the crowd detector does not rely on this step and it works also with uncalibrated images, as long as the oblique viewing angle of the camera does not exceed 35°.

Detecting human crowds in still aerial images with a spatial resolution of 9 cm is subject to ongoing research. At this resolution, a single person appears as a small blob of roughly 7x7 pixels. In very crowded scenes it is hardly possible to discriminate these blobs due to occlusion, they rather form a heterogeneous texture without any orientation or regular pattern structure (Figure 1a). Moreover, the appearance of these "crowd textures" distinctly



(a) High crowd density, many occlusions (b) Low contrast

Figure 1: Example of two  $100 \times 100$  image patches containing human crowds ( $GSD = 9cm$ ). The major challenges are a varying lighting condition, varying background, and mutual occlusion.

changes depending on background pixels and lighting conditions (Figure 1b).

To overcome these problems, a patch-based approach using the Bag-of-Words (BoW) framework (Csurka et al., 2004) to detect crowds is proposed. It consists of three main steps: local descriptor extraction, vocabulary learning, and feature coding. The first step in this method is local feature extraction, which creates a set of rotation-invariant feature vectors (Liu et al., 2012) for a given image patch which have low intra-class variability with changing texture orientation but high inter-class variability between non-crowd and crowd textures. In the second step,  $k$ -means clustering is applied to partition the local feature space into distinct and discriminative regions or "visual words". Building the vocabulary of visual words is done only once before training the classifier and does not need to be repeated during classification. A  $k$ -d tree data structure is applied to speed up querying the nearest cluster centers from the dictionary for a given local feature. In the third step, with the local feature and its nearest cluster centers the final feature vector can be computed using the "Vector of Locally Aggregated Descriptors" (VLAD) feature encoding (Jegou et al., 2010) where the sum of a weighted distance of the local feature to the cluster centers is computed. This feature vector is directly used for a Support Vector Machine (SVM) with a linear kernel. Finally, one probability per image patch is derived from the SVM's output to create a regular grid of probabilities with the same dimensions as the original aerial image (Figure 2).

The next step converts this grid of probabilities to an estimation of the crowd density. Therefore, the crowd density is considered as a probability density function (pdf) with a Gaussian kernel over the image domain. To get the pdf, the FAST interest point detector (Rosten and Drummond, 2006) is applied on the original image, weight the detected corners with the above computed probability grid and convolve this with a Gaussian filter. In this way, the crowd density estimation can be expressed as an intensity value which can be assigned to every pixel of the original image and not only to a finite and very sparse set of detected corners. This crowd density estimation is not calibrated to a reference dataset of validated crowd densities, however, it is still sufficient to serve as a two-dimensional cost function in this context.

## 2.2 Movement Trajectories from Smartphone Data

The smartphone data is gathered with an app that records various internal sensor data via the Android API. In this case, relevant information is derived from the current GPS position and the estimated moving speed. The app transfers this data into a spatial PostgreSQL/PostGIS database on a webserver. Afterwards, the data is integrated in a GeoServer to be accessible via the WMS and WFS interfaces. Above that, the data can be directly processed in the database or by integration in WPS processes.



Figure 2: This regular grid shows the concept of the crowd estimation process. For every tile of the grid a probability is computed which is derived of the classifier's output. The higher the probability the more convinced is the classifier that this tile contains a crowd texture. Red: high density, blue: low density. The red circles mark the two center stages.

The speed of the smartphone user at the specific geographic location should afterwards be used to reevaluate the crowd density. High movement speed is an indicator for a low crowd density whereas a slow movement speed suggests a high crowd density. However, it has to be determined whether the user is only moving slowly or is actually standing. This might lead to a misclassification of high crowd density.

## 2.3 Information Fusion

The crowd density derived from the aerial image and the smartphone sensor data is combined to result in a cost layer for the lowest-cost navigation. For this purpose, the raster-based geographic information system GRASS GIS is used.

The density layer derived from the aerial image data is used as the base layer for the cost navigation. To avoid high cost ranges from 0 to 255, the layer is reclassified to a range between 1 (no density) and 10 (high density). As even regions with no people do have a certain movement cost, the base value is set to 1. The smartphone data can be integrated in GRASS GIS and is converted from vector to raster format. Afterwards, the point measurements are expanded to a certain size to affect a larger area on the one hand and to balance the GPS inaccuracy on the other hand. The cost layer is then complemented with the costs from the smartphone measurements. Finally, the lowest-cost path between two arbitrary points can be calculated with GRASS GIS functionality.

## 3 APPLICATION SCENARIOS

In this section, two test scenarios are outlined in which our real-time navigation approach can be useful. The first scenario describes a tool to escape from emergency situations whereas the second scenario presents a generic decision support application

that can be used in various situations. The "give and take" principle is essential for all applications as the user have to provide information about the location and the speed via smartphone to receive a result.

### 3.1 The Fastest Way out of a Crowd

One application in which the real-time navigation can be used is during music festivals. The crowd in front of the music stages is often very dense. Combined with extreme weather conditions and physical fatigue this might result in dangerous situations. Pictures of security guards pulling helpless and faint people out of the crowd are omnipresent in news and social media. In such crowds there is no chance to overview the situation and to find the best way out, especially for persons with a low height (e.g. young girls). Because of the lack of orientation, the person might go towards an even denser region within the crowd, not knowing that a free space was very close by.

Our navigation approach can be integrated in an emergency app provided by the event organizer. The guests have to provide their current location and speed measured with their smartphone. In exchange for that, they are able to view an overview map with the current crowd distribution in the event area and are able to use the described emergency navigation (Figure 3).



Figure 3: Schematic representation of an emergency smartphone app. The current crowd density is visualized in the background. The fastest escape route is emphasized with a red arrow.

### 3.2 The Fastest Way Towards a Specific Point

Another situation in which the real-time navigation can be used is during and after football games. In this particular use case, dense crowds are gathering only in short time frames (e.g. after the game is finished or in the half-time break). A concrete example for this is reported by officials of the Borussia Park in Mönchengladbach (Germany). After the football games, the main way towards the parking spaces is commonly blocked by the police to escort the fans of the opponent to their buses. In the meantime, many people have to wait while more and more people are streaming out of the stadium towards the parking spaces. The organizers are trying to avoid complications by opening gates that allow the people to go the longer way around the stadium on the other side. In this situation, this way would be much faster compared to waiting in the dense crowd. However, the people that are streaming out of the stadium are often not aware about i) the blockade by the police and ii) the option to use an alternative way. Our navigation approach can help to ease this situation by navigating some visitors to the alternative route. Figure 4 is illustrating the generic workflow in which the user has to provide the current location as well as the goal of the routing. This information is sent to a webserver where the actual calculation for the lowest-cost route is executed. Afterwards, this route is visualized on the smartphone of the user who constantly keeps on reporting about his location and speed. For this example, it is essential

to integrate the smartphone data of as many users as possible to avoid potential jams caused by the navigation system itself. As soon as the alternative route is crowded as well, the system carefully has to decide which direction to choose. If the main way is open again and the crowd dissolves, the navigation system routes the people along the typical way. Thus, potential mass panics or at least a big gathering of people can be avoided. In general, the real-time navigation can be used during any major event for example to reach the nearest refreshment stand during a musical festival or city event. Even a navigation through the city streets to a specific parking garage with an emphasis on avoiding large crowds (e.g. in front of stages) can be useful. In any case, the advantages are on both sides, for the event organizers and the guests. The guests utilize the tool to avoid stress and excitement on the one hand. The organizers on the other hand can ensure the security during the event and increase the attraction by providing modern smartphone apps.



Figure 4: Conceptual design of a smartphone app for lowest-cost navigation during or after a football game.

## 4 PROTOTYPE IMPLEMENTATION

A prototype implementation is conducted as a generic proof of concept to integrate the two data sources. The test data for the prototype is recorded during the music festival Wacken 2013 and shows the festival area with the two main stages (red circles in Figure 2). One can clearly see the dense crowd standing in front of the stages. In a first step, the crowd density is estimated using MATLAB. Consecutively, the calculation of the lowest-cost path is performed using GRASS GIS.

The real-time aspect is not considered in this stage. However, real-time functionality can be easily enabled by transferring the functionality to a webserver in form of server scripts or WPS processes as described in detail at the end of this section.

### 4.1 Image Patch Classification and Density Estimation using MATLAB

In the following, the different processing steps which are necessary to generate the geo-referenced crowd density map out of an aerial image are described. The georeferencing itself can be done either before or after the crowd detection, which is an independent method and does not rely on geo information. In this implementation, the real-time orthorectification by (Kurz et al., 2012) is used which implements direct georeferencing in C++ on a GPU. Due to the GPU's parallel hardware architecture a typical 21-MPix image can be processed in under 0.2s to get the 9-cm ortho photo. Afterwards, the methods to estimate the crowd density are called by a MATLAB script in this prototype version. The computationally intensive local feature extraction function (Liu et al., 2012) is implemented in C++ and called as a compiled MEX file by the main MATLAB script. For the subsequent

clustering with  $k$ -means, building of the  $k$ -d tree data structure, and the VLAD feature encoding, the VLFeat computer-vision library (Vedaldi and Fulkerson, 2008) is used which provides a MATLAB interface. A C++ version of the FAST interest point detection is available from the author, which is then used to generate a MEX file. In short, the computation of the crowd density layer is controlled by the MATLAB script which triggers external or MEX libraries for the computationally expensive parts. In the future, this control script could be replaced by an OpenCV implementation with reasonable effort. Also, certain parts like the local feature extraction, could be accelerated substantially because after image tiling all tiles can be processed independently and in parallel.

#### 4.2 Calculating the Lowest-Cost Path using GRASS GIS

The resulting density layer is afterwards integrated in GRASS GIS as the basis for the cost calculation. In a first step, a cumulative cost layer is created based on the current location of the user. For testing purposes, a position in front of the stages a highly crowded area is assumed. The cumulative costs can then be used to navigate to a defined point or to navigate towards a less dense area out of the crowd. For the latter case, a point within a less dense area has to be identified using for example a nearest point functionality. If the arrival point is known, the lowest-cost path can be calculated. An exemplary result can be seen in the left image of Figure 5.

Afterwards, the smartphone data are integrated in GRASS GIS in the vector format and converted to raster representation. The data is then expanded to create a more realistic impact on the cost layer. After the costs derived from the smartphone data are added to the cost layer, a new lowest-cost path can be calculated (see left image in Figure 5). It can be seen that the route for the user has changed compared to the result shown in left image because of the newly added density information derived from the smartphone movement data.

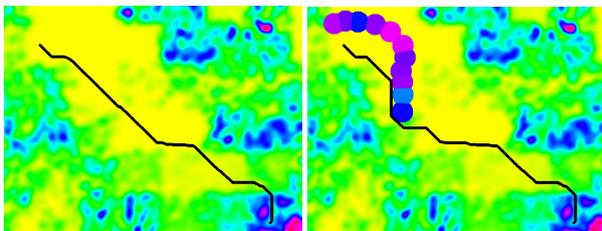


Figure 5: Resulting lowest-cost paths based on the density layer derived from the aerial image data (left) and with the addition of density information derived from smartphone sensor data (right).

#### 4.3 Real-Time Processing

To exploit the real benefit of the lowest-cost approach it is necessary that it can be conducted in real-time. This means that in a first step, the information from both data sources (image and smartphone) has to be accessible in real-time. This can be achieved via existing standards of the Open Geospatial Consortium (OGC) like the Web Map Service (WMS), the Web Feature Service (WFS) or the Web Coverage Service (WCS). Furthermore, the processing of the data and the calculation of the cost layer as well as the lowest-cost path estimation has to be available in real-time. Using GRASS GIS, all processing steps described in the previous section can easily be integrated in a web-based infrastructure to enable the actual real-time usage of the lowest-cost navigation. A python implementation of the Web Processing Service (WPS) standard of the OGC called PyWPS do support a

native link to all GRASS GIS functionality. As GRASS GIS is traditionally console-based, the exact same processing sequence can be executed over the Internet within a PyWPS process.

## 5 CONCLUSION

In this paper, we propose a lowest-cost navigation based on the fusion of aerial image data and smartphone sensor data. The image data is used to estimate a crowd density map. For this purpose, a patch-based approach with a Bag-of-Visual-Words Framework for texture classification in combination with an interest point detector and a smoothing function is used. The GPS location information and the current movement speed of a user are gathered via a smartphone app and is afterwards analyzed to enhance the density estimation. Afterwards, a lowest-cost navigation is conducted based on the combined density information using GRASS GIS. All processing steps can be integrated in a web-based infrastructure in the future to enable the real-time aspect. Two possible applications for the integration of our navigation approach are presented. The emergency application can support people that quickly want to escape from a dense crowd for example during a music festival. Furthermore, a generic navigation application can help in various situations for example after a football game where certain routes are blocked by the police. Overall it can be stated that the advantages of our approach can be seen on both sides, for the event organizers and the guests. The guests utilize the tool to avoid stress and excitement whereas the organizers ensure the security during the event and increase the attraction by providing modern smartphone apps.

## REFERENCES

- Csurka, G., Dance, C. R., Fan, L., Willamowski, J. and Bray, C., 2004. Visual categorization with bags of keypoints. In: In Workshop on Statistical Learning in Computer Vision, ECCV, pp. 1–22. printed.
- Haarbrink, R. and Koers, E., 2006. Helicopter UAV for photogrammetry and rapid response. In: Proceedings of the 2nd International Workshop: The Future of Remote Sensing, VITO and ISPRS Intercommission Working Group I/V Autonomous Navigation.
- Jegou, H., Douze, M., Schmid, C. and Perez, P., 2010. Aggregating local descriptors into a compact image representation. In: Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on, pp. 3304–3311.
- Kurz, F., Türmer, S., Meynberg, O., Rosenbaum, D., Runge, H., Reinartz, P. and Leitloff, J., 2012. Low-cost optical camera systems for real-time mapping applications. PFG Photogrammetrie, Fernerkundung, Geoinformation 2012(2), pp. 159–176.
- Liu, L., Fieguth, P., Clausi, D. and Kuang, G., 2012. Sorted random projections for robust rotation-invariant texture classification. Pattern Recognition 45(6), pp. 2405 – 2418. Brain Decoding.
- Rosten, E. and Drummond, T., 2006. Machine learning for high-speed corner detection. In: European Conference on Computer Vision, Vol. 1, pp. 430–443.
- Schmidt, F. and Hinz, S., 2011. A scheme for the detection and tracking of people tuned for aerial image sequences. In: U. Stilla, F. Rottensteiner, H. Mayer, B. Jutzi and M. Butenuth (eds), Photogrammetric Image Analysis (PIA), LNCS, ISPRS, Springer, Heidelberg, Munich, Germany, pp. 257–270. printed, crowd.
- Vedaldi, A. and Fulkerson, B., 2008. VLFeat: An open and portable library of computer vision algorithms. <http://www.vlfeat.org/>.