Supporting Mobile Applications with Real-Time Visualisation of GPS Availability

Anthony Steed

Department of Computer Science, University College London, Gower St, London, WC1E 6BT, United Kingdom
A.Steed@cs.ucl.ac.uk
http://www.cs.ucl.ac.uk/staff/A.Steed/

Abstract. Many mobile applications rely on the Global Positioning System (GPS) to provide position and location information. However, there are many problems with using GPS in urban environments due to the variable nature of GPS's accuracy and availability. This paper introduces a simple tool that visualises the current state of GPS availability in real-time. This tool can be used for scenario planning for certain types of mobile applications and as aid for analysis of location logs.

1 Introduction

Many mobile applications require positioning information [1], and those that operate outdoors often use the Global Positioning System (GPS) system. Applications that rely on GPS range from widely available guiding and mapping applications, through location-based services [2] to augmented-reality style presentations [3].

GPS technology is a rapidly moving field. However a consumer-grade GPS unit typically needs to be able to detect the signal of three or more of the GPS satellites in order to be able to generate a position. At the time of writing there are thirty-one GPS satellites in orbit that provide good coverage given an open sky. However, in urban environments, the skyline can have a very significant elevation. This restricts the amount of sky that can be seen and reduces the likelihood of being able to see the requisite number of satellites. A second complicating factor is that the satellites are in non-stationary orbits, so even if a GPS unit is in a static position, the GPS availability will change over time.

In this paper we introduce a tool, *satview*, which visualises the current likely availability of GPS coverage. The tool takes a 3D model of the local environment, and the satellites positions. In real-time, the tool shows where on the ground plane one would likely be able to see three or more satellites and thus reliably get a position fix. We developed this tool in response to two mobile application scenarios in the EQUATOR project. We will briefly discuss how we have used or plan to use satview to improve the effectiveness of users in these scenarios.

2 Visualising GPS Availability

GPS units usually report navigation information using the NMEA 0183 communications standard [4]. Aside from messages that give position estimates, NMEA 0183 supports a message that describes the satellites in view. This message gives the azimuth and elevation of up to 12 satellites that have been calculated to be in the sky above the unit, along with a signal strength for each.

If we have a 3D model of the area near the unit, we can easily visualise the visibility of any one satellite using a graphical shadow algorithm. If we consider that satellite to be a light source, then everywhere that is in shadow will be hidden to the satellite. Fig. 1. shows a 3D model and the areas of the ground that would be visible and invisible to one satellite. Our model was generated using the automatic process described in [5] which uses commonly available vector map data. A review of methods for generating urban models can be found in [6].

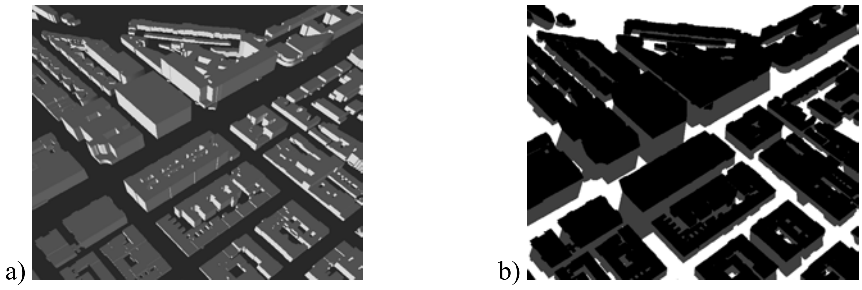

Fig. 1. a) Simple 3D block model of an area near Baker St in London, UK. b) Visualisation of the visibility of a single satellite in that area. White regions on the ground can see the satellite, grey cannot. The buildings are drawn in black

In our scenario, because we are interested in visibility on the ground plane, an algorithm such as the “fake-shadow” algorithm is sufficient [7]. This algorithm works by projecting all the scene geometry on to the ground. In a graphics API such as OpenGL [8] this can be done with a by rendering the buildings after pushing an appropriate projection matrix onto the modelling matrix stack.

In order to combine multiple shadows and find regions that satisfy various conditions on visibility, we use the stencil buffer that is available on most 3D graphics accelerators. The stencil buffer is a set of bit planes into which values can be written (refer to [8] for a detailed description). Like the depth buffer, the stencil buffer is not directly shown on the screen, rather arbitrary binary tests can be made against the stencil bit planes when filling pixels in other buffers.

For each frame we proceed as follows:

1. Clear the colour, depth and stencil buffers
2. Draw the city model and ground plane into colour and depth buffers.
3. Disable drawing to colour and depth buffer but leave depth testing enabled.
4. For all *i,* draw the fake shadows from satellite *i* into bit-plane *i* of the stencil buffer.

5. Read the stencil buffer back
6. Enable colour drawing. Disable depth testing.
7. For every pixel on the screen, if the value in the stencil buffer indicates that our visibility condition is met, then plot that screen pixel in white.

For Step 7 we have to interpret the bit mask returned in the stencil buffer as a visibility condition. For example, if there are N satellites, and we are interested in every point that sees three or more satellites, we need to plot all points for which no more than N-3 bits are set in the stencil buffer. Given that a maximum of twelve satellites are visible this can be easily implemented with a lookup table. Other visibility conditions can also be encoded in a similar manner. A minor implementation issue is that graphics cards vary in the number of bit planes they support in the stencil buffer. Eight is a typical value. This implies that the stencil buffer might need to be read and cleared multiple times.

Fig. 2. shows the visualisation tool itself.

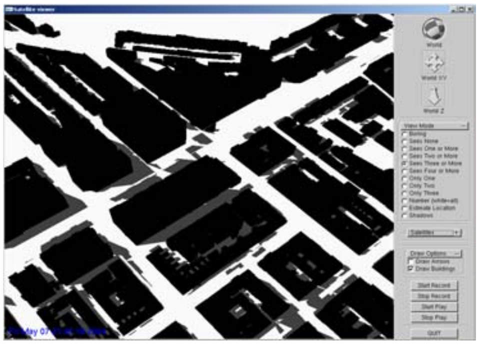

Fig. 2. The satview tool. All areas that can see three or more satellites are shown in white

3 Scenarios of Use

The development of satview tool was motivated by two scenarios. The first was the experience of colleagues who used GPS as the positioning technology in the Can You See Me Now? (CYSMN) game event [9]. In CYSMN runners in the real world were tracked using GPS. They had to “catch” online players who were navigating about a map of the same area. A catch occurred when a runner got a GPS position fix in the vicinity of the position of an online player. The runners would often experience *black spots*: areas where GPS position fixes were so inaccurate that they would be unlikely to catch nearby online players. This caused frustration for the runners. Inaccuracy would also cause frustration for the online players because they could get caught by one of the inaccurate position reports from a runner.

Designs for future interfaces for such events have proposed two strategies for dealing GPS uncertainty [10]. The first uses historical logs of GPS position fixes to identify likely black spots. The second uses satview to predict current black spots. The latter is only useful if it can be conveyed to the runners in real-time, either by presenting maps on the PDA, or relaying instruction via an operator.

The second scenario is concerned with the collection of logs of pollution levels. On the Advanced Grid Interfaces for Environmental e-science in the Lab and in the Field project we are investigating the use of GPS-tracked pollution monitors to make dense maps of pollution [5],[11]. In one study we are attempting to make a map of an area of London that is a case study for the Dapple project [12]. We are using satview to plan which regions to map in particular sessions, by noting whether east-west or north-south roads are more favourable and whether there are enough high satellites to survey narrow streets. Fig. 3. shows how GPS availability was predicted to vary over an hour at one road junction.

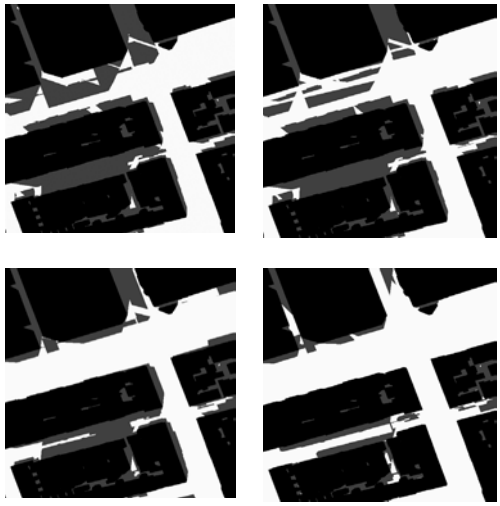

Fig. 3. Predicted satellite availability changing over an hour

4 Discussion

The satview tool provides a simple way of visualising GPS availability. At the very least it provides a useful tool for explaining why users' experience of GPS positioning is as varied as it is. It also provides a tool that can be used in certain scenarios to plan when to visit a region.

This is a work in progress, and in the near future we will add the capability to extrapolate satellite paths to predict availability over the next few hours. We are also looking at the requirements for detailed modelling of building heights. Our model of Baker St uses only crudely estimated heights but we do have detailed height models of other parts of London [5]. Note that with this tool we can't model the effects of reflected signals, nor of signal diffraction.

In the longer term, the tool might provide a method of improving GPS accuracy by exploiting knowledge about visibility and invisibility of satellites to provide a constraint on the position fix. It is worth noting that accuracy and availability issues are being vigorously attacked in the navigation community. In the longer term systems such as Galileo that are complementary to GPS will be launched and these will improve accuracy in urban canyons [13]. However we expect that a tool like satview would still be useful to detect potential black spots or regions of differing accuracy due to different numbers of satellites being available.

Acknowledgements

This work was supported by, EQUATOR Interdisciplinary Research Collaboration (EPSRC Grant GR/N15986/01) and Advanced Grid Interfaces for Environmental e-science in the Lab and in the Field (EPSRC Grant GR/R81985/01). Thanks to Richard Milton for generating the models of the Baker St area. Movies of satview are available from the author's web page. The software itself is also available on request.

References

1. Hightower, J., Borriello, G.: Location Systems for Ubiquitous Computing. Computer, 34(8), August (2001) 57-66
2. Cheverst, K., Davies, N., Mitchell, K., Friday, A., Efstratiou, C.: Developing a Contextaware Electronic Tourist Guide: Some Issues and Experiences. Proceedings of CHI 2000, The Hague, The Netherlands, 17-24 March (2000).
3. Feiner, S., MacIntyre, B., Höllerer, T., Webster, T.: A touring machine: Prototyping 3D mobile augmented reality systems for exploring the urban environment. Proceedings First IEEE Int. Symp. on Wearable Computers, October 13-14, Cambridge, MA (1997)
4. National Marine Electronics Association, NMEA-0183 Standard.
5. Steed, A., Spinello, S., Croxford, B., Milton, R.: Data Visualization within Urban Models. Theory and Practice of Computer Graphics 2004, 8-10 June, University of Bournemouth, UK (2004)
6. Shiode, N.: 3D Urban Models: Recent Developments in the Digital Modelling of Urban Environments in Three-Dimensions. GeoJournal 52 (3), 263-269
7. Blinn, J.: Me and My (Fake) Shadow. IEEE Computer Graphics and Applications, 8(1), (1988), 82-86
8. Shreiner, D., Woo, M., Neider, J., Davis, T.: OpenGL Programming Guide 4th Edition. Addison Wesley (2004)
9. Benford, S., Anastasi, R, Flintham, M., Drozd, A., Crabtree, A., Greenhalgh, C., Tandavanitj, N., Adams, M., Row-Farr, J.: Coping with uncertainty in a location-based game. IEEE Pervasive Computing, September (2003) 34-41
10. Crabtree, A., Benford, S., Rodden, T., Greenhalgh, C., Flintham, M., Anastasi, R., Drozd, A., Adams, M., Row-Farr, J., Tandavanitj, N. and Steed, A.: Orchestrating a mixed reality game 'on the ground'. Proceedings of the 2004 CHI Conference on Human Factors in Computing Systems, April 24-29, Vienna (2004).
11. Advanced Grid Interfaces for Environmental e-science in the Lab and in the Field (Urban Pollution), http://www.cs.ucl.ac.uk/research/vr/Projects/envesci/, accessed 4 May 2004.
12. Dapple, http://www.dapple.org.uk, accessed 4 May 2004.
13. O'Donnell, M., Watson, T., Fisher, J., Simpson, S., Brodin, G., Bryant, E., Walsh, D.: Galileo Performance: GPS Interoperability and Discriminators for Urban and Indoor Environments. GPS World, June (2003).