Augmented Reality platform for enhancing integration of virtual objects

Mohamed El-Zayat*
Supervised by: Selma Rizvic†

Faculty of Electrical Engineering, Sarajevo, Bosnia and Herzegovina

Abstract

With the wide spread of high end processors integrated in mobile devices, ranging from 1GHz processors, to dual core processors and hybrid processors (GPU and CPU on one chip), augmented reality became more popular solution for visualization and navigation. This paper proposes an augmented reality platform for organizing and enhancing integration of computer generated objects by introducing lights, shaders and shadows, in pursuing for better experience for the end user, emphasizing on outdoor environments.

Keywords: Augmented Reality, Platform, Enhancing Integration, Real Time sun Tracking

1 Introduction

Augmented reality (AR) is a relatively new and promising concept. The ability of superimposing digital elements on a physical world with means of interaction with the surrounding world is quite intriguing idea, since AR introduction in 1968 by Ivan Sutherland [4]. However, the technology by that time and for almost next 3 decades was quite limited to lab research, since the mobility nature of AR, and lack of capable mobile processors.

The rapid development of mobile GPUs, CPUs and recently hybrid processors, leads to an increase in popularity of the AR technology. Mobile devices play important role in AR technology as they combine processor, memory, display and interaction technology into one single device [14]. There are two main trends in AR research: registration, where researchers try to solve misalignment and world tracking problems; and integration, where researchers are directed towards the enhancement of computer generated object integration with the surrounding environment. This work proposes an AR mobile platform for enhancing integration of virtual objects in outdoor environments.

The rest of the paper is organized as follows: Section 2 gives an overview of the related work in the field; Section 3 illustrates the proposed AR mobile platform; Section 4 covers registration and Section 5 covers integration, proposing a real time sun tracking system for capturing the current lighting conditions of the environment. In Section 6 we present the algorithm. Section 7 presents results compared to other AR platforms. Finally, in Section 8 we conclude the paper and give some directions for future work.

2 Related Work

In recent years we have seen significant advances in two fields of user interface research: virtual environments, in which 3D displays and interaction devices immerse the user in a synthesized world, and mobile computing. Previous research in mobile AR has addressed a variety of application areas including 3D mobile AR systems for exploring urban environments [6], enhancing registration through making a hybrid registration for outdoor AR [1], improving teaching with mobile AR for learning and training [11], location based AR for indoor environments [9], enhanced computer generated objects rendering using environment illumination [7]. In pursuing better registration of AR objects, researchers are trying to combine computer vision with sensors for achieving more accurate results [5]. Additionally, a combined solution for illumination techniques for AR objects is discussed in [8]. One commercial platform that caught many mobile device users' attention is Layar AR browser [13] (Figure 1).

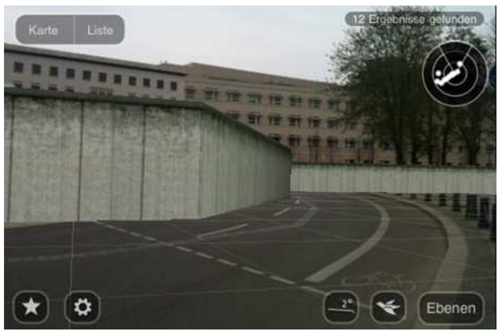

Figure 1: Layar Browser,the original appearance of the Berlin wall (image courtesy of layar.com)

This paper proposes the AR mobile platform for arranging AR objects with an emphasis on enhancing the integra-

*mohamed@fit.ba
†srizvic@etf.unsa.ba

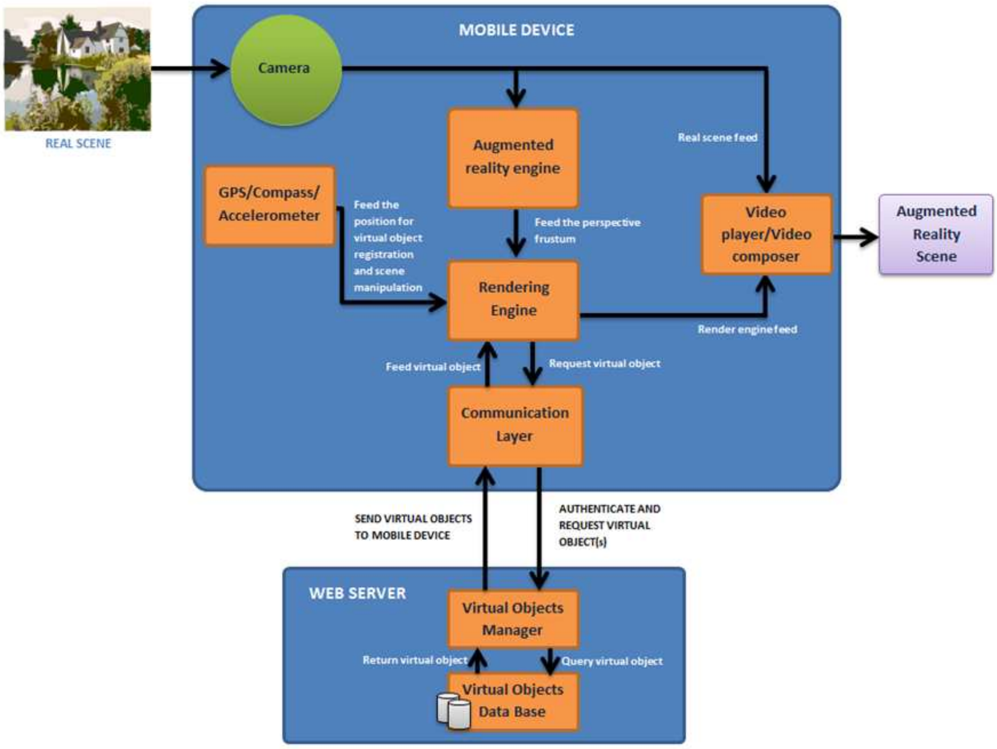

Figure 2: AR Mobile Platform architecture

tion of computer generated objects in the outdoor environment, by introducing lights and shaders to the augmented objects.

3 AR platform

Making AR systems that work outdoors is a natural step in the development of AR toward the ultimate goal of AR displays that can operate in any environment [2]. A user, walking outdoors, could see spatially located information directly displayed over live camera stream, helping to navigate and identify features of interest [1].

While designing the platform, the mobility and optimization factors were taken into consideration as described in the upcoming sections. Figure 2 shows the overall architecture of the platform.

In order to achieve higher performance and decrease the disk usage by 3D models and their associated textures, all 3D models are stored on a web server, and will be downloaded on the mobile device once the user is near the location where the virtual object should be displayed.

Comparing current revision of the proposed AR platform architecture (figure 2) and Layar architecture as shown in Figure 4, more support to developers and content providers is given from Layar, in contrast with the presented AR platform. Although the developers support was not considered in this current revision yet, one of the improvement ideas over Layar is saving 3D objects in relational SQL database as shown in Figure 3. This gives the opportunity to permitted organizations to run analysis and data mining techniques using off-the-shelf software to collect more data from users to further enhance the public service at certain most visited areas, thus further enhancing the proposed AR platform.

3.1 Server Side

A database containing 3D models along with the associated textures, and additional lighting data for improving the integration of the virtual object, resides on the server side. Lighting details are covered in Section 5.

The Virtual Objects Manager (VOM)is a web service responsible for handling authentication and requests to the database. This component would be crucial in case this platform is implemented for a mobile provider. In case that authentication is not needed, VOM will add the security layer needed to protect models database. 3D models are stored in a relational SQL database as shown in Figure 3:

3D models are indexed using their GPS coordinates. Geometry table contains the actual geometry data of the 3D model, stored as an array of doubles for each of Vertex, Vertex Normal and Texture Coordinate. Furthermore, the texture field holds compressed texture atlas for the geometry.

The illumination data (i.e. Artificial light) of the objects are stored in Lighting Data table, where color and position of the light are stored as array of doubles.

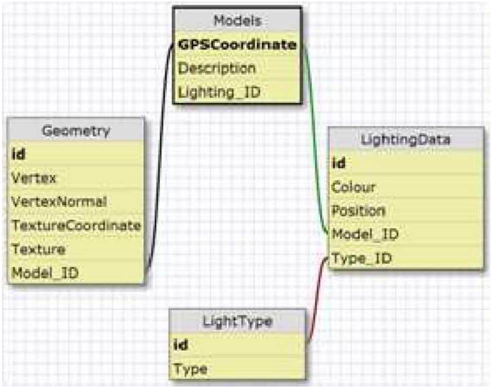

Figure 3: 3D models database

Organizing augmented objects in a relational SQL database provides the opportunity for other mobile devices/platforms that support connecting to SQL databases to benefit from the proposed AR platform. Therefore, it is possible to port the AR platform to wider variety of mobile devices.

Using the server side for managing virtual objects will relieve the users from constantly updating and downloading the complete application once 3D models get updated, thus increasing the performance and saving unnecessary storage load.

3.2 Client Side

This platform requires a mobile device that is GPS capable along with at least accelerometer and compass. Since most mobile device vendors are integrating these sensors as a standard in their devices, we believe that in few years these devices will be common among users.

Mobile devices contain two major components: registration and integration components. Registration component is responsible for AR registration, which could be done using a complete sensor based registration (i.e. combining GPS, accelerometer and compass data as discussed in Section 4), or hybrid one as discussed in [1]. A hybrid registration could combine some elements of computer vision and sensor data to improve the integration of computer generated objects. Therefore, the registration system could handle unstructured and unprepared environments [5], and in this case, the AR engine will be activated in the augmentation pipeline.

Once the user gets to a desired location detected by GPS, where a virtual object resides, registration component will generate the frustum that will be handed to the rendering engine. Rendering engine will send a request to the communication layer to load 3D object and its associated data from the server side, thus rendering the virtual object and ”clearing” the background with the camera feed, hence superimposing the virtual object over the physical world.

4 AR Registration

In order to enhance the integration of augmented objects, improving registration is required. GPS data is required to determine the position of the virtual object in the physical world and the position of the user according to the position of virtual object, hence calculating the position of the frustum according to the physical world using equation 1:

$$(x_2 - x_1)^2 + (y_2 - y_1)^2 < r^2 \tag{1}$$

x_1, y_1 represents user position, while x_2, y_2 represents the virtual object’s position. r is the range value .If the user is in the range of the detected object, he/she will be notified and the frustum will be generated.

In order to detect the rotation of the user, a compass will be used for azimuth rotation direction, and the accelerometer will control frustum altitude as shown in figure 5:

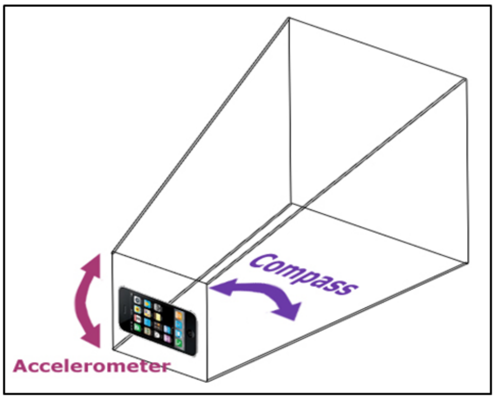

Figure 5: Frustum controlled by sensors

The registration is sensor based, similar to [14] and [13], in pursuing for saving CPU/GPU cycles to enhance the rendering of augmented objects, hence improving the integration. One of the registration types that are crucial for displaying correct imagery in augmented reality is precise alignment between the projected image and the features on the display surface [16]. Therefore, the elevation of the 3D object is determined from elevation data provided by the GPS, in addition to the developer adjustment to that value, because of the inaccuracy percentage of GPS in mobile devices.

5 AR Integration

There are several factors that have to be taken into consideration for outdoor rendering of augmented objects. One of these factors is lighting, which is a crucial component in rendering any object in a scene.

In order to improve the integration and create a realistic scene, AR platform should track sun position in real time, thus approximating lighting conditions of physical objects along with their shadows. Therefore a directional light is used to simulate sunrays.

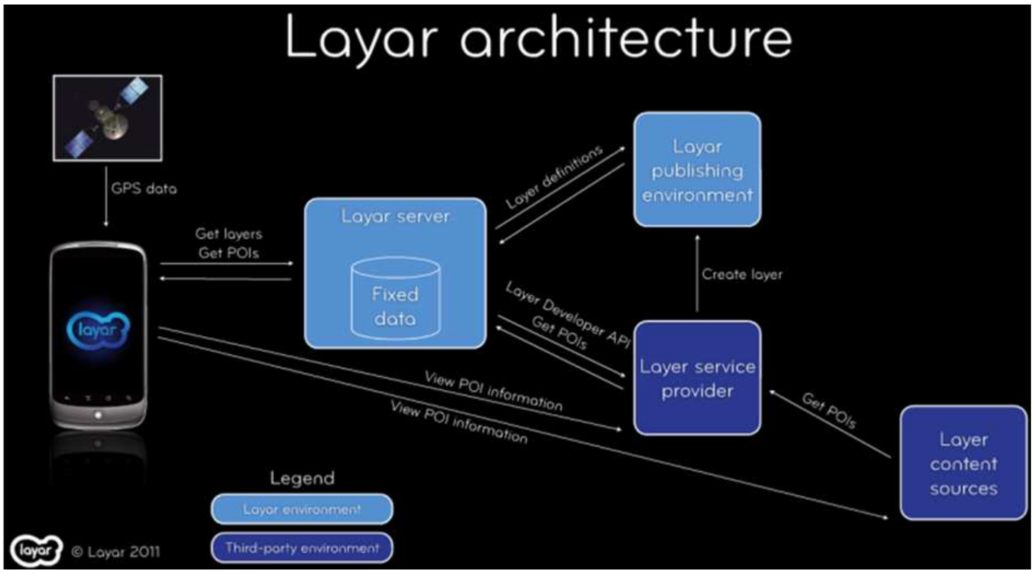

Figure 4: Layar Architecture as presented in Layar Developer Conference 2011 (image courtesy of Layar.com)

Earth is relatively spherical celestial object that rotates around itself eastwards every approximately 24 hours and around the Sun approximately every 365 days. The axis on which Earth rotates is the Polar axis. The great circles that intersect with the Polar axis are called meridians. The great circle equidistant from the North and South Pole is the equator [12]. Earth rotation axis is tilted by 23.4, which results in changing the relative position of the Sun as the Earth moves in orbit. This change reflects on the angle of the Sun rays according to the equatorial plane. This angle is called declination.

Figure 6 shows the Sun position towards the Earth along with the above described angles.

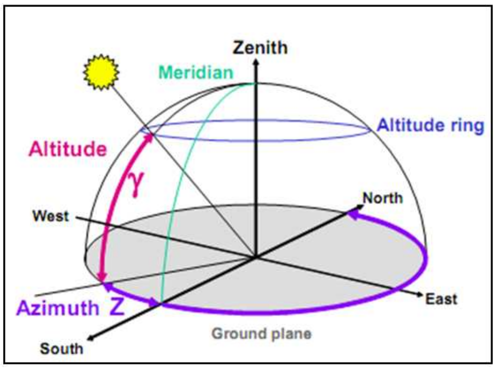

Figure 6: Sun position towards the Earth [12]

γ is the altitude of the Sun above the ground (horizon) plane and z is the azimuth, which is the compass direction of the Sun on the ground plane. Declination is calculated using equation 2:

$$Declination = 23.4 \times sin(\frac{360 \times (284+N)}{365})degrees \quad (2)$$

Symbol	Variable	Definition
D	Declination	The angle of the Sun rays to the equatorial plane, positive in the summer.
L	Latitude	The angle from the equator to the position on Earth's surface
H	Hour angle	The angle the Earth needs to rotate to bring the meridian to noon. Each hour of time is equivalent to 15 deg.
N	Day number	The day number, January1stis 1.

Table 1: Azimuth Altitude equation legend

where N is the number of the day for which the declination is being calculated, January1st being day number 1. The Azimuth may be expressed in two ways: either as the angle clockwise from North or as the angle East of or West of South. Although the former is most often used, we used the latter convention. Azimuth and altitude of the sun can be calculated using the following equations [12]:

$$sin\gamma = (cosD \times cosL \times cosH) + (sinD \times sinL) \quad (3)$$

$$cosz = \frac{(cosD \times cosH \times sinL) - (sinD \times cosL)}{cos\gamma} \quad (4)$$

Table 5.1 shows the legend for the above equations:

The color of the Sun rays plays an important role in displaying the time of the day, and also determines the color temperature of objects. This problem can be ap-

Source	RGB(0-255)	RGB(0-1)
Sun at sunrise or sunset	182 126 91	0.71 0.49 0.36
Direct sun at noon	192 191 173	0.75 0.75 0.68
Sun through clouds/haze	189 190 192	0.74 0.75 0.75

Table 2: Sun color at different times of the day

proached by implementing atmospheric scattering algorithms to change sun and atmosphere's color depending on the time of the day. However in order to decrease the CPU/GPU load, and increase the frame rate, Sun color could be determined through basic hard coded RGB values [3] as shown in table 5.2:

Using those values, the RGB numbers between the two stages of the sun during the day could be extrapolated based on the starting time of the simulation.

Another factor that has to be taken into consideration is the night time, where no sun or any light source is available except the presence of the Moon, at a certain times of the month. In real life, the moonlight illumination is almost unnoticeable in urban or artificially illuminated areas, thus the moonlight factor will be neglected in this case (Figure 7).

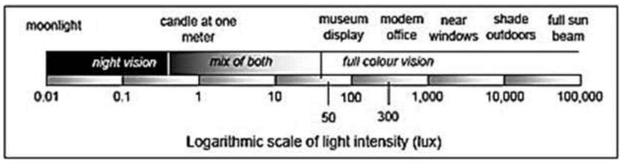

Figure 7: Logarithmic scale of light intensity (image courtesy of Canadian conversation institute)

In order to compensate the loss of Sun light, Rendering Engine component queries for artificial light that is associated with the desired model from database on the server side. The artificial light data contain position, light type, and the diffuse components. At the early dawn or late sunset, where Sun's illumination is not strong enough to illuminate the augmented object, the directional light is turned off, and the queried artificial lights are activated.

One possible way to speed up lighting calculations is by performing the light map, which is especially useful when used in conjunction with multi-texturing. Additionally, texture baking could be used as an alternative technique for increasing the frame rate [10].

6 Algorithm

In previous sections, registration, organization and integration techniques used in this AR platform were discussed. This section discusses the pseudo flow of events that are triggered in the AR platform, starting with user authentication and ending by augmenting the 3D model on the user's screen.

1. Authenticate user.

2. Get all GPS coordinates from server.

3. For every x period of seconds, check if user is at a close distance from a target area.

4. If user is close to a target, get permission to download the 3D object from user.

5. Create frustum based on the gravity vector extracted from the accelerometer and get rotation angle from compass.

6. Start calculating light position.

6.1 If it is day:

6.1.1 Create directional light resembling sun light.

6.1.2 Rotate light in .. axis for .... degrees.

6.1.3 Rotate light in .. axis for .... degrees.

6.1.4 Set directional light diffuse RGB values based on sun diffuse RGB lookup table.

6.2 If it is night:

6.2.1 Get artificial light data from server.

6.2.2 Create artificial lights based on their types.

7. Augment 3D object on the video stream.

When working with mobile devices, it is important to optimize mobile - server communication frequency. Therefore, after user authentication, all GPS coordinates are queried and stored in a list on the client side of the AR platform as shown in line 1. Thus it is not necessary to connect to the server unless the user permitted the client to download the target object as in line 4, or when it is night time and the client downloaded the artificial light data from the database as in line 6.2.1.

For every user-defined period of seconds, the application checks if the user is in the radius of the target location (i.e. line 3). When user confirmed to download the 3D object, the rest of the calculations are carried on the mobile device to minimize the communication frequency with the server.

Sun rotations around x and y axis are calculated using formulas 3 and 4.The altitude and azimuth of the frustum are derived from the accelerometer and compass data. Finally after all calculations are done, the object is rendered on mobile device's screen, and the frustum is updated based on the GPS, accelerometer and compass data.

7 Results

Figure 8: Layar Augmented 3D object at noon (image courtesy of Layar.com)

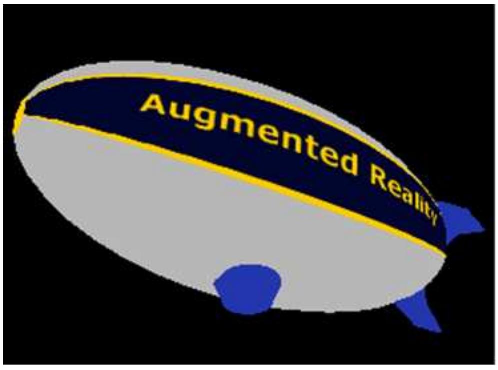

Figure 9: Default OpenGL Light (i.e. OpenGL lights not enabled)

Figure 10: Direct Sun at noon (image courtesy of Layar.com)

By the time of writing this paper, tests have been running under iPhone emulator, and since computers video cards can handle OpenGLES2.0 with almost no frame rate loss, as a result, frame rate data will be neglected for the time being. Current models are using per-vertex illumination, and the shaders component of this platform is still under development. As a consequence, directional light and spotlight with insignificant diffuse values for smoothing shadows have been added. However, the loss of per-pixel illumination does not highly affect the outcome; an image displaying the color temperature of the models at different times of the day is shown in Figure 8 compared with Layar render for the augmented 3D zeppelin model:

Initially, comparing Figure 8 with Figure 9, that Layar AR Platform did not include any lights for illuminating its 3D objects. Nevertheless we could see a dramatic change even on per-vertex lighting when comparing 10 and 11 renders. It is worth noting that comparing Figure 10 and 11 carefully, the latter has lighting temperature close to the camera feed and almost has the same sunray direction. Color temperature of objects during noon could have a bit of yellow tint to it, as shown in Figure 10 on the building bellow the zeppelin and the test zeppelin render (Figure 11). Figure 12 shows the zeppelin model under sunset lighting condition, while for the cloudy or hazy weather as showed in 14 light color tends to be white. We believe that, completing the shader component we will gain better results. After several tests, one could disagree with the

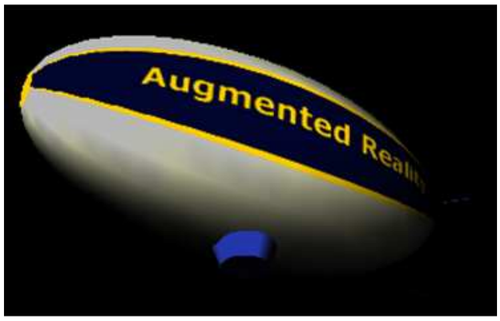

Figure 11: Test render direct sun at noon

Figure 12: Sunset

approach proposed by Birn for using same values for sunrise and sunset [3]. The reason is that during sunrise (i.e. dawn), objects' color temperature tends to be cold, thus has blue tint to it as shown in Figure 13. Therefore, one could suggest adding an additional RGB value for sunrise. We suggest that these values would be (0.50 0.49 0.60) for R, G and B, assuming that RGB values goes from 0 to 1.

8 Conclusions and Future Work

In this paper a solution for AR organization and integration problems, in a context of generic AR platform is presented. This solution deals with two major topics: integration where virtual objects are illuminated according to the time of the day, by tracking sun position in real time, thus estimating the correct color temperature and shadows; and organization where all objects are organized in a relational SQL database along with its illumination data on remote server. 3D objects are downloaded upon a query from the client side (i.e. mobile device), hence superimposing the downloaded object after light calculations are finished.

The presented platform is still under development, though several features could be implemented, such as calculating length of shadows, compensating for rainy, snowy or cloudy weather, where most of the objects tend to have no shadows since light distortion is very high and the object is illuminated almost from all sides. Implementing several lighting techniques, such as light mapping or texture baking could assist in increasing the frame rate.

Expanding database to include the ability to subscribe to

Figure 13: Sun at Dawn

Figure 14: Cloudy/Hazy weather

the web server could encourage developers and/or clients to upload their own content. Furthermore, database could be expanded to contain other types of multimedia, such as audio, that could be associated to a certain 3D model.

Other improvements to this AR platform would include giving the ability to developers and content providers to add to and enhance the platform. Current calculations for predicting Sun position still lack a very important parameter, which describes the accurate position of the user on Earth. Therefore Sun position would be calculated with a higher precision.

Another important factor that could affect the experience of the users is the quality of the camera in their mobile device. Since modern mobile devices are equipped with autofocus and/or have automatic exposure correction, one possibly would notice the difference of color temperatures and light intensity between the augmented object and the physical world. This platform could not fully enhance the integration if augmented objects exist in urban areas, as when the augmented object's position has special geological properties, or it is surrounded by other higher objects that cast shadows on it.

References

[1] A hybrid registration method for outdoor augmented reality. In *Proceedings of the IEEE and ACM International Symposium on Augmented Reality (ISAR'01)*, pages 67–, Washington, DC, USA, 2001. IEEE Computer Society.

[2] Ronald T. Azuma. The challenge of making augmented reality work outdoors. In *In Mixed Reality: Merging Real and Virtual*, pages 379–390. Springer-Verlag, 1999.

[3] J. Birn. *Digital Lighting & Rendering*. New Riders, 2006.

[4] S. Cawood and M. Fiala. *Augmented Reality: A Practical guide. The Pragmatic Programmers*. 2007.

[5] Lin Chai, William A. Hoff, William A. Hoff (corresponding, and Tyrone Vincent. 3-d motion and structure estimation using inertial sensors and computer vision for augmented reality. *Presence*, 11:474–492, 2000.

[6] Steven Feiner, Blair MacIntyre, Tobias Hllerer, and Anthony Webster. A touring machine: Prototyping 3d mobile augmented reality systems for exploring the urban environment. *Personal and Ubiquitous Computing*, 1:208–217, 1997. 10.1007/BF01682023.

[7] Masayuki Kanbara and Naokazu Yokoya. Real-time estimation of light source environment for photorealistic augmented reality. In *Proceedings of the Pattern Recognition, 17th International Conference on (ICPR'04) Volume 2 - Volume 02*, ICPR '04, pages 911–914, Washington, DC, USA, 2004. IEEE Computer Society.

[8] Saulo A. Pessoa, Eduardo L. Apolinario, Guilherme de S. Moura, Joao Paulo S. do M. Lima, Marcio A. S. Bueno, Veronica Teichrieb, and Judith Kelner. Illumination techniques for photorealistic rendering in augmented reality. In *SVR2008*, 2008.

[9] Gerhard Reitmayr and Dieter Schmalstieg. Location based applications for mobile augmented reality, 2003.

[10] P. Ridout. *iPhone 3D Programming*. O-REILLY, 2010.

[11] Reiner Wichert. A mobile augmented reality environment for collaborative learning and training. In Margaret Driscoll and Thomas C. Reeves, editors, *Proceedings of World Conference on E-Learning in Corporate, Government, Healthcare, and Higher Education 2002*, pages 2386–2389, Montreal, Canada, 2002. AACE.

[12] M. Wilkinson. *Building Enironment 1*. University of Bath.

[13] www.layar.com.

[14] Suya You, Ulrich Neumann, and Ronald Azuma. Orientation tracking for outdoor augmented reality registration. *IEEE Computer Graphics and Applications*, 19:36–42, 1999.