LiveLayer – Live Traffic Projection onto Maps

Simon Walton and Min Chen[†] and David Ebert[‡]

Abstract

We present our work-in-progress for a novel new approach to visualising real-time traffic data. The system provides for the projection of a number of camera video streams to their corresponding projections on a map so that viewers may view live traffic in real-time. Along with a user-friendly method of defining the projections that does not rely upon inaccurate computer vision techniques, we also discuss a number of interesting visualisation and technical challenges involved in such a system.

Categories and Subject Descriptors (according to ACM CCS): I.3.6 [Computer Graphics]: Methodology and Techniques—Interaction techniques, H.5.1 [Computer Graphics]: Multimedia Information Systems—Video

1. Introduction

Many online mapping services currently provide a means of viewing geographical traffic information (such as SigAlert [Sig11], and Google Maps), and the data is usually represented as colours (red: heavy traffic; green: smooth traffic). But what if we show real, live traffic video streams on top of these mapping services as an additional layer by utilising the camera feed already available in a city's infrastructure? We aim to investigate the interesting visualisation and interaction challenges that arise from such a system.

2. User Interaction

The user interface of the system, developed using Qt, is shown in Figure 1. The user can drag a camera feed onto its real-world location on the map, which opens the *projection editor*. This feature allows for the registration of the camera's feed with the associated coverage area on the map.

We have judged for automatic registration methods to be impractical for the purposes of registering the video stream with the map projection due to differences in lighting/shadow, infrastructure locations, foliage, etc. There are many examples of systems that allow for manual projection definition in the literature (such as [MBCM97]) for these reasons. Our system offers simple tools to enable quick registration, with only around 30 seconds required for a basic road.

Map data ©Tele Atlas Imagery ©2011 Bluesky, Infoterra Ltd & COWI A/S, GeoEye, Infoterra Ltd & Bluesky, The GeoInformation Group

Figure 1: *The main interface. Left: The camera wall. Centre: The map view. Right: The projection editor.*

2.1. Defining the Projections

The user first draws two lines of projection over the video. The roads are then defined either manually (*freehand* mode), or using a simple template system that covers a variety of road structures (see Figure 2) and can be refined once in place. The choice of freehand verses template is the user's, and depends on the complexity of the road being monitored. As templates are selected, the system updates a visual road projection and places it over the video. The user can interact with the road projection by dragging control handles that define particular aspects of the road. Simultaneously, the system makes a guess at where to place this road on the map view. This is based upon:

- *Road information* – the system uses Google Maps' asynchronous directions service to obtain semantic data on the

[†] Department of Computer Science, Swansea University
[‡] School of Electrical and Computer Engineering, Purdue University

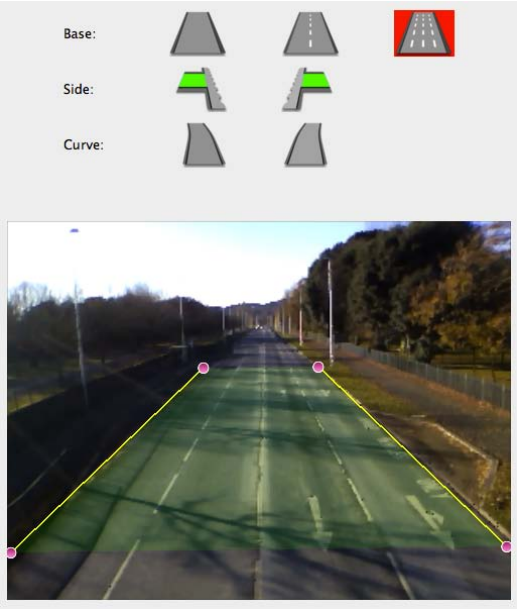

Figure 2: *The user selects a base template for the road (above, highlighted in red), and the system creates the road (below, shown between yellow lines and highlighted with green).*

roads surrounding the camera, effectively learning about the roads surrounding the camera;

- *Template information* – the road width is calculated based on the base template and a predefined constant representing the width of the average road lane. If the user has selected any side roads, these are also taken into account.

Based on this information, the system will stretch a road from the camera position along its chosen road on the map. If the guess is incorrect, the user can correct the guess by dragging the start/end of the road to a new road, which then snaps into place. The 'snap to road' functionality can be switched off also.

3. System Architecture

The camera video frames are continually fed into the renderer. Operating upon the camera frames is a separate worker thread that computes a background mask (to remove the background road from the input video, leaving only moving objects).

3.1. Rendering

The rendering system uses OpenGL to render the projected traffic streams. Camera frames are uploaded as a texture, along with the background mask obtained from the background model. The renderer draws the projected bounds of the road over the map view, and a fragment shader performs the backwards-mapping operation necessary to perspective-warp the video image texture in place, discarding texels where the background model has identified a background pixel. The homography matrix for the mapping is calculated using OpenCV from the user specification of the projection.

3.2. Background Model

A background model used in traffic situations needs to be resilient to changing weather/lighting conditions [CK05, Zhu09], as well as camera/tree movements due to wind. We currently use a per-pixel Gaussian mixture model to model the background, which we have found is able to recover well from changing conditions. However, sudden camera movements tend to reset the model temporarily due to the model being per-pixel.

The background model is multi-resolution; the system automatically switches between resolutions upon changes to the viewing parameters, refining the resolution of cameras with greater priority. This helps to balance CPU usage against visual quality depending on the current viewing parameters. The update frequencies of each camera's model vary according to these rules also.

4. Future Work & Acknowledgements

We are currently in the process of developing interesting visualisations of the traffic from the video stream by extracting semantic information from the videos on-the-fly, along with an improved background model that provides for improved results in areas where traffic regularly comes to a standstill, avoiding situations where such vehicles become part of the background. Additionally, we are investigating visualisations of the overall system using clustering algorithms to assist the user in identifying areas of interest.

This material is based upon work supported by the U.S. Department of Homeland Security's VACCINE Center under Award Number 2009-ST-061-CI0001.

References

[CK05] CHEUNG S.-C. S., KAMATH C.: Robust background subtraction with foreground validation for urban traffic video. *EURASIP J. Appl. Signal Process. 2005* (January 2005), 2330–2340. 2

[MBCM97] MCLAUCHLAN P., BEYMER D., COIFMAN B., MALI J.: A real-time computer vision system for measuring traffic parameters. In *Proceedings of the 1997 Conference on Computer Vision and Pattern Recognition (CVPR '97)* (Washington, DC, USA, 1997), CVPR '97, IEEE Computer Society, pp. 495–. 1

[Sig11] SIGALERT:. http://www.sigalert.com. 1

[Zhu09] ZHU F.: A video-based traffic congestion monitoring system using adaptive background subtraction. In *Proceedings of the 2009 Second International Symposium on Electronic Commerce and Security - Volume 02* (Washington, DC, USA, 2009), ISECS '09, IEEE Computer Society, pp. 73–77. 2